

# Operation: Make Faster Game (OMFG)

Hey folks,

It's been a while since my last optimization post, so want to make sure you know that we're going to be posting a lot more often than we have been.

## **WHY HAVEN'T YOU DONE ALL THIS ALREADY?**

We've spent the better part of two years with PlanetSide 2 measuring and reviewing the elements of the game which are slow. When we've found things we could fix quickly, we did. When we found general optimizations that didn't have much of a risk to them, we worked on them. But mostly, given that we were a live game since the opening of the PS2 Tech Test, we kept our optimizations as "safe" as possible. Optimizing often involves reorganizing code that has already been through rigorous testing, and that means as we move it around we can...Bust something...No matter how hard we try not to. Many optimizations that we found just required far too much "soak time" to be feasible at that stage of the project.

## **WHY ARE YOU DOING THIS NOW?**

PlanetSide 2 has reached the point in its development where we think the core game is THERE. Designers have a vast array of options from here on out to create content, artists have at least a minimum set of tools for creating art, the UI works well and is understandable, we have an actual tutorial (no more "Welcome to the game! \*POW\*!"), and players around the world are able to find PlanetSide 2 and play it in their native language as available.

We have reached an opportune time, and recognizing this, Smed stepped up and said, "Fix it, now." We all have wanted to get the time necessary to do some of these bigger changes, and do them right. Smed also recognizes fully that no matter how much work the design department or art department puts into making the game shine, it simply isn't as fun of an FPS experience at a low frame rate.

With this proclamation, all of our teams have been given the freedom to fix things that take significant time to fix. What you're going to get at the end of this is a serious, noticeable increase in frame rate, PLUS some ancillary benefits that are capable as a result of some of these architectural advances. Every single member of the team, no joke, is actively looking for and implementing ways to make the game faster so we can deliver a better gameplay experience to you.

## **ALRIGHT ENOUGH APPETIZER, GIMME SOME MEAT!**

I'm going to introduce you to some fairly technical terms. If you're Vanu, you can skip this disclaimer and you probably know it all anyway. TR, if you don't understand something ask your C.O. If you still don't understand, you're probably a Heavy...don't worry about it... If you're an NC, all this is probably just some plot by the man to keep you down. Just read it anyway.

## **Physics**

-----  
This game has physics scenes that are unrivaled in their complexity. While a normal game may have upwards of a hundred different shapes bouncing around and solving for their collision, we sometimes have scenes with over a thousand shapes, or more! Those huge tank battles really do a number on the physics solver!

So OMFG will see us spending significant time restructuring the complexity of our physics solved in a way we call “Adaptive Level of Detail.” What this means is that we will be lowering the amount of work the simulator does, based on frame rate and distance to object. Basically, we give the physics simulation less to do by taking things that are a distance away from you and making them simulate less.

“But wait Ryan!”, you say to yourself, which is a funny thing to say to yourself, “Won’t lowering the simulation quality result in a crappy simulation!? I love PlanetSide 2’s awesome physics and would never want to see any harm done to it! PlanetSide 2’s physics are the best ever!” Well, now you’re getting a bit too...Okay...I’ll accept that compliment. To answer your question, though, we simply don’t know.

Some of the things we’re doing are experimental. We have great ideas from some of the best in the biz on how to solve our performance problems, however coding is 60% science and 60% art. Some of the tasks I’m talking to you about today are still in their early stages, and may not make it into the final game as described, or at all!

With Adaptive Level of Detail, we hope to emulate the frames we don’t simulate, “smoothing” out any simulation errors that may occur, but it is going to take a lot of work and caffeine to make it, then even more to determine if it is good enough. As far as effect on frame rate, it won’t have much of one unless you have 4 or less cores, but it will have a HUGE impact on physics “hitches” and anyone with 2-4 cores will see potentially massive increases in frame rate when in a big battle.

## **Overhead Manager**

-----

Right now, when you’re in that Big Battle, we spend a lot of time drawing those little triangles (we call them Doritos, but get no product endorsement money...Can you believe that?) above people’s heads. They don’t look like much, but there is a complicated system under the hood to make sure we show them even though their owner may not be visible to you, and don’t show them when we shouldn’t and heck tons of other rules. We’ve seen real improvements from re-factoring that entire system to be more efficient and change only what’s necessary, when necessary. We’re also re-factoring its rendering code so it will be much faster as well.

