EPISODE 891

[INTRODUCTION]

[0:00:00.3] JM: The PlayStation is a line of game consoles created by Sony, PlayStation devices include the PS2, PS3, PS4 and the PSP mobile system. Tony Godar worked as an engineer in the PlayStation ecosystem for 15 years, and he joins the show to give a retrospective on his time in the console industry.

Developing hardware and software for game consoles differs significantly from the world of web development. Tony describes the culture of the game development world and the challenges involved in the domains of software tooling, custom operating systems and streaming media. In 2010, the PS3 was hacked by notorious tinkerer George Hotz, a previous guest on the show and the founder of comma.ai. This event was discussed by Tony in today's episode. We also discussed the modern world of gaming and VR technology. Tony currently works as an engineer at MelodyVR, a company that makes virtual reality live music experiences.

[SPONSOR MESSAGE]

[0:01:17.5] JM: My favorite way to hire developers is contract-to-hire. In the contract-to-hire model, you pay a contractor to work with you on your company. If you enjoy working with them, you hire them full-time. Compare this to the traditional hiring model of bringing an engineering candidate in for several rounds of interviews. Instead of spending lots of time crafting questions to be done in front of a white board, contract-to-hire lets you actually work with the engineer on a real project and it compensates that engineer for their time.

It's always surprised me that contract-to-hire is not more widely used. Part of the reason for that is that there hasn't been a great platform that encourages contract-to-hire. Today, that has changed.

Moonlight is a platform for hiring high-quality software developers from all over the world to work on your project or your company. You can hire part-time or full-time developers. If you enjoy working with them, you are free to bring them onto your team. There are no lock-in effects.

Moonlight does not charge you a finder's fee. They will work with you to make sure you and your developers that you hire are all happy.
Go to moonlightwork.com/sedaily and get 50% off your company's first month of hiring access. That's moonlightwork.com/sedaily. If you're a developer who's looking for remote work and community, check out moonlightwork.com to join for free. I'm a customer of moonlight and I'm also a small investor. I love the platform and I'm actually amazed that something like it did not come out sooner. Check out moonlightwork.com/sedaily and get 50% off your first month of hiring access today. That's moonlightwork.com/sedaily. Again, full disclosure, I'm a small investor in the company, but I'm also a customer and I love what they're doing.

[INTERVIEW]

[0:03:25.6] JM: Tony Godar, welcome back to Software Engineering Daily.

[0:03:28.1] TG: Thanks for having me back.

[0:03:29.5] JM: You worked as an engineer in the PlayStation ecosystem for 15 years. How does console gaming engineering differ from modern web application engineering?

[0:03:42.9] TG: It's extremely different. I think, the main difference is we're always aiming towards a really clear framerate and we're always going for very advanced graphics effects. I think one of your recent podcasts about Google Earth actually hit it on the button and talked about it quite similarly, where Google Earth also has the same requirements of having to get a really fast framerate and never be interrupted. Whereas, when you're developing on the web, you are always – you never have that control and it's a very different programming paradigm.

[0:04:21.7] JM: You eventually joined Sony to work on the PlayStation 3. Before that, you worked as – well, you also worked on the PSP when you were at Sony. Before that, you worked at as a independent PlayStation 2 game developer. What games did you work on as an independent game developer?

[0:04:41.8] TG: Yes. I was initially got introduced to development in Japan where we were right at the time, the PlayStation 2 was at its peak. It was like the iPhone when had its first boom where everyone wanted to make an app. In Japan, especially everyone wanted to make PS2 a game. Our game was called Cam Station. It was essentially a video chatting online game, where you play some, I guess, initially some gambling and card and that games against each other.

One interesting way we kicked it off, because we didn't have PS2 development experience, was we started prototyping it with it with PS2 Linux, and which didn't require any special license or any special hardware. It can be off the shelf. We were able to get it to a certain state where we could show it off to Sony and then get them to agree to give us a development license.

[0:05:37.0] JM: What were the constraints of making a game on a PS2 aside from those core constraints that you talked about a little bit earlier, the framerate?

[0:05:45.3] TG: It's I guess trickier to develop on, I guess the SDK at the time. On the PS2, you had to have a Linux box set up with a special flavor of Red Hat Linux. From that, you'd be developing onto a dev kit and then that dev kit would run it. I guess at the time, it was actually quite easy to develop on. Since then, it's evolved so much where that looks quite difficult nowadays, right? Nowadays, everyone expects to be just on your normal desktop being able to emulate it and then run it on your device. That definitely wasn't the case back then. The interesting thing about the PS2 Linux in the time is you were developing and running on the same device.

[0:06:28.1] JM: What were the particularly hard engineering problems that you worked on in that PS2 independent game developer environment?

[0:06:36.7] TG: I guess, one of the main things was – so we were doing an online game with video chat and we didn't have the vast resources of github out there. We didn't have the vast resource of everything out there. A lot of it had to be invented from scratch. I guess, that was the biggest challenge. Other big challenge was actually getting the graphics to be a decent framerate. It was a small team, so learning all that.

We actually ended up using some technology developed here in London by the R&D team at the time. That's how I actually initially started meeting the guys here in London and eventually I ended up working for them.

