Postmortem: Sniper Studios' Crazy Taxi Fare Wars

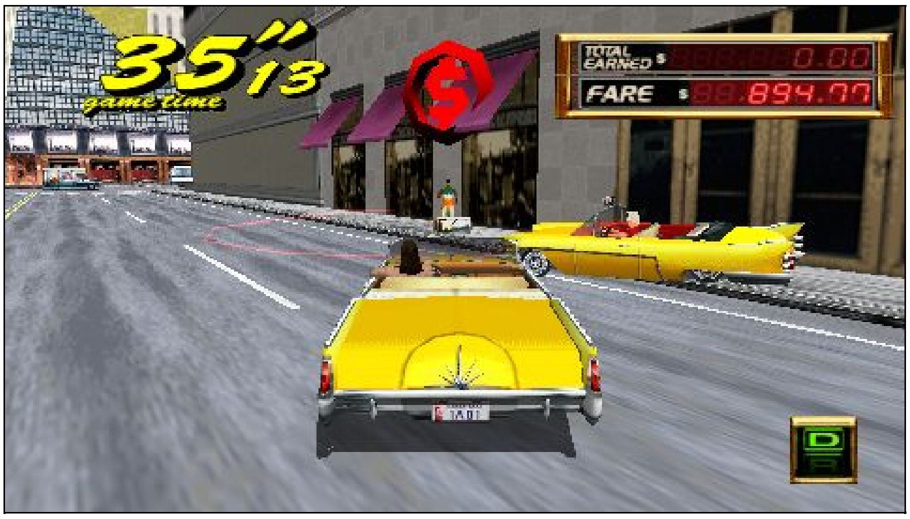
By Jeff Hasson

[In this Gamasutra-exclusive postmortem, Jeff Hasson of Redwood City-based Sniper Studios discusses the developer's experience in creating Crazy Taxi Fare Wars, a PSP version of Sega's arcade and console game series, with fascinating technical detail on the game's creation.]

When discussions began with *Crazy Taxi Fare Wars*, we spent a lot of time going back and forth on what the final packaged product would contain. Was it going to be a straight port? Should we have a career mode? In the end, we settled on two key elements: 1) to bring over a true port of both *Crazy Taxi* and *Crazy Taxi 2* and 2) to add multiplayer gameplay modes.

Ultimately, we all agreed that this was a sound plan. *Crazy Taxi* in our minds had near-perfect gameplay, and bringing that same game to the PSP made a lot of sense. Adding multiplayer would give the game a new feel, and provide a significant enough challenge to engage the programming team.

The game would be built on a global scale, literally! We would be working with Black Hole Games in Budapest, Hungary. So in the end, we were counting on source code from Japan, published out of Sega in San Francisco, multiplayer developed in Redwood City, CA, and ported in Budapest, Hungary. Constant communication would be key to the completion of the project and for the most part this happened.



In a perfect world, bugs would be delivered at the end of the day in North America to hand off to the team in Budapest at the beginning of their day. Inevitably, there were days when this didn't happen and days would be lost. Like any project, many things went well and many things didn't go as planned. The following is a highlight of our development cycle.

What Went Right?

Partnership – In Sega we found a great mixture of friends and people that we have worked with for many years in our careers. Additionally, it helped to have our publisher only 20 minutes up the road. While we probably didn't take advantage of that enough, it was always an option. The team in Budapest was also a team we have worked with in the past. Since this was their first PSP project there were many things that could have gone wrong but they dealt with many of the curve balls that were thrown at them. Their availability, talent level and cost effectiveness really helped make the project possible.

Budget – Predicting the budget was a difficult task. In essence, we were asked to give and schedule a budget that was based off of source code that we had not seen yet. By the end of the project, our budget was only 13% higher than our original guess and this was due to circumstances out of our control. This was not a game where we were going to make any money and in fact would take a small loss. But in the end it was worth it to put out a solid game and add to the legacy of *Crazy Taxi.*

Staffing – Like the budget issue, it was difficult to predict what team size we would need to develop the game without being able to see what the source code entailed. This was a game that was not built for multiplayer and the complexity of the squeezing these features in proved to be no small task. However, the lead programmers at Sniper have nearly 20 years of experience each. With occasional contracted support, our staff size proved to be the right size.

Design – We all agreed that doing a direct port would not be enough to even satisfy the most hardcore *Crazy Taxi* fan. Throwing in *Crazy Taxi 2* definitely made it seem better but still not taking advantage of the new platform. Adding a single player campaign was very intriguing but it would have taken even more time and money to develop balance and tune. Multiplayer seemed like a big challenge but worth it for the player. The final piece of the puzzle was the custom music player so that fans could bring in all of their favorite songs.

Testing – Both the Sega of America and the Sega of Europe test teams did a phenomenal job. In general they would run "around the clock" teams so there always seemed to be coverage on the project going on. With the different time zones to manage on this project, it could have been a limiting factor and even caused some delays. But due to the constant coverage we received quick and accurate information about our bugs with very few exceptions.

What Went Wrong?

Looking back over the development cycle for *Crazy Taxi Fare Wars* PSP there were several key areas that posed technical challenges not only to the performance of the game but even threatened our ability to finish the game over all. There were difficult hurdles with two cabs in multiplayer, framerate issues in all modes, having both *CT* and *CT2* on the same UMD, streaming data off the UMD and using the default Sony game state for Ad Hoc.

