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: SuperGeeks

Team members: Tim Giroux, Gustavo Valencia, Kyle Austria, Pengfei Liu

Course number and name: CS486C Capstone Experience

Semester: Spring 2022 Date this reflection completed: 3 May 2022

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 planned a comprehensive sprint schedule with user stories for each sprint. During each sprint, team members took on one or two user stories per week. Some sprints were done in parallel, which involved splitting the team in two to work on different sprints.

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?

This process worked well because team members could collaborate and take on work based on their individual strengths and weaknesses. It fostered a steady development pace and a good learning environment.

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

We realized at the end of the sprint schedule that our user stories did not get us to full coverage as per our requirements specification. It would have been better to be more careful in writing user stories to make sure they covered everything.

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 actually 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 source.

  Vim, Atom, and PyCharm were used for creating and editing Python and HTML files.

- Version control: How did you manage your codebase?

  Git was used for version control. The codebase was managed via a GitHub organization with a dedicated repository for the project.

- Bug tracking: How did you keep track of bugs, who was working on them, and their status

  Bugs were always very high priority, and we would assign them as user stories. The progress and status of these bugs were tracked via a dedicated text channel on our preferred communication platform, Discord.

- UML modelers and other miscellaneous tools:

  Draw.io was used for UML modeling, as well as for making other types of diagrams and charts. Additionally, we used Google Drive to organize team documents, as well as Google Docs and Google Sheets for collaborative word processing and schedule making.

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?

This went well. Our tools were very simple industry standards, and the problems we faced weren't complex enough to dive into more complex development tools.

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.) up front? Or was it more just "everyone does everything as needed"?

The team leader was usually in charge of meetings and assigning internal deadlines. The team scholar was usually responsible for having a detailed understanding of class deliverables and deadlines. Other roles filled in as needed and everyone shared responsibilities.

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

- Regular team meetings? If so, how often?
  Three meetings per week for two hours each.

- Impromptu team meetings? If so, roughly what percent of total team meetings were of this sort?
  Rarely, but we used additional time as needed. This would account for maybe 2% of all team meetings, and they were almost exclusively held remotely via Discord.

- Emails to all members? If so, explain briefly: about how often, what used for?
  No, but emails to clients and our mentor generally had all team members CCed.

- 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 pull requests and code reviews were a large part of our workflow.

- Other communication channels used? Facebook, wiki, text messages, phone conferences, etc.
  Discord was indispensable for team communication. We communicated through Discord nearly every day.

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.

Team communication was very successful because of how often we had team meetings. There was very little confusion about deadlines or responsibilities.

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.

We probably still could have been just as effective with shorter or less frequent meetings. This would have required more asynchronous communication, but it could have freed up our schedules and reduced some stress for team members.

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!