[0:07:16.5] JM: Can you tell me more about the tool chain and the languages and frameworks that you used? I mean, you're saying you had to build a lot of the tools in-house that there wasn't much stuff to take off the shelf, but what kinds of tool chains did you have to build?

[0:07:32.8] TG: Well, everything's essentially C. You're also able to use C++, but everything that they supply to you is in C and you just adapted up to C++. They're really up until the recent generations wasn't anything any higher level, just because of performance-wise. At the time, C, C++ was very still considered extremely user-friendly and extremely easy to develop compared to the past and before that.

The key is being able to compile your code on your host PC, being able to upload it to the game console and be able to run it, debug it in real-time. Once you had that workflow, development becomes quite a simple process, which is I guess time consuming to some extent. Technology that's out there, a lot of it isn't really reusable. There is a core set of technology that the game console providers give you and there's also a wide variety of middlewares. That's the side business that that spawned from game consoles being closed is these middlewares provide the tools and libraries that aren't traditionally available to the niches.

[0:08:49.2] JM: You eventually joined Sony. When you joined Sony, you were working on the PSP, which is the mobile version of the PlayStation. How did the constraints of mobile gaming compared to console development?

[0:09:01.7] TG: PSP was actually by far, my favorite, favorite console ever. It was simplified in a way that there wasn't any complicated extra, I guess floating-point CPUs to DMA. It wasn't anything very complex and it was quite simple to develop. Or it was using a similar architecture to the PS2. It was MIPS as well.

The GPU again was simple, but powerful. The tools and the debugging worked really well, where you just plug into your PC and you can quickly get stuff running on it and quickly be able to debug it. It was just an overall nice little console. I think that's one of the reasons why it'll really appeal to the homebrew community once it was cracked unfortunately, but it was really was a massively popular in the homebrew community making emulators and then small homebrew games. Part of it was its architecture and its simplicity.

[0:10:00.2] JM: You said it got cracked. What's the story behind the PSP getting cracked?

[0:10:04.5] TG: I guess, from consumers' perspective, there were special firmwares where you could download on the PSP. From then, you could run your own code and run code that people made. There's a big community of emulators out there, so you could easily run Super Nintendo games and Nintendo games and that was quite fun and quite appealing everyone to have it in your pocket. Then it evolved and then people were starting to run commercial games on it and so forth, which wasn't exactly what Sony preferred you did with it. Overall, it really drove a lot of [inaudible 0:10:42.5].

[0:10:45.7] JM: Well, maybe let's come back to that if there's a good place in the conversation to continuing the story chronologically, how did working for a console company compared to working as an independent game developer? Because you were going for – to working for a large corporation on a console, from working as an independent. Well, I guess you were working on a small team for these independent games. That seems a dramatic shift both in terms of the company structure that you're working for and just the type of development you're doing.

[0:11:17.8] TG: Well, I was quite young and it was quite a dream come true in that. My first day at the job at PlayStation, I had a PSP dev kit on my desk. Yeah, and all of the libraries and everything that I could ever wanted to use. It was quite exciting in that aspect and it's always quite exciting to be one of the first ones to use and to experiment and try and figure out how to use these consoles.

[0:11:44.8] JM: What was your job when you were working on the PSP? What were you tasked with?

[0:11:49.6] TG: I guess, initially my expertise from my PS2 days was working on a video streaming, video chat and that evolved into video playback. On the PSP, I was working heavily with – PSP had a UMD disc drive and it's a proprietary little mini-disc type drive. There's also movies you could buy on it that were DVD quality. I was working a lot on that. Essentially, that evolved to just video playback in general, and so on that, turned into PS – on PS3.

Before we had streaming Netflix or anything like that, we developed a terrestrial TV tuner with a really good EPG and really quite gaming interface and with even some social interface, where you can share with your friends and chat with your friends as you're watching TV and that sort of thing. There's always around video streaming, video and interacting with video.

[0:12:51.1] JM: We've done some shows about Netflix and other streaming video platforms. There are some prototypical challenges of handling streaming media. This is a problem that never dies. What are the prototypical challenges of handling streaming media over an internet connection?

[0:13:09.7] TG: Well, getting to an internet connection or, actually jumping a few generations down into PS3 already.

[0:13:17.1] JM: Oh, that's true. That's true. Right. Yeah. I guess we should start with pluggable media.

[0:13:23.0] TG: The PS2 and the PSP didn't really have internet capabilities. They were essentially offline. If you get an attachment to the PS2, they gave it an Ethernet, but essentially it was offline. The PSP had Wi-Fi support and it had some game sharing capabilities, but it was essentially an offline device. Later on, it’s like the store started and then the PS3 as well started off very the offline and then eventually became online. It wasn't a while later when streaming video services came into the picture.

[0:14:00.0] JM: Okay. Can you talk about the canonical engineering problems that have emerged in the streaming media space?

[0:14:08.6] TG: Okay. On the PS3, so essentially we started off with terrestrial TV, which was had bandwidth limits and were able to record well, pretty much perfect quality video on to hard disk. When streaming video started appearing, there wasn't adaptive streaming, so a lot of it was your early days of Netflix, where – not Netflix. Like in YouTube, where you could see the buffering bar go ahead and you'd always be trying chasing that buffering bar.