Multiplayer – The first topic to mention is the multiplayer modes of the game and specifically the Head to Head mode. We took the original Dreamcast source code from Sega of Japan and used it as our basis for porting. From a design standpoint, multiplayer was the next logical step in the evolution of the franchise, but re-engineering the original code to support two independently controlled cabs in the same space, on the same map, at the same time, proved to be a difficult challenge.

The original source code layout for the collisions and physics of the cars was spread out into several areas, making the task even more technically challenging. Our engineers were perplexed by this layout and knew that it would take too long to try and completely track down all of the effects that a single change might make to either part of the code.

So overall it was decided that we would not attempt to change any part of this code, both to keep the original feel of the game as true as we could and also so we didn't accidentally create bugs that we would then never be able to track down. This isn't a complaint against SOJ's Dreamcast code; it's just a fact that the original code was never intended to support two players at the same time.

The first step to getting the second car working correctly was making both the cars aware of each other on the map. Next was making the AI cars correctly spawn and track to each user-controlled cab. We ran into many issues with memory and framerate problems here too, especially in *CT2,* where the player cars are all loaded individually, as compared to *CT,* where the four player cabs/characters are all loaded from the same file at the same time. One of the few limitations of porting *Crazy Taxi* to the PSP was having to compromise on the amount of AI traffic in multiplayer. In the single player game you'll notice that there is much more AI car traffic around your cab compared to the head-to-head mode.

The PSP didn't have enough memory to handle many more than five AI cars per player-controlled cab without severe framerate slowdowns. So if both players are within sight of each other on the map, then there should be approximately ten AI cars total in the world. These AI cars had to be made aware of their physical location in relationship not only to the cab that they were created to be around, but also the other cab and its AI cars as well.

UMD – Another area that proved to be challenging was the fact that we had two complete games on one UMD. In the original design for *CTFW* we technically had four complete games with the addition of the multiplayer code for both*CT* and *CT2*. We were going to have the multiplayer modes available from the main menu, but this quickly became a problem with load times because the entire multiplayer code is based off the single player code.

If we wanted to play a *CT2* head-to-head game, we had to first load the*CT2* single player game code. The reality of having a multiplayer button on the main menu was that the user had to wait for the single player game to load anyway, so it seemed like it was taking forever to launch a head-to-head game. We decided to have the single player game executables on the main menu and then add the multiplayer game options to the existing GUI for each version of *CT*.

Although we ended up not having a direct route to the multiplayer modes from the main menu, we actually slightly improved the load time of getting into a multiplayer game by keeping these modes in the game menus and off of the main menu.

Streaming data off the UMD also proved to be a difficult issue for us to nail down. We were on a compressed schedule to try and finish *CTFW* as quickly as possible. The lead times to request and obtain testable UMDs burned from SCEA were prohibitively long, so we didn't get any UMDs made until very late in the development cycle.

The theoretical assumptions we made early on in the design of streaming the data weren't tested until just before we were preparing to submit to SCEA for approval. What we found were long load times and decreased frame rate that we had to adjust for. You'll notice that if you use the custom music player instead of the in-game music, the performance of the game improves a bit. This is due to the fact that the custom music player streams the data from the Memory Stick and not the UMD, thus the seek times for the rest of the game data improved.

Wireless Connectivity – We found that using Sony's default game state for establishing Ad hoc games wasn't as stable inside our codebase as we had hoped for. Most of the problems that we encountered were not apparent until we started getting feedback from Sony on communication interruptions that they were encountering at their testing facilities -- that we couldn't reproduce.

Making the adjustments to handle more robust communications error checking was a bit of a stab in the dark because we weren't ever sure of the results until we received updated feedback from Sony on these changes.

It's challenging to fix bugs when you can't reproduce them in-house. We had been warned, and had spent a considerable amount of time on the Sony PSP developer boards trying to anticipate the scenarios under which Sony was going to test the Ad hoc communications. In the end, we made the communications more robust to handle a wider range of possible communication interruptions that we never encountered at our studio.

Source Code – During the planning phases of the project, we wanted to get*Crazy Taxi* up and running before tackling the sequel. It took several months and was completed on schedule. When we turned our attention to getting the *CT2* source code up and running we encountered unexpected problems that inevitably created delays in our schedule.

In retrospect we should have spent time up front to do due diligence on the *CT2* source code; if we had, we would have been aware of these issues months earlier. But as they say, hindsight is 20/20. The net result was a hit to our schedule and a small adjustment to our budget. In the future we will handle this differently, by purchasing longer licenses for software development packages as well as increasing the amount of time to properly evaluate any codebase we use.

Communication – Yes, we knew from day one that this would be key to our success. In the end, it is still tough to execute when you are dealing with teams working together but separated by half the globe and a different language. We planned on code integration between multiplayer and single player right after first playable. While that went pretty much as planned, we were not prepared for the problems that came with bug assignment. In case of *Crazy Taxi*, many bugs were not distinctly multiplayer or single player.

That being the case, how did we decide who was responsible for fixing a bug? Would that fix impact the multiplayer/single player game? Many bugfixes worked for fixing immediate issues but would impact work being done by the other team -- when changes made were not clearly documented. Although the bug information was coming in steadily, it became a full time task to manage that data and disseminate it to appropriate team members either in Budapest or Redwood City. It became another layer to bug management that demanded a lot of communication.

Return to the full version of this article