How Running A Live Game Impacts Your Code

Brian Apgar
Distinguished Engineer
Zynga Eugene

Quick Introduction

• Who am I?
• Who is Zynga? What do we do in Eugene?

• Disclaimers:
  • Speed
  • Acronyms
  • Language
Running A Game Versus Building A Game

• How Are They Different?
• When Do You Care?
  • What platform(s) are you developing for?
  • Are you selling a game once?
  • Do you have in-game purchases?
  • Do you have downloadable content?

Where Is He Going With This?

• Cheating / Hacking
• Operating Expenses
• Downloadable Content

How those impact the way you write code so you can run a game long-term... or really the intersection of business and game coding.
Cheating / Hacking

- Cheating is awesome!
  - OK, not really. But it’s an interesting problem
- Do I care?
  - Are you building a single-player game?
  - Would my cheating impact your experience?
  - Or are you building a game where your cheating impacts other players (think PvP, or leaderboards...)?

---

Cheating / Hacking

- If other players can see somebody cheating, the game is perceived as unfair.
- People are less likely to spend money in a game they consider unfair.
- This is known as “destroying the economy” and it’s a Bad Thing™.
### Programming Models

- Client-only - insecure, but (shrug)
- Client-authoritative - insecure, but some options
- Server-authoritative (optimistic) – more secure, but not all rainbows & unicorns
- Server-authoritative (lockstep) – very secure, but online-only and impacted by latency

### Programming Models And Cost

- Server-authoritative games are more secure
- But generally harder to develop
- And it costs (potentially a lot) more
  - Up front costs (development time, usually)
  - Ongoing costs (mostly servers, but also storage)

- So you’re not JUST worried about framerate, battery life and thermal-throttling, you’re worried about server-calls...
Operating Expenses

• How do you even estimate/calculate this?
  • Estimated DAU? Steady or cyclic?
  • Frequency of calls?
  • CPU time per call?
  • Load testing/server types and ‘padding’?
• Magic around auto-scaling groups or Lambda
• The differences can be Real Money™

How About An Example?

• 1M DAU, evenly distributed.
• Each player plays for 30 minutes/day.
• Each client sends a batch of data every 20s.
• Server takes 200ms to process a batch.
• That’s 90M calls / day and 18M CPU-seconds.
• ...
• 208 CPU-days, or ~40 8-core servers running 70%
• ~$7k/month for hardware... but can swing wildly!
Oh Yeah... DevOps

- If you have servers, you likely need to manage upgrades & versions.
- How does your server deal with new clients, or new downloadable content?
- How does your server deal with old clients?
- Are you running multiple server versions?
- Does your game have downtime when you upgrade? Or do you have rolling releases?

Downloadable Content

- How do you handle playing against somebody who has content you don’t have / haven’t downloaded?
- How do you handle the memory pressure of a potentially infinite set of collectable items?
- How do you make sure that everybody in a multiplayer event or leaderboard is using the same version/data?
How Do You Release New Content?

- If you need to patch, update, tune or otherwise release something, how do you do it?
  - Apple / Google have approval processes.
  - Stores take time to propagate to all countries.
- Can you wait for that? Do you build a system to download within your game?
  - Tend to be complex systems. Do it early!

Quick Recap

- We’ve talked briefly about cheating, and options to prevent it.
- We’ve talked briefly about game content, and concerns around versioning and getting it to your players.
- We’ve talked about how those decisions impact the $$ of keeping a game running.
- One last aside: How do you know if it’s working?
Stats / Data Collection

• What kinds of data do you need to collect?
  • Business data
  • Technical data
    • Load funnel
    • Performance metrics (by device type?)
    • Crashes?
    • Soft-locks, or exceptions.
    • Asserts / Log messages?

• Where / how do you collect it?

What’s Actually In Stats?

• Do you have enough information to diagnose & fix a problem?
• Or just enough to know that it’s happening?
• In some of our games, we record inputs.
  • If you get into a bad state, we send the inputs to our logging servers
  • Can replay to see what went wrong – if we’re deterministic!
• Logging isn’t free (see operating expenses, above)
• Kinda’ like insurance
Wrap Up...

- Building a game isn’t just about finding the fun (which is hard enough by itself).
- The business of running a game impacts technical decisions – often significantly!

Questions?

- Is anybody still awake?

Thank you!