Then that progressive download evolved. I think, essentially because we're located in Europe, Smooth Streaming Microsoft technology was one that took over the majority of our partners here in Europe. In America, HLS, HTTP, Live Streaming became more prominent. At that point, in the life of the PS3, a lot of the technologies were similar to what you see on the PC. A lot of the video streams you'd play on other devices would also be playing on the PS3 and we'd also be playing on their TV and so forth.

I guess the biggest challenge per se was actually a harder challenge for a lot of the streaming services to get the video onto their TVs than it would be onto the game console. The game console was a far more faster, evolving platform. Whereas, TVs were six months, one year ahead. We're jumping around a bit here though.

[0:15:39.2] JM: We are. We are jumping around. Okay, coming back to the chronological thread, with the PSP, can you just tell me more about the operating system that the environment of the PSP. What was it?

[0:15:50.3] TG: It was very much so similar to PS2. We had a proprietary operating system, nothing really that fancy enough to boot up –

[0:15:57.3] JM: It was a version of Red Hat?

[0:15:58.7] TG: Oh, no, no, no, no. The version of Red Hat we saw in PS2 Linux was essentially sitting on top of the operating system. It was really in its own virtual machine per se.

[0:16:08.4] JM: Oh, okay.

[0:16:09.5] TG: On PSP, it was just a really bare-bones operating system, enough to get you booted up and into your game.

[0:16:15.4] JM: An OS that was written by Sony?

[0:16:17.4] TG: Yes, yes, yes.

[0:16:18.5] JM: Wow.

[0:16:19.5] TG: Then that's pretty much allowed the same hardware to perform to pretty much of its max. That's always been the goal with all the game console operating systems is to eliminate any in between processes, to eliminate anything any complexities.

[0:16:36.0] JM: I don't know how much you know about this, but when a company like Sony writes their own operating system for a device, like a console, or the PSP, are they forking Linux? Are they forking something, or are they just writing everything from scratch?

[0:16:51.6] TG: Usually, they're writing a lot from scratch, but they're also writing a lot from their previous consoles, bringing it over. They still have a lot of good code that they trust and they're familiar with. That's when the crutches and that's one of the issues with the exploits that have happened over the years. You'll see exploits on the newer mobile, the PS Vita that are exploits that were inherited from the PSP. You'll see exploits on the PS2 that were inherited from the PS1. You have that inherited codebase, inherited architecture always.

On the flip side, by having the same architecture, the same behavior, someone who's making a game on the borderlines, they're making it on the PS2 and then they need to make on the PS3, because the generation changed. It's not a major port. They don't have to rewrite a lot of the thought from specific code from scratch and change the whole behavior around.

[0:17:51.8] JM: What was the interaction between the hardware and the software teams within Sony?

[0:17:58.4] TG: It really depends on the console generation. Overall, you'll have the core SDK team, which will work fairly close with the hardware team. Then on top of that, you have the general SDK team, which will be based in Europe and America and partially in Japan. Those ones will sit a couple layers away from the core hardware team. The core hardware team will make samples and libraries that'll get exposed enough to see all the hardware, but then the next layer of the SDK is to make it easy for game developers to use.

[SPONSOR MESSAGE]

[0:18:48.3] JM: Today's show is sponsored by Datadog, a monitoring and analytics platform that integrates with more than 250 technologies, including AWS, Kubernetes and Lambda. Datadog unites metrics, traces and logs in one platform, so that you can get full visibility into your infrastructure in your applications.

Check out new features, like trace, search and analytics for rapid insights into high cardinality data, and Watchdog, an auto-detection engine that alerts you to performance anomalies across your applications. Datadog makes it easy for teams to monitor every layer of their stack in one place.

Don't take our word for it. You can start a free trial today and Datadog will send you a t-shirt for free at software engineeringdaily.com/datadog. To get that t-shirt and your free Datadog trial, go to software engineeringdaily.com/datadog.

[INTERVIEW CONTINUED]

[0:19:55.2] JM: When you worked on the PSP, eventually you evolved into a position where you were focused on this streaming media, including streaming video and streaming music. This was around 2005 through 2009. Can you just tell me a bit more about what you were doing then?

I think most of people listening to this probably either do not remember, or were just simply not even plugged in to the PSP ecosystem around then. They have no idea why there was a streaming – so much streaming media and streaming video and stuff on the PSP. Just give some more insights into what was going on around 2005-2009 and what you were building.

[0:20:41.8] TG: Early on in the PS3, one of the key features it was able to stream to your PSP and then essentially, you could use it as a remote screen. That was essentially using standard streaming media technologies. Streaming video was essentially an evolution from the terrestrial video side of things.

We hit the end of the project of the DVB-T player and then it was from there, do we evolve into more terrestrial type video solutions, or go up into cloud streaming. At the time, BBC iPlayer was massively getting big. They worked really close with us on technologies to bring the BBC iPlayer onto PS3.

At the time, the PS3 was – this was before they even had the iPhone version. It was beyond, I guess PC, it was their second most popular way of watching BBC iPlayer was on the PS3 at the time. From there, so BBC sets the standards and a lot of the broadcasters will follow those standards. It's from there, supporting other broadcasters who have similar services with the same capabilities. That evolved into streaming protocols, like smooth streaming, HLS and eventually MPEG Dash.

[0:22:08.3] JM: What kinds of problems did those new streaming protocols solve? Why did we need new media streaming protocols?

