Darkstar Post Mortem

Overview

Project Darkstar is an open source project administrated by Sun Microsystems. It can be summed up best as follows: “Project Darkstar is a software platform that simplifies the development of horizontally scalable servers for online games, virtual worlds, and social networking applications. Its revolutionary design will eliminate serious problems like zone overloading, data corruption, and inefficient server utilization, while enabling new dimensions of play such as evolvable virtual worlds and very large scale battlefields.” – Project Darkstar website (www.projectdarkstar.com)

The Entertainment Technology Center (ETC) project “Darkstar” will be making a number of games over the duration of the semester in order to test drive the Project Darkstar system and providing feedback to Sun about what is good, what is not so good and how we think they can further improve it. Further, we will be providing the Project Darkstar server and a python/Panda3d client interface to the students currently taking Building Virtual Worlds in order to get more perspectives and additional feedback on the system. Further we will also learn how easy it is to teach to others, an important usability factor in the real world. Out last goal is integrating Project Darkstar into the Panda3d to provide the engine with networking capability.

What went right

We finished a Massively Multiplayer Online Game (MMOG) framework. This is something so many people getting into the industry say they are going to and then never actually do beyond a token effort. And actually building a framework with a group of other people is an incredibly rewarding experience as very few younger people have practical experience working on MMOG frameworks, let along building and designing it themselves. Further, our framework is scalable and we know for a fact that it will work for at least 25 people, so 32 and 64 are easily foreseeable for just about any game. Combined with a MMOG like our demo, it can likely scale to thousands, which is pretty good considering it was built in under a semester.

We learned network programming. We all gained valuable experience and insight in how to do network programming. What makes this so valuable is that networking programming is one of the hardest game programming tasks to perform.

We integrated Panda3D and Project Darkstar. Panda3D has been cited as having three main weaknesses: visibility culling, level editing, and networking. So with our project at a close we can say that this has made Panda3D a stronger and better engine for everyone. And then for Project Darkstar, it had never been integrated with a game engine proper, you could use it with anything and the APIs were there, but it was not actually part of another engine. Add in the fact that both Project Darkstar and Panda3D are open source and you have a good match and starting point for anyone looking to start working on networked game without having to buy/lease or integrate themselves, the heavy lifting has been done.

We deployed to the Building Virtual Worlds (BVW) class at the start of their "Round 2". Getting a package that could be deployed inside that early in the semester got the Panda3D integration off to a strong start. Thanks to this, by the end of the semester we had a nice simple package for Project Darkstar installation/integration into Panda3D without much trouble or rush.

Client was absolutely thrilled. When we asked for feedback in terms of things going right and wrong, they had nothing but positive feedback and superlatives. In the three semesters I have been at the ETC and of all the feedback I have heard, I had never heard feedback as positive as this. This was critical for us as our project was a client project, so in addition to everything we did for the ETC, we always have to make sure that the client is still getting what they want.

What went wrong

We spent too much time learning network programming. It is hard to say "We should have learned that faster", but this is one thing that went wrong. We took about seven weeks or almost the entire first half of the semester to learn network programming which left us only eight weeks to make our MMOG framework and game. We were able to reuse some of the material from the first seven weeks, but in the end, code reuse had a negligible impact on the schedule.

BVW students did not use Project Darkstar. This went wrong primarily for two reasons, the first of which is simply that Project Darkstar is overkill for the networking required in BVW worlds. Most worlds don't require networking and even those that require it usually only need to send a few messages between clients, so Project Darkstar's conflict resolution, persistence and other amenities are way beyond the needs of the average BVW student. This reason this was a problem is that in BVW, the students simply do not have enough time to make their world in addition to learning the Project Darkstar API. The other reason is that the code we developed and provided was our own project code which was designed and written with a MMOG in mind, not for simple message passing which is what the BVW students would need.

We did not seek help with art soon enough. By the time we sought help with art assets, everyone was busy with project work and there were no artists to spare. While we did not need art for our demo as it was intended on being a technical demo, some better art assets could have alleviated some of the pressure that forced us to stress the technical aspects and avoided continually having to ask people to look beyond or behind the visual aspects of the demo. This was a bit of a blessing in disguise in terms of making us focus on the technical aspects of our own demo, as they would have to be rock solid if they were the only thing that was well done in the demo. But I doubt this benefit outweighed the downside of "programmer" art.

We did started play testing and stress testing late in the semester, so we did not get in many iterations with feedback from both players and the server. Also were not able to stress test the server to its limits, partially because we were not expecting it to scale so well and partially because we did not plan enough time to both test the server, and write a proper dummy client to test the hundreds and thousands levels of scalability.

Lessons Learned

Backend framework will always take longer than you expect. Three weeks into our front end work schedule we were still working on the backend because when working on the front end, you realize that the backend needs to do something you had not originally thought of so that the front end can work.

Conclusions

All of our code is in the client's hands if they wish to add any aspects from our server code officially to Project Darkstar. The official Panda3D integration (having this in a release version) is moving forward and will be finished by the team after the semester ends as all that is left is the administrative details. Also, the code will be available as a starting point for anyone wanting to work on their own MMOG, pitched project or not through our project website.