Cloudflare Workers for Gaming Product Strategy

By Celyna Su

Background & Thesis

Serious game development isn't limited to having programmers code an entire game engine or having enough money to license a big name. "Serverless" computing, or FaaS, has been around for a decade. It's been proven useful for software developers; FaaS manages servers and horizontal scaling for you. Money wise, you only pay for what you use. Now, it's time to revolutionize the online gaming industry with Cloudflare Workers for Gaming (CWG).

The online gaming market requires fast internet and quick fixes to their issues. For popular games and AAA titles, there are thousands of users around the world accessing these servers. **Here's how CWG can benefit the game development industry:**

1. ***Improved Performance & Accessibility*** - Cloudflare's edge network offers seamless horizontal scaling with 117+ locations around the world to a customer base of millions(which reduces latency, a common problem seen in multiplayer games).
2. ***Enhanced Security*** - Implement custom security procedures and filters while Cloudflare Workers (CW) ensures that code deployed to Cloudflare will not harm their network/others. Scripts are securely run using V8 Javascript engine.
3. ***Increased Reliability*** - Fix mistakes fast: implement load balancing and failover systems to achieve high availability.

Goal: Game development with CWG should be anything but difficult: it should be **scalable, safe, and simple** to all with the slightest interest in game development.

Potential Risks

1. Mainstream cloud providers have put out other similar platforms:
    - Microsoft Azure - Simplifies complex, stateful coordination requirements in serverless applications but lacks the performance and scalability
    - AWS Lambdas - Good for rapid scale-out and handling massive workloads compared to Azure, but HTTP endpoint integration is an additional fee; both Azure and Lambdas lack extensibility
    - Google Stadia - Visual quality is good, but there are issues with latency
2. Delivering effective security for serverless functions is challenging.
    - Risks in serverless functions can stem from application code, cloud environment, or other elements

Customer Segmentation/User Research

The possible consumers for CWG are separated into 4 customer groups:

1. First time developers/hobbyists (i.e. UC Riverside's game development course final project)
2. Application/smaller scale companies (i.e. games developers on Steam)
3. Larger scale/AAA title companies (i.e. Activision Blizzard)

Target: Customer segments **1** and **2** are our ideal targets for CWG. To put it straightforward, they may not have the investment(time, money, workload-wise) and would benefit from FaaS. CWG would allow them to focus solely on individual functions in their application code without the complex infrastructure typically involved with game development.

Approach: Each group has their own pain points. User interviews or polls are to be conducted to determine those, and questions per group will vary. However, **scalability, safety, and simplicity** are the main goals and those should be emphasized during those interviews/polls. Gamers are not one of the primary persona groups, but their input is valued as they directly interact with the released games.

Development & Prototyping

Consumer insight is extremely important to this product. Taking into consideration the user interviews and polls, developers can start organizing sprints to address groups **1** and **2**'s pain points. CWG needs to stand out amongst its FaaS competitors, so possible concepts to implement are:

- Build an API that enables constant monitoring to detect problems quickly (i.e. Anomaly Detector by Microsoft Azure)
- Build real-time multiplayer APIs that enables network communication between multiple players, player interface, stores participation/inventory (i.e. Google Play)
- Build a messaging and notification service for player use. (i.e. IoT interaction: SignalR)

Open Beta Testing

Launching an open beta testing with the newly built prototype is crucial in ensuring a successful product. Limit the number of developers that can sign up for the beta version(similarly to Fortnite's slow release of their mobile application) for better controls. Game developers must be from our target groups and will test out the features along with an easy way to report issues and give feedback as soon as the problem occurs. A rubric for a game will be given to ensure a controlled environment and accurate results.

Gamers, ranging from casual players to influencers, test the game created and give their feedback as well.

Public Release & Launch

Incentivize its use; go to university hackathons and offer game developers that use this product a prize if they place(similarly to what Snapchat did with their Spectacles or what Microsoft did with Azure). Feedback reports should be referred back to constantly: is it **scalable**? **Safe**? **Simple to use**? One thing that CWG offers is increased reliability: fix issues fast.

Metrics: Indicators of success are reduced latency times—game loading, multiplayer, etc. The number of developers within groups **1** and **2** should be releasing their games(if not, why?). Long-term goals can then be discussed if it is proven to be successful. If not, further user interviews will be conducted along with feedback reports.

****Thank you for this huge learning opportunity! As someone without any gaming background, it was a unique experience and quite a lot of research. Hope to hear from you soon!**