## **Player Updates**

-----

The more players you have around you, the more time your game client spends updating the information about those players. Be it sounds they’re playing, bullets they’re firing, stats that are changing, etc. Your client spends an amount of time per dynamic entity making sure the things that are dynamic get dynamic-ed. In an effort to make this more efficient, we are only going to update a number of them per frame and we are going to be smarter about whether or not they need a full update. In early testing, this has resulted in less than 10% of all dynamic entities actually NEEDING to be updated though right now 100% of them get their update. Needless to say, in large battles this will be nice to have.

## **Animation Updates**

-----

PlanetSide 2 has a really robust animation system, with every tech animation buzz word you can think of (except for ragdoll... Yeah... Why isn't ragdoll done yet? Seriously!?) We animate a lot more stuff than most games out there, through some very complex state networks. For our animation optimizations, we're looking at ways to process fewer animations per frame and when they do process, have less complexity per process to iterate through while producing the same high quality result. This is tough work done through collaboration between the art and code teams, but we're confident we will get a good result here.

## **User Interface**

-----

Our user interface is incredibly complex. It may look simple (on purpose.... It's actually a heck of a lot MORE work to make something look simple!) but there are many tens of thousands of lines of code running all the time to keep that data fresh. However, now that we've had over a year to manage its functionality, we see areas where we can reduce the per-frame overhead and have it to give you the same data, but cost less to do so. We are actively optimizing the minimap and hud to reduce per-frame time to as little as 10% of what it currently is.

Furthermore, we are upgrading our UI middleware to its latest version, which includes optimizations that further decrease the time UI takes to draw.

Also, we are decoupling the part of the UI that manages the data, manages the on-screen movie, and renders the movie into three sections- each of which can run in a different thread. This decoupling also allows us to run each piece at a different rate, so they don't all have to update every frame.

## **Occlusion/Visibility**

-----

PlanetSide 2 has a huge world. 64 km<sup>2</sup> filled with over a hundred thousand objects and thousands of players per continent! Each combination of mesh (the polygons that make up an object) and texture (the things we paint onto the mesh) has to be individually described in detail to your video card and then rendered. So how do we render hundreds of thousands of things per frame? By NOT doing it! Easy, eh?! Instead we first split the continent up in a whole bunch of ways and then use 10 man-years' worth of code to determine what subset of those things you can actually see. This is Visibility. We also want to avoid rendering something that is behind something else, be it a "dynamic" object (like a person or bullet, something that changes) or "static" object (something that doesn't change, like a building or Higby's opinion). This is Occlusion.

We use a third-party code source called Umbra to do most of our Visibility/Occlusion. As part of OMFG, we are upgrading our Umbra to the latest version, which also requires a significant re-factor of how our visibility pathway works, changing several factors of our core client and tools. This is going to reduce the time spent per-frame on visibility by a sizeable amount, which means that the more complex your scene (like, for example, a base where there are thousands of objects) the more benefit you will get. We are also putting our visibility fully in another thread, so those of you with multicore systems (most of you) will see added benefit, and AMD users will see a pretty significant benefit from the parallelization of visibility

operations.

## **Multithreaded Renderer**

-----

Many elements of PlanetSide 2 are already in separate threads, but each of those threads often have to be in lock-step with the main thread and the renderer. So we are taking the part of the renderer that actually talks to DirectX and we're putting it into its own thread just like the UI renderer. This will give the main thread less to do per-frame, further increasing the frame rate for those with 4 or more cores.

## **More Instrumentation**

-----

We are working on adding even more tuning dials and nobs for ourselves, giving us even more power to see what's happening in these big battles and where we should tune next.

I'm also excited about our ongoing efforts to create a Benchmark Level. Old-school "run through a rail and compute your frame rate" here. SOE believes in empowering its users like nobody else, and this is another example. We're going to be giving you the ability to actually measure your frame rate in a static environment so you can, with every patch, have a one-button "How am I doing?" solver. You fly around a scene, while we compute all sorts of metrics on fill rates, object update times, UI times, and hundreds more... Then, at the end, you can see a score, discuss it and WE get that score as well, along with your hardware setup. Being able to have a static scene custom-purposed to find performance problems is HUGE. No more, "Well I got X frame rate at a bar in Indar on a Tuesday while it was raining." Now you'll be able to say, "Last week I got a score of 8300, but this week IT'S OVER 9000!!!!"

## **IN CLOSING**

We're going to keep talking about what we're doing and why we're doing it. These optimizations DO improve everyone's life, not just the low-end machines. They are significant increases in performance that will change your gameplay for the better.

You are a part of this. Questions welcome.

Ryan Elam - Tech Director PS2