Reflection

by Brad Buchanan

Our fifteenth week is all preparation for the end of the semester – getting ready for our final presentation, our final demo, the hand-off to the OCCO, submission to competitions and more. And so, we'd like this our final newsletter to be a reflection on a few lessons we've learned over the past fifteen weeks, and what we will take forward to our future projects.

1. Schedule aggressively
We were especially fortunate this semester because on day one our client established this pattern for us. They told us the game had to be feature-complete by halves, which sounded like an impossible goal to us at the time. But we got right to work, and it's amazing how much time you don't waste when you're under a tight deadline from day one. We responded by setting equally tough deadlines for ourselves during the semester, and as a result the team surprised ourselves with how productive we could be.

2. Assign roles
One reason our team worked well together is that we assigned clear, distinct roles to each team member. Not just one role per member either (that's insane on a small project) but each person had at least six titles giving them authority and responsibility for those parts of the project. Not only did this help us know who to look at when a question or task arose, it also helped us not overlook any part of the project – it's hard to say "oh, someone else will do that" when it's only in your job title. Ironically, the one role we forgot to assign was sound designer, and it had to be picked up by two people late in the semester.

3. Designing for the unknown is hard
In week two when we pitched three game ideas to our client, we still had only the roughest idea of what the TV would be able to do. We knew it could handle 2D graphics, and had seen only a very simple 3D demo running on it. We had no idea what kind of trouble we would encounter with the GL layer, or how harsh the video memory limits would be. Knowing what we know now about the hardware, the team agrees we would have designed a totally different game.

4. Two is too many
This isn't a hard-and-fast rule. In fact, we're quite pleased with what we were able to do with the two different platforms working together. But having two platforms introduced a number of challenges that would rarely come up in a single-platform project. For instance, we didn't anticipate what a mess we were getting into when we used Unity Asset Server to manage our Unity source and Perforce to manage the TV source. Two version control systems is definitely too many! Another challenge was building the same game world in two different game engines, then trying to keep them completely synchronized over the network. This introduced a lot of headaches we could have avoided with a different game design. If you're working on two platforms, look out for these pitfalls!

5. Don't save polish until later
We panicked a little during our first week when we learned that we were supposed to be feature-complete by halves. We agreed as a team that the first half of our semester would be devoted to building gameplay and playtesting, and that the second half could be implementing content, polish, and other small changes we needed at the time. We learned that some polish is absolutely critical to game design – it's the details that communicate important information to the player, and make the game playable and enjoyable for them. We should have been polishing the game as we built. Our playtests might have been more useful that way.

Thanks for following our project this semester! We hope you've enjoyed reading about our game half as much as we've enjoyed building it! From all of us here at BarrelEye, have a great holiday and keep your ears open, since we'll be entering some competitions in the near future. Thank you!

Team Member _ Bradley Buchanan, Brian Lee, Romain Deciron, Haewon Nam, Jaewan Park, Eren Song, Neerav Mehta
Advisor_ Carl Rosendahl, Jiyoung Lee