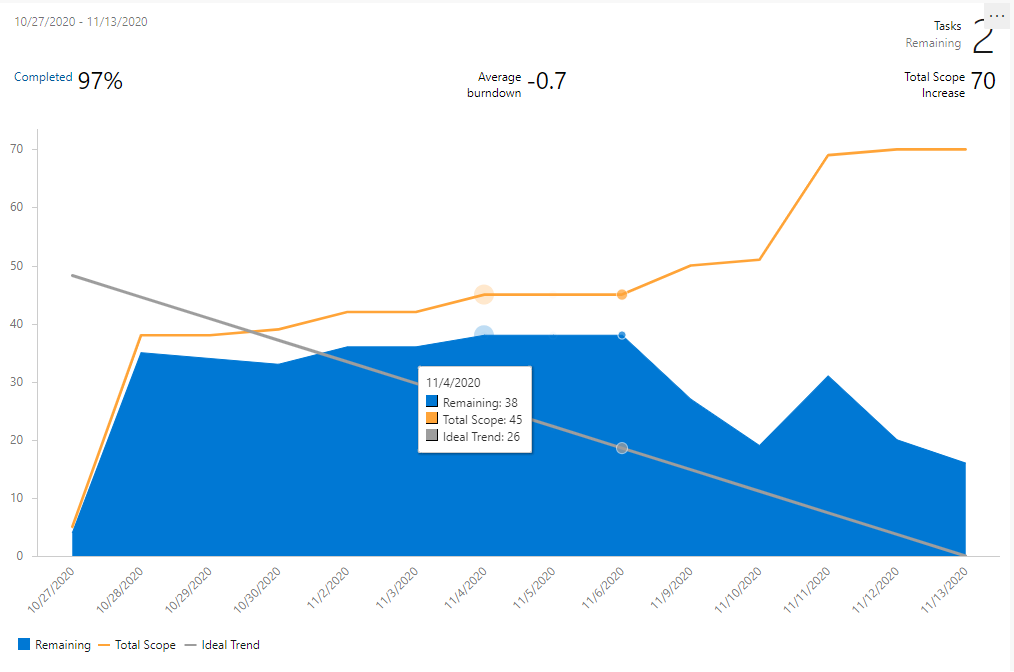
**Sprint 4 Reflection**

**Team Number: 4**

Hunter Figgs, Jared Perkins, Sergei Fedulov, Patrick Haughn, Jeffrey Gaydos

CSE 3902, Professor Boggus

26 October 2020



Summary:

Compared to Sprint 3, Sprint 4 got off to a much quicker start. We assigned tasks very early on allowing us to begin development earlier as well. Along the way, we had several meetings where we tracked each member's progress to make sure we are on track and in addition added a few more tasks as more thing's bugs/features came up. As far as the implementation goes, our group is very pleased with our game state implementation, which helped us create winning, losing and room transitions states. In addition, the new audio engine and HUD made the game feel much more polished, responsive and complete. We also worked on adding a few other minor details such as a “puzzle” mechanic that allowed us to open closed doors and spawn items depending on a rooms condition, as well as a random item dropper to more closely replicate the real game. Regarding code quality, we re-worked the collision detector and made a task to remove all magic numbers from the code. Overall, we were much more proactive for this Sprint allowing us to have fewer bugs, more features, better implementations and in general better code quality.

Planning and Documentation:

Learning from issues with the previous sprint, the team began by ensuring that many small tasks were created as soon as possible. Also, the team started early for the same reasons. However, throughout the sprint it was found that many more tasks were needed to polish up and create all the quirks of the game. As shown in the burndown chart, the number of tasks added to our board nearly doubled. Part of this drastic increase has to do with the lack of polish completed on previous sprints, and the idea that polish could be pushed to further sprints. Despite the double in workload, the team maintained a burndown as close as possible to 0 and finished all allotted tasks and more.

In the future, the team may decide to start with assigning polish tasks, and then move on to what new features need to be added. Additionally, considering that the team has only 5 members, the team plans to first produce a finished game with all the typical requirements for a 6-person team, and then move on to some extra features after this completion.

Implementation:

In this sprint, the HUD, room transitions, and audio were the biggest additions. The room transitions also came with the addition of a Game State interface. The Game State uses the Monogame Sprite Batch methods in order to transitions between the rooms. The HUD required the addition of sever new sprites and an IHudItem interface for the items displayed in the HUD. The HUD can be controlled with a controller, mouse and keyboard. Another large addition was the audio. Music and sound effects were added using Monogame’s Sound Effect. In addition to these major changes, we also fixed several bugs such as enemies getting knocked through walls, enemies not having death particles, and the speed of enemies. Also, we were able to add the “puzzle” mechanic that allowed us to open closed doors and spawn items depending on a rooms condition, as well as a random item dropper to more closely replicate the real game.

Code Quality:

Since we got a much better head start on this sprint compared to last sprint, we had more time to sit down and analyze the code we were writing, and the code we had written, and clean it up and make it more maintainable and readable. The team looked through the code and removed most of the magic numbers and cleaned up classes we decided were long and hard to read. Most of the new code was designed with the ability to add new features in mind, while doing our best to retain the original functionality of the game.