NAU–CS Team Project Self-Reflection Worksheet

Team Name: *Untitled Developers*

Team members: *Herbie Duah, John Loudon, Michael Ortega, Luke Sanchez*

Course number and name: *CS486c - Capstone Experience*

Semester: *Spring 2016* **Date this reflection was completed:** *05/04/16*

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 followed the SCRUM approach in which we met once a week as a development team (Wednesday's at 5:30pm) along with meeting with our mentor at the start of every week (Monday at 2:30pm). We also included our client in the development process, as we took a user centered design approach to collect feedback bi-weekly.

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?

Yes, all relevant stakeholders were updated periodically on the progress of the website. This method worked because we were able to remain individually productive throughout the rest of the non-meeting week.

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

The only change that we can think of would be to use an effective tool for bug tracking, as we used multiple different channels for communicating bugs (Google docs, Slack).

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.**
    - Sublime: main text editor
    - Bracket: Text editor mostly used for the styling because it showed the changes in real time
    - Gulp
    - Sketch 3: Used to create initial mock-ups
    - Marvel: Prototyping tool
    - Google MyMaps API: Used to create interactive maps

  These tools were used in their appropriate contexts for their various uses. Sublime and Brackets were used for editing things like PHP, Javascript, and CSS. Gulp is a task runner that we used to compile .scss files into a single .css file. Sketch and Marvel were used in design. And the Google MyMaps API was used to create the three interactive maps that were a key requirement for the site as a whole.

- **Version control: How did you manage your codebase?**
    - We used Github to manage our codebase

  Through the use of git and github we were able to share a single code base and all make changes as needed. It also allowed for quicker deployment to the server through the use of SSH.

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

  Slack was our primary communication channel within our team. It allowed us to have a semi-permanent record of what issues we had been finding and allowed us to determine who was going to work on those issues as well as their status as time went on. Google Docs was used to keep track of tasks on a weekly basis. We used a single google doc to track who was doing what tasks from week to week (based on Wednesday's team meetings).

- **UML modelers and other miscellaneous tools:**
    - Draw.io for architectural, graphical diagrams

  Draw.io was used as a quick way to put together architectural diagrams, primarily for presentation purposes.

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?

We didn't really use any tools that we didn't need, however we could have used an actual issue tracker like Trello rather than just trying to keep everything straight manually.

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"?

We never formally assigned roles to the team members. We identified tasks that needed completing and assigned tasks to group members based on their strengths and weaknesses.

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

Regular team meetings? If so, how often?

We had a team meeting every week, apart from our mentor and client meetings.

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

We usually had impromptu meetings when there were holidays. The impromptu meetings would probably only account for 5% of our total meeting times.

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

We all rarely used email for communication between developers. We used email to communicate with the other stakeholders (client and 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?

Slack was the primary communication channel that we used to communicate issues.

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

We used Slack a lot because we could share files and create different channels for different topics of conversation.

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.

Intra-team communication was good for the most part because we had our team meetings on the same day and time every week. With other meetings we usually communicate a lot to make sure if they are still happening or where we are going to be and so on. There wasn't any major breakdowns.

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 could use a professional task management / issue tracker (such as Trello) to better organize our weekly sprint goals. We could have also spent more time and energy formalizing our architecture as the project developed.