

Operation: Stop Core

Post-Mortem

GAM 450 – Spring 2005

What went right:

1) One thing that really helped us out was deciding to use a scripting language for the core game engine. Specifically, we used Python for the engine and all the game logic. There were several benefits in using Python for the game engine. Built into the game is a script console, which allows us to test and debug while the game is running. We could tweak with parameters or object attributes and immediately see the results of those changes. Another advantage to using a scripting language is that it requires no compile time. We can change some bits and pieces in the Python scripts without having to recompile the project. This was a bit time saver.

2) Another thing that turned out very good is how the game engine fits together in its entirety. From the outset, we wanted OSC to be cross-platform which required that we carefully select the libraries and APIs to use. For our game we use SDL for input, SDLmixer for the audio portion, OpenGL for the graphics, and Python for the game logic. The process of incorporating all these technologies together went very smoothly. It ended up creating a very robust game that can run on many different machines (it runs all the machines that we have tried so far).

3) We were able to create some useful tools for creating the game world. In particular, the World Editor tool we created was quite useful. It was fairly easy to use and allowed us to construct the world very quickly once we had all the objects and items for the game. It allowed us to add rooms, link up the rooms via doors, place objects, place items, and place lights. It made this part of development run a lot smoother. Investing the time to create good tools is well worth the effort.

4) The look and feel of the game seemed to turn out pretty good. We got some good feedback on the look of character's and other models. They fit the game well. People who have played or seen our game in action seem to chuckle when they see the characters or hear the dialog. This is a very good sign since OSC is intended to be humorous.

What went wrong:

1) The biggest problem we faced was losing a teammate halfway through development. This particularly hurt us in the area of content creation. The teammate we lost was scheduled to create a large percentage of the models in the game. With his leaving the project we had to reassign this responsibility along with his other responsibilities. So, we spent a lot of time working on portions of development we had not originally expected.

2) Another major problem we had was failing to stick close to the original timeline. This was partly due to us losing a team member, but it probably would have happened anyways. When we got behind, we didn't reevaluate the timeline. As a result, we had to cut out some of the features we had planned to include. We originally thought to implement realistic lighting and shadows, but we ended up having to scrap that idea once it became evident that there simply wasn't enough time.

3) The scope of the game also ended up presenting us with a problem. There were just too many models and other art assets that we had to create. We spent vast amounts of time just creating the models, textures, etc. This took away from the time that might have been better spent on the programming aspects of the project.

4) Finally, the last significant problem we had in development was that it took us a long time before we could test content within the game itself. This was due to us not having a completed game engine until halfway through development. Also, we scheduled the World Editor tool to be implemented during the last half of the project. So, it wasn't until roughly 75% into the development cycle that we had the game world built. And only once we had the world could we start populating the game world with objects and such.

What we would do differently:

There are several things we would do differently if given the chance. First of all, we would have completed most, if not all, of the game engine during the summer before our senior year. This is really necessary for a game that has a lot of content. The reason for this is that you need to be able to load up content such as models and test them in the actual game engine. This way you can see early on whether the look and feel of the game is going to work. Secondly, we would have scaled back the original design. We thought that if any problems arose, we would just alter the game a bit and still make it work. This isn't really possible with a content intensive game. It becomes even more of a problem if the changes occur well into development. Any changes made to the design will possibly require that new or different content be made and cause some of the content that exists already to be thrown out. It just results in a lot of wasted time and effort. Lastly, we would have done a better job scheduling the tasks necessary for us to complete the game. The original timeline was not well thought out and did not take into account possible risks like the losing of a team member. We would definitely have liked to enhance the game a bit more visually with realistic lighting, shadows, and other effects.