[0:22:13.8] TG: I guess, the main issue across Europe was I guess quite extreme differences in internet capabilities from country-to-country and we're supporting a lot of countries. Adaptive streaming was quite necessary.

[0:22:27.2] JM: Adaptive streaming. Is that the bitrate ladder thing?

[0:22:30.4] TG: Exactly, exactly.

[0:22:33.1] JM: Explain what that is.

[0:22:34.5] TG: Essentially, if you have low-internet bandwidth capabilities, a lower quality plays, but you don't buffer. If you're just higher, it'll step up to the higher bitrates. The key is no buffering.

[0:22:49.6] JM: Lower quality, meaning lower fidelity, the just grainier picture, less refined picture.

[0:22:57.6] TG: A lot of work has been put into this. What you'll notice is on the lower bitrates, lower quality. They aren't necessarily that grainy, but they're more fuzzy a bit, but there's lower the resolution and you lower the bitrate. As you raise your bitrate, you also raise your resolution up until your peak. Now when you get quite a nice step and when it changes resolution, there's essentially just snaps into place and people are used to that by now. Back then, it was just quite revolutionary. The first few generations were definitely just progressive video and some of them even had choosing your quality; low, medium and high quality.

[0:23:38.8] JM: I'm starting to realize, it seems most of your time working on PlayStation was actually working on the video streaming and the music streaming side of PlayStation engineering. Were you working on game development, or game engineering at all?

[0:23:56.6] TG: To some extent. The game engineering side of it was to put a user interface on to these, to take code that's general-purpose and bring it onto this game console. That's where knowing the game console, using the game consoles capabilities came into play, because it's not a traditional architecture, where its CPU-heavy and the GPU separate. It's always quite the untraditional architecture, but it's always usable in that aspect. Yeah, it was a good mix of the two.

[0:24:29.2] JM: How had the platform evolved from the PS2 era to the PS3? What had been updated? What were the major technological breakthroughs that were happening on the PS3 that moved game development forward?

[0:24:45.8] TG: PS3 caught up in a lot of spaces, so way more memory to work with. Way faster memory. You had a big chunk of GPU memory, you had a modern NVIDIA GPU in there.

Essentially, you can run shaders, you can run a lot of the graphical effects that you expect from a game.

It was essentially just at that point, bringing it equal, or slightly above what the PCs were doing at the time, and then throwing in some heavy floating-point capabilities, so it had its own custom chips in there and that allowed it to exceed what a PC could do in its time. Those were quite useful and that they could also be repurposed for decoding video. When I was using those, like a support the following point CPUs, they were essentially used to decode the video and they were decoding essentially for free, because they're off the CPU.

[0:25:46.9] JM: How did the PlayStation and Xbox ecosystems differ from one another? Because Xbox was coming up around the same time as the PS3, I believe.

[0:25:57.8] TG: Yes. The Xbox 360 came out a year before the PS3.

[0:26:01.5] JM: Oh, so the first Xbox had been around for a while.

[0:26:04.4] TG: Yeah, the first Xbox came out after the PS2 and that one was, I guess an a good start. Xbox 360 was a very good piece of hardware, which I couldn't openly say at the time, but it was quite impressive, where it had a unified memory, it had fairly good ATI or AMD GPU in there and it had –

[0:26:26.3] JM: Wait, unified memory. Can you explain what that means?

[0:26:29.1] TG: On your traditional PC, you have normal CPU memory and you have memory on your GPU and you have to communicate between the two. On the PS3, we had the same thing. If you're creating some data you want to put up the GPU to create in the CPU and then batch it up so it sends it to the GPU.

On the Xbox 360, as far as I know, I've never actually developed for it from the specs and from what people had told. This all the same memory, you just remap the memory to behave on CPU or behave on GPU. You'll see that on modern devices modern, modern consoles, like the PS4 and Xbox One. Yeah.

[0:27:04.6] JM: Why is that useful?

[0:27:06.3] TG: Essentially, it saves a lot of memory. What we saw a lot in the PS3, because it had 256 Megs of CPU memory and 256 Megs of GPU memory. A big problem was when you had to process it with your CPU, you have to bring it down onto the main memory and then copy it back up onto the GPU memory when you want to draw it. Ideally, you want to just remap it so you can access it from the CPU, do your changes in-line, or whatever you want to do it and then just remap it back up the GPU and use it as is. The whole unified memory is a big win in that aspect.

[0:27:43.3] JM: If I understand correctly in the divided memory model, the CPU is talking to the game environment and the CPU is redrawing the world, then the CPU is going to render some representation and then it's going to need to copy that representation over to the GPU. Whereas in the unified memory environment, it just gets written once to the GPU and then it gets defined as okay, now render this.

[0:28:16.9] TG: Essentially, yeah. Most of the time you don't have to copy stuff over. Most the time, you can bring it up and always live on a GPU memory. There's always in certain cases, where it's quite useful to have that, the unified architecture. Even just for simplicity of development, that's a big win in that aspect.

Yes. Xbox 360 came out, it had a lot of fans in the development community. PS3 was harder to develop for with its unique architecture. That was a challenge at its time. PS3 was also a lot more expensive, almost twice the cost an Xbox 360. A lot of games looked better on Xbox 360, so it was quite a challenging time.

