SWERG Project

Benjamin Weidner

Online Multiplayer Game Dev Support for the Rowan Game Dev Club

Context: Game Development and the Rowan Game Dev Club

The game development club at Rowan has worked on many projects, some are completed, but many cannot be finished within the span of one semester considering the fluctuation of membership (graduation, upcoming finals, etc.). I would like to focus this project sponsorship on a project that would be able to assist any club member of the game development club across semesters.

The game development industry is not just game developers. We need software developers as well: the people who make what the game developers use to make games!

Any individual can create a game from a programming language they choose, but development of a game is time consuming, tedious, and much more difficult than many assume. This is where we introduce programming frameworks like Pygame: a module for Python that allows for easier development of games.

However, with this approach the developer needs to set up each tool themselves to create a game in an efficient way (aka, not-individually-programming-every-single-little-aspect-of-the-game-from-scratch). Say, in Pygame, game developers have the capacity to generate images and move them on screen, but there is no "level editor" for tile placements. In fact, a tile system may not even be implemented. The developer must create these tools themselves, working off the programming framework, to create a collection of game development tools in a single program or directory to help the developer create their game in an easier manner.

Imagine if the Mario developers needed to code every single Goomba in the game from scratch. We already did it once! As software developers supporting game developers, the task at hand is to make the game development process as seamless and efficient as possible. Utilizing a collection of programming tools to create a game is what we call a game engine. Unity2D is the game engine of choice for the game development club given its ease of use with new users and huge amounts of professional and amateur tutorial content available online.

Context: Online Multiplayer Game Development

The main idea of this semester's project is to work towards a way to adapt developing multiplayer games (or already existing multiplayer games) into online multiplayer systems. If a game has local multiplayer, translating this functionality into an online system is not as easy as one would think. Local multiplayer (otherwise known as "couch co-op") must be programmed with intention. As opposed to online multiplayer functionality, in many cases, if a game is not programmed to start with this functionality in mind it will never have an online multiplayer option.

On the flip side, there are numerous online games with massive numbers of users logging in and playing every single day. World of Warcraft is one of the most well-known examples of an MMORP (Massive Multiplayer Online Role-Playing Game) where people from across the globe log in to play. League of Legends is another famous example of an online game where users log in from around the world to play, the only difference is the genre: MOBA (Multiplayer Online Battle Arena).

Consider the first paragraph talking about games made with local multiplayer, now consider the second paragraph talking about games made from massive online audiences. Games with local multiplayer, even without some online multiplayer adaptation, can still be played locally on the couch with your friends, for example Mario Kart. But let's say you don't have any nearby friends that can meet up to play games in a physical space, you would most likely rely on online multiplayer to be able to play Mario Kart with your friends. However, these online games require server support; financial burdens to companies where the player count has diminished.

Let's look at a game like World of Warcraft. This game could be played with just your friends, and many do while ignoring the other players that exist in the world that log in from around the world. What would happen if Blizzard, the company that owned World of Warcraft, suddenly went under? Thousands of players would lose access to a game they may have been playing their entire life.

I grew up playing a game called UB Funkeys, a game where you would collect in-person figures to then connect to a terminal and access the online world and gain new functionality. After the company went under, I was unable to log in and play this game that I could very well enjoy on my own. And now, all those figures and the terminal sit in a box in my basement. This feels like a terrible waste involving physical product, online software, as well as really tearing away some heartfelt moments for the players involved. Imagine if one day you could no longer play your favorite sport because the field was not managed, it would be devastating.

But what if players could manage the field themselves? This is what is colloquially known as a private server. Many games allow players to host their own lobbies for their own players. And even after some games lose online support, there are dedicated communities that figure out ways to program and include private servers for people to connect to so they may play these online games without having to rely on the original company's servers for connectivity. This is where the terms "client-side" and "server-side" differ. Many of these fan-adapted games are hosted by the client, the people playing. Whereas when operated and owned by a large company like Blizzard, people connect to the server that Blizzard hosts and manages.

Regardless of the engine, company, or team, many developers will start games from game jams, passion projects, or for any other reason to create some interesting games. Many of these instances are not ones where the developer is looking far into the future. In fact, most of the game's design is usually not fully complete when a development environment is being set up. It is much easier to develop a game with local multiplayer than it is with online multiplayer. What if that transition could become more seamless, regardless of whether the game was complete, incomplete, or if online support was severed completely?

The core goal of this project is to give the Rowan Game Development Club a tool that will allow for the ease of development of online multiplayer games. For the sake of the software developers, we will not go into modding already completed single player games into multiplayer games. This would require learning all the ins and outs of the ways the single player game was developed, which is usually not feasible for someone who is not a superfan of a given game (ex. Skyrim Multiplayer Mod). **We want to give a game that already has multiplayer functionality the ability to either be brought or returned to online functionality.** The code for League of Legends still exists without Riot's server support and can still theoretically be played by yourself with bots. The game simply cannot be played online with other people.

We want to focus on games that are either in-development local multiplayer, already complete local multiplayer, and already complete online multiplayer where server support has been severed. Further discussion will decide which exact method will be chosen by the SWERG team.

This does not necessarily mean that Rowan will host the server to play on (client-side vs server-side), but the Game Dev club should be the one who has more control over seeing the connectivity of the players in the game (aka, this project should try its best to not

rely on a third-party company to host the game, as that would cost the club money that it does not have).

Here are some examples of what a game developer may use this tool for:

- Adapting a couch co-op GameCube game into having online functionality (ex. Gotcha Force, Mario Kart Double Dash, Kirby Airride, Smash Bros Melee)

- Reviving a game without server support so it may be played in a private server, whether it be by yourself or by inviting others personally (Lego Universe, U.B. Funkeys, Concord) *I am using Concord as an example of a game that was so bad it lost support in a short time. However, some people may have wanted to play it and the developers of the game deserve to have some player base enjoy their game, even if it is unofficial. I'd want people to play my game no matter what!*

- Adding online multiplayer functionality to a game already in development (The game dev club was working on a Smash Bros clone a few years back. Allowing that functionality to seamlessly be transitioned to online multiplayer would have been a big motivation boost for the club.)