*SCRUM*ptious

A more palatable SCRUM 👋🍪

Seneca Park Zoo Tigers

Change Log

Date	Modified By	Comment
09/10/2019	Nikolas Tilley	Created initial document from meeting notes.
09/19/2019	Team	Filling out details of sections.
09/23/2019	Nikolas Tilley	Clarifying effort measurements
10/07/2019	Nikolas Tilley	Adding links to artifacts

The Seneca Park Zoo Tigers participates in a slightly modified proprietary version of SCRUM with an Iterative and Incremental software delivery model, while also shifting some focus away from the documentation and ceremony that true SCRUM requires.

Why *SCRUM*ptious

As we were not handed a large document with clear and well defined requirements, we decided that we needed a methodology that allowed for maximum flexibility while keeping the project on course, and effectively managing risk. Most of the requirements will come through face to face communication with our client.

Features

2 Week Sprints

Each Sprint is made up of Sprint planning, the actual sprint, delivery, and the Retrospective.

From historical data, 1 week sprints proved to be too short. The ratio of deliverables and ceremonies to time worked just did not make sense.

Daily (Meeting Day) Standups

At the start of every meeting, we go over what we did since the last meeting, what we are currently in the process of doing, and any concerns or risks we feel should be tracked. This information is aggregated into the Four-Up Charts.

Grooming

We are going to groom our sprints with a modified version of planning poker, in order to determine the most accurate pointing for each task. We have defined our pointing system as follows: 1 point: 1 to 2 days of work; 2 points: 2 days to 1 week of work; 3 points: 1 week of work; 5 points: 1+ week of work; 8 points: 2 weeks (full sprint) of work. Anything deemed to be greater than at 8 will be split up into smaller subtasks.

We consider “one week” of work to be ~8-12 person hours (in the context of the senior project).

From there, we will use our velocity to determine how much work we can pull into a sprint, and groom our sprint tasks based on the number of points, our availability, and the tasks’ priority in the backlog.

Burndown Charts

Burndown charts will help us plan for sprints more accurately as the project progresses. The chart will be examined after each sprint to determine our velocity. This allows us to make an educated estimate on how many story points to allocate for the next sprint.

Our burndown charts can be found here in a google sheet.

Retrospective

At the end of each sprint, we will have a retrospective meeting to discuss things that went well, things that need improvement, and recognize individual accolades and accomplishments. This will help us improve each week and make improvements in both our process and at an individual level, as well as give recognition to excellence.

Our retrospectives can be found here in a google document.

Four-Up Charts

Four-Up charts are a lightweight artifact to track a week’s Progress, Next Steps, Risks, and Needs. A new Four-Up is created every Tuesday after our meeting with the project sponsor. With the sponsor, we review the previous week’s chart to keep them informed of our progress.

Our Four-Up charts are compiled here in a google document.

Metrics

Velocity (VoT)

Velocity is the number of story points completed during the previous sprint. Velocity allows us to estimate the amount of work to commit to while planning the next sprint. Average velocity can be used to give the customer a best case, worst case, and expected amount of value added by the end of the project.

Burndown

Burndown is the number of completed story points during a sprint. This metric is closely related to velocity. Burndown is often tracked using burndown charts to visualize the completion of sprint backlog items over the duration of the sprint.

Our Burndown charts can be observed in this google sheet.

LOCs (Lines of Code) per Sprint

LOCs will be tracked to average the amount of development productivity.

Acceptance Test Coverage

Acceptance test coverage is the approval of features via manual testing by our client. This will provide both verification and validation.

Tools

Github

Github is used for our source code management. We will push new code to a feature branch, then make a pull request to the master branch. To be merged in, the creator should post a link in slack and the pull request must be approved by two other people.

Jenkins

Jenkins in a CI/CD tool that will act as a build server for us and automatically run tests and deploy code.

Slack

Team communication tool. We have channels for different groups, ex. #rit for everyone on campus, #jenkins to view the output of the latest build, #se-accounts to store credentials, etc.

Trello

Backlog and task management tool. We will use this to visualize the status of individual tasks.

Meetings

Meeting agendas should be made and sent out ahead of schedule. Sent out by Nikolas Tilley, the scribe.