[0:28:57.9] JM: What's your perspective on Nintendo? How did that company play off against the PlayStation and Xbox ecosystems?

[0:29:06.4] TG: They always did their own thing. They stuck to different cycles, whereas Xbox always latched onto the PlayStation 2. Nintendo did its own thing and didn't really get in our way, to be honest. There was always good relationship between the Sony developers, Nintendo developers. Often, they weren't working on the same games and so forth. There's respect on both sides. Then everyone loves that everyone grew up with a Nintendo games, so no one did –

[0:29:29.3] JM: That's true. A company still has such an affinity among everyone, basically. That's such a strong brand.

[0:29:38.0] TG: Big time, big time. Yeah. I loved my Mario and everyone I think does.

[0:29:42.3] JM: The PlayStation and Xbox ecosystems, was that a zero sum war between these two ecosystems, or did a lot of people buy both of them? What were the competitive dynamics of those two ecosystems?

[0:29:57.5] TG: In the PS3 days, I think a lot of people were on different sides. Then I think it evolved later on in the console lifecycle where people had both, because they just want the games on that other console. It wasn't a horrible thing in that. They still ended up selling the console to people. Ended up selling games to people. Overall, there's a big boost in that generation for the 360.

[0:30:25.8] JM: What about the desk top gaming ecosystem? How does the desktop gaming ecosystem differ from the console ecosystem?

[0:30:34.6] TG: I guess, desktop system is quite different in that. It's a different type of person. It's not one who's casually using it on their TV. They definitely have to have far more expensive computer. You're paying more upfront then yes. I guess, the standard image is someone has to buy a GPU that's more expensive than any game console out there, just to be able to play games, or equal to the game consoles. I'm not a big fan of PC gaming, but I guess it just depends on the type of game.

[0:31:07.4] JM: Why now? Why do you prefer the console ecosystem?

[0:31:10.6] TG: You can pick it up and play it. You can play it on your TV, you can – there's not just games. There's especially the stuff I like is being able to watch Netflix on it, being able to watch BBC iPlayer, being able to do a lot of things outside of gaming. It just works, so anyone in the family can do it.

[0:31:28.4] JM: Why do we still need these consoles? Why not have all the gaming take place on my phone, or my MacBook, or some Chrome-casted experience? Why do we still need a dedicated hardware device for gaming?

[0:31:43.5] TG: It's a good question. Do we? I think as time goes on, maybe we won't need it. Maybe we won't. I think right now, there's still quite a demand for these really immersive games. Some of these games even yeah, include VR headsets and really, they're quite immersive and quite a step ahead of what you have on your phone, or on your tablet by far.

[0:32:09.4] JM: If we jump forward to VR, you worked some on VR at PlayStation. Let's talk a little bit about that, then maybe we can talk a little bit about the Oculus or other VR environments. You worked on in PlayStation R&D from 2012 to 2017. Can you tell me more about what kinds of R&D projects you were working on?

[0:32:32.9] TG: The majority were – again, based around video. Anything VR related was around playing back video, playing back 3D videos, playing back 360 videos. I was quite involved in 3D TVs at the time, because a lot of the content was 3D video content. Some of the proof of concept work out there was head tracking and so forth, which evolved into a lot of what became the PS VR headset.

PS VR evolved in a similar time to Oculus and they took a very different way. Instead of brute force, they went more towards a simpler way of rendering to ensure that it's very smooth head-tracking. It's not the highest resolution, but it's a very smooth head-tracking, smooth experience and quite appealing to the average gamer and that they don't get sick and that it's quite comfortable.

[0:33:35.6] JM: Can you tell me about the runtime loop there? Because it sounds like, so if I'm wearing a VR headset, it's doing head-tracking, it's rendering a 3D video onto my screen, latency needs to be really low. What's the runtime loop look like?

[0:33:52.7] TG: That's a nice thing about having it on a custom operating system, in that access to video frames before they're in the frame buffer, before they're required, is possible so that tricks can be made to reduce the latency of the video that's on the headset. Essentially, due to that, the capabilities, the I guess, initial PS VR could pull off 120 Hertz, which really gave that smooth head-tracking, even though the game itself only had to render 60 Hertz at the 1080p, which is quite achievable on pretty much even the standard PS4 game console. We're jumping around a bit more, but sorry.

[0:34:36.7] JM: We are.

[0:34:37.3] TG: It's hard for people to follow us a bit, because we're –

[0:34:39.2] JM: No, that's okay. Let's double down on VR for a little bit. Tell me more about the lower level engineering aspects of building a VR framework, or building a VR environment. What are the hard engineering problems that you have to solve?

[0:34:54.7] TG: The hardest – oh, there's many, many. Yes, one of the biggest one as I touched on is your head-tracking. What you see on your screen needs to move when you move your head and it needs to move fast enough that your eyes don't get confused. When I'm working with video, it doesn't necessarily have to be attached to that head-tracking. Your video can be running at 24 Hertz, it could be film style video. If your move head is moving in I guess a refresh rate of 120 Hertz, then it doesn't feel odd at all. You can be watching a film, which feels it's a film stock 24 Hertz video, but with by moving your head, you still get the feeling that you're in that immersive experience. That's the biggest challenge.

