Cloudflare Workers for Gaming

by Daniel Kunis (CV, GitHub, LinkedIn)

Big Picture Vision

I believe that cloud-gaming is the future of the industry. It will change the way we play and what we play. It creates new business models, new ways of advertising and places the products only a click away from the users.

This is when the **infrastructure becomes** as **crucial** as it has never been before. Game developers will choose the platform which provides both the fastest and the most reliable service. Besides, the user's choice of the platform will rely on the quality and quantity of the provided content. This leads to the "chicken and egg" strategy problem, but if I were to choose — I would start with **building tools and ecosystem for the developers** and when the time comes — users will naturally go for the best content.

My opinion is supported by the strategies employed by the leading tech companies (Google Stadia, Microsoft's Project xCloud, Amazon's Wavelength) and by top VC funds analysts (e.g. Andreessen Horowitz).

Considering all of the above, I suggest to start with **building cross-platform solutions for typical mobile game-development problems** (Mobile Backend as a Service) that will attract game developers to Cloudflare's products and then iteratively, step-by-step, build an ecosystem for cloud-gaming based on the company's infrastructure.

Mobile for 4 reasons:

1. Mobile game development is fast-paced and quickly adopts changes and new technologies
2. Teams and companies are smaller compared to Desktop and Console developers which means that they more often rely on third-party solutions rather than building everything in-house
3. 5G is coming and it will begin its spread on mobile devices
4. The suggested products might be useful for all types of apps

Below I briefly describe the two suggested products that could serve as a starting point for building Cloudflare Workers for Gaming offering to the market.

Product №1: Cross-platform In-App Purchases

One of the most popular app monetization models is In-App Purchases (consumables, non-consumables or subscriptions), but all platforms (mainly App Store and Google Play) have different API's and approaches for managing them and notifying the server-side of the application which inevitably leads to the **problem of data synchronization** between user sessions, different devices and platforms. Cloudflare could provide a "Gold standard" solution for storing vital information in the Workers KV (which is perfect for this purpose) and managing it using the **In-App Purchase Worker**, which would have the following features:

- Server-side receipt validation for different platforms (could even be one as an MVP, like this project for iOS receipt validation) and synchronization based on storing vital information in the Workers KV.
- Integration with Slack (fraud alarms, signals when vital third-party dependancies like markets are down, etc.) and CRM systems, analytics and growth platforms like Amplitude to analyze users (the more, the better, but with focus on leading players).
- Integration with the A/B Testing Worker to allow users to test different subscription schemes
- Template Worker Site with dashboards to visualize the gathered data. Might be used as an alternative to external analytics systems
- Improvement through by providing some lightweight native SDK's for the biggest platforms

Product №2: Cross-platform On-Demand Resources Management

App size is one of the top metrics game developers try to minimize since it directly correlates with number of installs according to different studies. Hence, on of the most popular techniques of app thinning is to move all of the non-essential resources (such as level-specific textures, images, etc.) to a server and download them on-demand. Apple has provided a native SDK for managing such resources and has written a concise introduction to this approach. Right now developers either use different methods for app thinning that are specific for each platform, either configure their own servers to store the resources which at some point becomes a sophisticated infrastructural challenge.
Suggested product — the **On-Demand Resources Manager** Worker — aims to be a cross-platform solution for managing resources stored on the server using the Cloud Storage Worker with the ability to:

- Download resources from provided storage and cache them in the Cloudflare CDN using a straightforward cross-platform API: the request should specify the resource name and client's OS and the worker will return the appropriate asset. This could be the MVP.
- Manage different versions of the resource for A/B tests and old versions support of the app.
- Resample and resize images to optimize assets for different devices. Instead of hosting several versions of each image optimized for different screens — the Worker will resample/resize the original image on the fly and cache the result.

Market research

To create a good product it is crucial not only to investigate user problem and needs, but also to search for current solutions and apply weak spot analysis and other techniques to come up with a better approach. It's useful to keep in mind that if you think of a problem that is not being solved yet - it is not important for the user and hence there is no sense in solving it.

These are the methods I would use to analyze the market needs:

1. **Customer development** through interviews — in this case, the most efficient way to make hypotheses based on the gained insights of game development, though it can be applied only after in-depth investigation of the issue
2. **Analyze reports**. The industry has already been researched and most common problems and their solutions summarized
3. **Browse forums and blogs** to find the hottest and most relevant topics in gamedev
4. **Analyze competitors**. What are the top products of other Mobile Backend as a Service companies and what are their strengths and weaknesses? Apply SWOT analysis for their most and least popular offerings.

Below are the options I believe to be very efficient, but they require much more human resources:

5. Make a local hackathon within the company: set a goal to build a small mobile app. What are the technical drawbacks you encountered? What processes in gamedev would you call pain points? Analyze information gathered from all teams. Moreover, this would help the engineering team understand the user
6. Find a gamedev studio that could host you or one of the engineers for some time. Learn their processes and gain insights of their problems. Visit conferences and engage with professionals.

Methods for improving

The one and only way to find out if the product is promising is to let the users trial it. Join or organize a gamedev hackathon and promote your technology. Make a track and workshops so the developers actually use your product and collect feedback. Organize an exclusive demo with existing Cloudflare's customers from the Gaming industry. The closer to life this test-flight gets — the better.

Risks

mBaaS is a highly competitive field with many single-product companies such as RevenueCat and top tech companies like AWS, which has been acquiring mBaaS companies since 2017, and the main risk is to loose the market share to other companies.

Another risk is to choose a wrong business model. Both mBaaS and Cloud gaming are new and rapidly changing markets, therefore there is yet no proof of concept.

Goals

Precise goals should be set after thorough analysis of the market and estimation of competitor's key metrics. Due to lack of information about competitors and Cloudflare's marketing resources, the values I set as goals are a rough estimation.

Product 1:
Success should be measured by Monthly Tracked Revenue (MTR), total number of apps, Workers, users and Requests Per Day (RPD), Average CPU time per request (ACPR).

I assume RevenueCat has up to 100 paying customers what generates a $100k. To generate a third of that we need to have 330 subscribers assuming each user will generate $10 per month.

Goal: Get 330 monthly subscribers in 3 months after launch

Product 2:
Key metrics are: Requests Per Day (RPD), Daily Transferred Data (DTD), total number of apps, Workers, users and Average Response Time (AVG), Average CPU time per request (ACPR).

Goal: Reach 300k requests per month in 3 months after launch, though in the context of this product I would concentrate on technical metrics rather than those that generate revenue since smooth image delivery is the core technology for cloud-gaming. Thus, the goal is to scale this product in order to test Cloudflare's infrastructure.