Architecting Effective Map Streaming in Low-Bandwidth Scenarios

An inspiring and amazing research question that everyone wants to know, but was too afraid to write a 2000-word essay about.

Louis Foy

Falmouth University

March 13, 2018

The Question

- Which C++ architecture(s) could best support QoE-oriented map streaming in a player-building-based multiplayer game?

A scenario

- My company wants to make a large-scale, multiplayer building game
- Players can build complex structures in real-time
- It's releasing across the world

A problem

- Map data downloads take a lot of bandwidth
- Large markets, such as Egypt, Brazil, and China, suffer from slow broadband
  - Even though China is the worlds strongest gaming market! [1]
- This risks leaving players waiting for downloads

Intro: QoE

- QoE means Quality of Experience
- Focuses on directing data flow to the users best interests first
  - For example, downloading their characters running animations, before downloading the distant city of Goradonia
- Commonly used in video streaming to balance motion and detail

User experience != bitrate

- User experience is more complex!
- For example, evidence suggests users prefer a smooth low-quality video over higher-quality, inconsistent video [2] [3]
- The amount of action in a scene can affect their perception [3]
  - More action, less focus on detail. Less action, more focus. Could this apply to games as well?
- In a nutshell, intelligent prioritising of streams is key

Prerequisites: Map streaming

- To identify ideal architectures for implementing QoE map streaming, lets explore the necessary ingredients

Prerequisites - Baseline

- Map data uploader
  - Sends map data to clients
- Map data downloader
  - Takes map data from server
  - Must tolerate incomplete data
- Map constructor
  - Builds the map in real-time
  - May include collision generation, textures, and LoD

Prerequisites - With QoE

- Map data prioritiser
  - Decides which map data each player needs soonest
  - Must work with data uploader
- Map data compressors
  - Reduces bandwidth by brute force
  - Adds processing time on downloads: may introduce FPS lag
- 'Network LOD' generators
  - Reduces details on distant player-made areas
  - Different to regular LOD–regular LOD is predefined on your local PC

Preliminary architectural theories

- Multi-threading
  - Could reduce distressing lag during downloads
- Adapters?
  - Could wrap compressed map data into usable geometry
- Observers?
  - Could live in the map class, awaiting incoming map data
- Or mediators?
  - Alternative to data transferral between networked map data and geometry
  - Which one is better?
- OOP?
  - OOP principles support all the above design patterns quite well

More to come

- More specific design patterns will be isolated pending further research into existing game titles

Any questions?

- References:


C. Cui and P. Harding-Rolls, "China - largest games market continues its growth story." `https://technology.ihs.com/593405/china-largest-games-market-continues-its-growth-story`, jun 2017.
Accessed: 13/03/2018.


B. Villa, K. De Moor, P. Heegaard, and A. Instefjord, "Investigating quality of experience in the context of adaptive video streaming: findings from an experimental user study," in *Norsk informatikkonferanse NIK*, pp. 122–133, nov 2013.


D. C. Robinson, Y. Jutras, and V. Craciun, "Subjective video quality assessment of http adaptive streaming technologies," *Bell Labs Technical Journal*, vol. 16, pp. 5–23, March 2012.