To handle the difference on all VR headsets, so on the PlayStation they used a camera and lights and they track that camera – that camera tracks the lights at the 120 Hertz, and that's one of their tricks. Then on other systems like the Oculus, the HTC VIVE, they use infrared lights and they use tracking through either cameras, or base stations which are similar to cameras and track your head that way.

Then on the flip side, you also have a lot of gyro sensors. Essentially, your phone is stuck inside the headset to give you the XYZ and any other portions of gyro gravity and so forth. That's the biggest challenge. That's why in the 90s when you had these VR headsets, they felt horrible. That's why even leading up to all of these modern VR headsets, you really can't play them for long at all, because they just didn't match reality.

[0:36:44.7] JM: Did you get sick a lot working on this stuff?

[0:36:47.2] TG: Sometimes, yeah. Sometimes stuff really felt – I felt sick. I'm probably getting carsick, so I never want to sit in the backseat. I'm quite happy to say that yeah, pretty much on the last few generations, it gets better and better and better. I never get sick under anything on the PS VR, because of the 120 Hertz. The latest headsets, Oculus Go, the Oculus Quest, no chance to get sick on those, those are perfect tracking. They're really good.

[0:37:14.7] JM: What are the key advancements that have been made that have allowed VR to – I mean, today VR it's not quite – it's not super popular levels, but it's at least at the levels where it doesn't make people sick anymore, people are excited to play it, people are a little freaked out in how amazing it is. What are the key technological advances that have allowed it to get there? Is it hardware?

[0:37:37.3] TG: Yeah, a lot of it is hardware. Essentially, so the first Oculus rift was taking pieces from mobile phones. Wasn't until then that it was even feasible. The screens being able to be extremely high density, and I guess, LCD screens and all that screens with fast refresh rates, pixels that can be stacked on top of each other so you don't have screen door effects, a lot of these are still cutting-edge nowadays and you don’t even see on many phones.

[0:38:07.7] JM: Pixels stacked on top of each other. What do you mean?

[0:38:10.7] TG: On some of the headsets, including the PS VR, instead of having the pixels like you have on a standard LCD screen right beside each other, because it's so close to your eyes, they stack them on top of each other. When you're combining colors, or when you have a singular color, it is in one big thought. Then there are other thoughts around that you don't have gaps around it of the non-big thoughts. That makes a big difference.

[0:38:36.7] JM: That's the RGB – you mean the RGB pixilation?

[0:38:41.4] TG: Yeah, yeah. That's one of the tricks that allows the resolution to be quite low on the PS VR, yet still being quite I guess, it feels quite good. It's a lot of stacked pixels, a lot of resolution, just stuck in the area right in front of your eyes.

[SPONSOR MESSAGE]

[0:39:07.3] JM: GitLab is the open source platform for DevOps. The successive GitLab has accelerated the adoption of DevOps across the enterprise market. GitLab Commit is the official conference for GitLab and it's coming to Brooklyn, New York, September 17th, 2019. GitLab Commit features discussions of technology, lessons learned and a close look at the DevOps lifecycle, including Kubernetes, CICD, DataOps, security and remote culture.

GitLab Commit is September 17th in Brooklyn. You can save $99 on your pass by going to softwareengineeringdaily.com/commit and entering the code COMMIT99 to save $99 on a conference pass, if you register by August 15th at 11:59 p.m. Pacific time.

GitLab is an exciting ecosystem of products and it's accelerating the world of DevOps. If you can make it to Brooklyn on September 17th, check out GitLab Commit and go to softwareengineeringdaily.com/commit to get that additional $99 off by entering the code COMMIT99.

Thank you to GitLab for being a sponsor of Software Engineering Daily.

[INTERVIEW CONTINUED]

[0:40:31.0] JM: There's this meme that the gaming industry feeds into the rest of the industry. The gaming industry often drives advances in machine learning and graphics and so on. What you're saying there is interesting, because you're saying that the mobile phone ecosystem, actually in this case, drove the ability for the VR headset industry to catch up to where it needed to be.

[0:40:59.7] TG: Definitely. Yeah. I guess, there's always borrowing of technologies, definitely. Gaming industry has pushed streaming video technologies in one direction. The Netflix has pushed it in the other direction, mobile phones have pushed resolution. I guess, sensors in one direction. VR itself is pushing another direction, so you're seeing they are capable on phones, because those phones have been made VR compatible. Before that, those phones really didn't have the sensors capable enough to have a really smooth AR experience. It just bounces back and forth as the niches become more popular.

[0:41:41.3] JM: The software development process over the last 5-10 years has really gotten a lot faster, largely due to things like the cloud and "DevOps" and infrastructure is code, better release processes. Some of these elements have come more slowly to mobile development, partially because the mobile ecosystem is fragmented compared to the web ecosystem. I wonder, is it the same for console games? Is the release process and the development process for console games is still – is it still a little bit painful relative to web development?

[0:42:21.2] TG: It's quite isolated in its own world. One big evolution in game development process is – the man who architected the PS Vita and the PS4, Mark Cerny has a game development methodology, and then that's really taking over the game development industry by storm, in that you essentially perfect your first level, you get your minimal viable product out there and then you develop your game. Before that, there was a lot of games where they developed the full game, until they realize it wasn't viable.

That minimal viable product has become a lot faster and a lot easier on mobile. That's again, this is this ping pong effect where mobile techniques of continuous integration will be brought back into games and debugging capabilities from gaming development will be brought onto mobile as the framerate needs to be improved. There's a lot of learnings from both sides, ping ponging back and forth.

[0:43:29.1] JM: How is the cloud impacted the world of game development? Has the public cloud had much impact on it?

[0:43:36.5] TG: Streaming of games has slowly how – it's been building up momentum. It's been around for quite a while now, where are you able to just from your PC, or from a light client, like a Chrome sticker and so forth, be able to play the games. It's slowly becoming a reality. I guess, the earliest version that I was exposed to, that was being able to essentially play the PS3 game on your PSP over the internet, or PS Vita over the internet.

Then shortly after that, Sony acquired a company called Gaikai, which was essentially doing that streaming games from a data center into your living room. Then so Sony started a service called PlayStation Now. I think Xbox and a few others have and Google of course have their own equivalents. Those are all pushing forward. Whether they will catch on, I'm not sure, because they're very different. Yeah, the next generation, there could be a new opportunity for another niche, which is a very light client, thin client.

[0:44:47.9] JM: Let's jump back to that brief anecdote you alluded to about the PSP getting cracked in the sense that people could upload SNES games to it, or basically do whatever they wanted to with their PSP. What happened there and then what was the fallout?

[0:45:05.3] TG: I guess, what happened was the security really wasn't that strong on the PSP. The homebrew and indie developers really took over with a lot of a momentum with it with their emulators and their homebrew. It was actually a good healthy community. A lot inside of Sony really admired them and a lot of those indie developers became real developers. That was a good thing in that aspect.

I guess, the downside was compared to the Nintendo DS at the time, the PSP owners weren't buying as many games. That wasn't a good thing for the spreadsheets at the end of the day. A lot of people were buying the PSP and hacking it and playing emulators; only buying maybe one game, two games in their lifetime. As the consoles, so as the PS3 evolved, they want to avoid that and they want to avoid people using their game console as a general-purpose PC for hacking at homebrew. They want to turn it more of a closed system.

[0:46:07.3] JM: Yeah. It's hard for me to put myself back in that timeframe. I wonder what would have happened if they would have encouraged that ecosystem. I don't know. Well, I don't even know if the emulator stuff might had dubious legality as well, so maybe you can't even support that thing.

[0:46:25.9] TG: Yeah, a lot of stuff you just have to smile and let them do it, because I can't officially really grasp about that and say yes, emulate the Super Nintendo. Good work. Yeah. I think what they wanted to do, at the time they were seeing the explosion of the App Store type mythology. They wanted to evolve more towards that aspect. We have a closed door and that's a high-quality content that's really heavily vetted and so forth. That's what you have today and that seemed to work. The key was just not to piss off the developers and really get on their side. Unfortunately, PS3 days, they didn't always keep to that. After that, they learned the lessons.

[0:47:10.9] JM: Tell me more about that. How did the PlayStation Corporation alienate the developer ecosystem?

[0:47:19.3] TG: Like I said, developer ecosystem, but the indie developer. What I was doing before I ended up as a PS2 developer, where I bought the PS2 Linux and prototype things, a lot of game developers were buying – indie developers obvious and so forth are buying the PS3 at the time, or jumping back to the PS3 days. They were running other operating systems. It was a feature that PS3 when it first launched, where you could run Linux FreeBSD. Then there was a various flavors of Red Hat and you've been to a whole bunch of different distributions. It's quite a thriving system there.

The architecture was quite interesting and that you could really do some complicated maths with the coprocessors in there. It was almost like a early I guess, AI type, physics processing. There's a whole bunch of different uses for that. That was going well, until a hacker discovered and exploit and that's locked things down. Then causing PlayStation to eliminate that feature and I guess, upset a lot of these indie developers, a lot of these hackers and then sparked the interest of the research of the security researchers.

[0:48:34.2] JM: That hacker was George Hotz, right?

[0:48:36.3] TG: Yeah.

[0:48:37.5] JM: He was a previous guest on the show.

[0:48:39.6] TG: Really?

[0:48:40.5] JM: Yeah.

[0:48:40.8] TG: I think I got to find that one.

[0:48:42.2] JM: It’s a good one. He’s pretty funny. Have you seen any videos of him?

[0:48:46.6] TG: I have, I have, I have.

[0:48:48.2] JM: He's an entertaining guy. He hacked the PS3 in 2010. Can you retell that story and tell me the fallout of that PS3 hack in a little more detail.

[0:48:59.3] TG: Okay, well I can give you what's available in the Wikipedia, because I can't tell you what was going on the other side, but I can tell you –

[0:49:04.3] JM: Okay, sure. That’s fine.

[0:49:05.6] TG: - roughly what was going on in general. Yes, so in 2010 he announced the breach. He didn't really release anything.

[0:49:12.9] JM: He just said, “I have hacked PS3.”

[0:49:15.7] TG: He given enough proof that it proved to the guys inside that he was telling the truth. That was enough for them to lock down the other OS and upset the rest of the indie world, the homebrew world and then geohotz became a bit famous in that aspect. For doing that, he was quite young at the time. He made a song on YouTube about it. It was quite interesting.

Also at the time that the PSP homebrew world was quite thriving, so that group was also quite upset what are they doing. Then that snowballed into a year later, so 2011 released the route keys. Essentially, the keys to the safe and that really ruffled a lot of feathers and ended up getting him sued. That snowballed into pissing off even more people and so many getting hacked and oh, yes. The rest is tracked heavily on Wikipedia.

[0:50:16.7] JM: Okay. Let's zoom out a bit. You were in the gaming industry for 15 years. What are your high-level takeaways about business and developer ecosystems from your 15 years in gaming?

[0:50:30.9] TG: The nice thing I guess about the game console side of things is it's quite an isolated world from standard PC, or mobile development. When done right, it's a massive market. There's still big opportunities in that side. There's still big opportunities to be made. I guess, another big opportunity is because there's a massive market and they heavily vet the titles on the game consoles, and because it's quite complex to become a licensed developer and to actually go through the whole process, anything that gets to the end of the process gets a lot of attention, a lot more attention than you would get on a website or on a mobile app. That's a big advantage to that still.

The technology, because it's such a light operating system, because it's such a fast console, fast device, you can really exceed the performance of any other PC, or any mobile game, or app you'd ever make and have the premium version of it on this game console.

[0:51:37.9] JM: I remember the first time I downloaded an indie game on – I think I downloaded an indie game onto a – was it a Wii? It might have been a Wii. I'm not much of a gamer, but I just remember the first time I read about an indie game and I downloaded, I believe to some console. I just remember thinking, this is such a cool development when the indie game world started to develop and you could just download games that were made by these one or two people. It was just so cool. What have been the effects of the development of that indie game, that indie ecosystem?

[0:52:15.4] TG: From that, so I think especially after all the George Hotz problems, Sony and I think even all the other, especially Sony Microsoft, they really embraced the indie community. They really worked hard to define all these indie games and to really bring them into the professional game developer world. Giving them hardware, instead of selling them $10,000, $20,000 hardware, it's really just loaning them hardware, going there and working with them and really bringing them onto the platform. There was a huge push on indie games, especially.

Those are quite, I guess quite important to any game console and in that they're small, fun games that they're retro and they give you a lot of memories from your youth. They aren't just these big blockbuster games. That's of one the good side effects of these – the problems they had early on with the PS3.

[0:53:15.7] JM: Our last episode, we talked a bit about the VR industry. How has your time in the gaming industry prepared you for the virtual reality world?

[0:53:26.1] TG: My specialty being more on the video streaming side, with some aspects of game development, so the video streaming side it was quite useful in my current position where we do live streaming and we do have quite high-resolution, so beyond 4K VR content. I guess, the gaming side is more towards being able to debug and ensure that the framerate is really high and acceptable, which a standard mobile developer doesn't really work towards.

For a lot of tasks, we definitely need game developers to be brought in. A lot of tasks, if we need to go and make a plugin that has to communicate in C or C++, then definitely a game developer needs to be doing that. Even on modern games which end up in mobiles, we still have to make sure that it runs extremely smooth.

[0:54:23.0] JM: Give me a prediction about how the gaming industry will change in the next 10 years that I would not hear from anyone else.

[0:54:33.2] TG: That's a good question. I feel the way Nintendo has been spinning in innovation with re-controllers and then now with a switch with a screen, it's really moving in the right direction. I think a hybrid of something, which is mobile, yet a console, definitely still has its place 10 years from now. It'll definitely be something that's augmenting reality, will definitely be something that you can touch and interact with, definitely be – won't necessarily be something that you stick on your head, but it will definitely be something you're looking through, and maybe something that augments your vision to some extent. There's so many directions it can go, it's hard to really guess.

I don't necessarily think it'll be a standard game console in the cloud. I don't think it'll be that simple. I think it'll definitely be more complicated. It's going to be this ping pong effect of taking some technologies from mobile at the time and bringing it towards a game console and back and forth. The camera evolution on phones is quite amazing and it's quite impressive how every single six months, there's two or three cameras added to each phone. The use for that could essentially be in gaming. The use for shrinking that technology and putting it into glasses where they could be used for gaming in 10 years from now.

[0:56:02.4] JM: Tony Godar, thank you for coming back on Software Engineering Daily. It's been really fun.

[0:56:05.9] TG: Thanks for having me.

[END OF INTERVIEW]

[0:56:10.9] JM: Podsheets is an open source podcast hosting platform. We are building Podsheets with the learnings from Software Engineering Daily. Our goal is to be the best place to host and monetize your podcast. If you've been thinking about starting a podcast, check out podsheets.com. We believe the best solution to podcasting will be open source. We had a previous episode of Software Engineering Daily where we discussed the open source vision for Podsheets.

We're in the early days of podcasting and there's never been a better time to start a podcast. We will help you through the hurdles of starting a podcast on Podsheets. We're already working on tools to help you with the complex process of finding advertisers for your podcast and working with the ads in your podcast. These are problems that we have encountered in Software Engineering Daily. We know them intimately and we would love to help you get started with your podcast.

You can check out podsheets.com to get started as a podcaster today. Podcasting is as easy as blogging. If you've written a blog post, you can start a podcast. We'll help you through the process and you can reach us at any time by e-mailing help@podsheets.com. We also have multiple other ways of getting in touch on Podsheets.

Podsheets is an open source podcast hosting platform. I hope you start a podcast, because I'm still running out of content to listen to. Start a podcast on podsheets.com.

[END]