ARCHITECTURE RECONSTRUCTION & DOCUMENTATION

Ashish Shrestha | as3828@drexel.edu

1. INTRODUCTION

Discord is a propriety VoIP solution built with gamers in mind. Although, primary designed for voice chat, Discord offers a lot more. As all other VoIP application Discord offers friend list, private chat, voice call, group chat, group voice call, video calling and screen sharing capabilities. But, what sets discord apart from the rest is its sense of community.

Unlike traditional VoIP applications, Discord offers a dedicated private space for people to socialize and build connections. Discord calls this space a server, also called “guilds”. People can join any server on invitation. A server hosts channels, and channels can be of two types – text and voice. Text channels are dedicated spaces to text chat on a particular channel topic with guild members. While voice channels are like chat rooms, where people are free to connect and speak at anytime.

As stated previously, Discord was designed with gamers in mind, but the application has grown to a stage where it is now a hybrid of Slack and TeamSpeak, allowing for many non-gaming communities to set up shop. Anything from anime, science and technology, and education to helpdesk chats can now be found on Discord.

2. DISCORD OVERVIEW

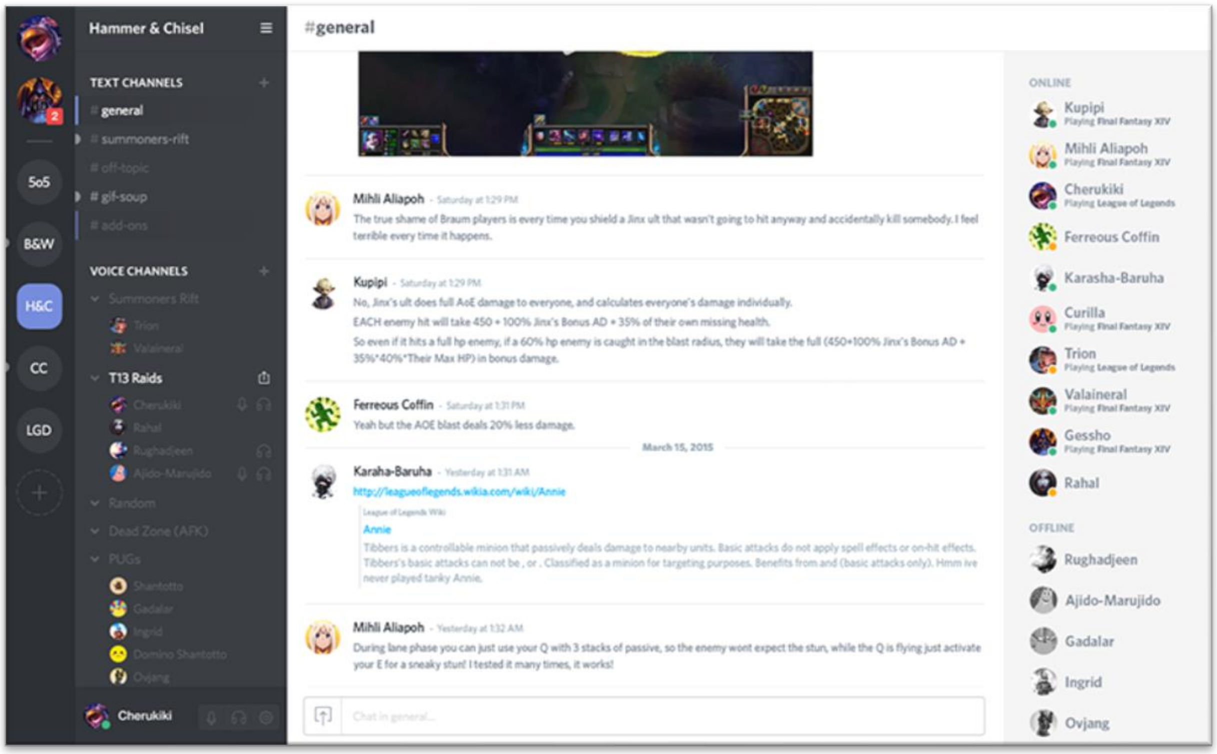
Discord architecture is close to that of Slack than to that of TeamSpeak. Like Slack, Discord uses the concept of channel for text chats, where server admins can create channels as dedicated space for certain topics. Discord takes it a step further with voice channels. Voice channel like text channel allows members to easily join in on the conversation, hence any disruption due to new member joining in or leaving is non-existent.

i. Server

Additionally, like Slack, guilds are hosted by Discord on their own server. Server admins can choose which geographical region they want the guild to be hosted from, but the eventual control of all content is with Discord. TeamSpeak on the other hand provides TeamSpeak server for users to host themselves. The advantage Discord has over TeamSpeak are 100% uptime, low latency, and preservation of chat history as guild don’t have to deal with local setup issues.

But all this also results in giving up data rights to Discord. This raises privacy issues, as the creators are not the true owners of the content they post, and there are no guarantees that Discord will not look through it. Discord does have URL sniffers built in to channels and they cannot be turned off. These URL sniffers looks up posted URLs and provides short description about its content to the users. A very helpful feature, but one that proves that Discord does look up what one posts. Discord has also banned certain kinds of content on their platform, which further affirms that communities are not immune to Discord prying eyes.

Even than privacy on Discord is far superior to that of Facebook, as user identity, in most cases, don’t correspond to real life identities. Anonymity and privacy has also allowed Discord to be used by alt-right groups to plan and organize rallies.



Mihli Aliapoh - Saturday at 1:29 PM
The true shame of Braum players is every time you shield a Jinx ult that wasn't going to hit anyway and accidentally kill somebody. I feel terrible every time it happens.

Kupipi - Saturday at 1:29 PM
No, Jinx's ult does full AoE damage to everyone, and calculates everyone's damage individually.
EACH enemy hit will take 450 + 100% Jinx's Bonus AD + 35% of their own missing health.
So even if it hits a full hp enemy, if a 60% hp enemy is caught in the blast radius, they will take the full (450+100% Jinx's Bonus AD + 35%*40%*Their Max HP) in bonus damage.

Ferreous Coffin - Saturday at 1:31 PM
Yeah but the AOE blast deals 20% less damage.

March 15, 2015

Karaha-Baruha - Yesterday at 1:31 AM
http://leagueoflegends.wikia.com/wiki/Annie
League of Legends Wiki
Annie
Tibbers is a controllable minion that passively deals damage to nearby units. Basic attacks do not apply spell effects or on-hit effects. Tibbers's basic attacks can not be , or . Classified as a minion for targeting purposes. Benefits from and (basic attacks only). Hmm ive never played tanky Annie.

Mihli Aliapoh - Yesterday at 1:32 AM
During lane phase you can just use your Q with 3 stacks of passive, so the enemy wont expect the stun, while the Q is flying just activate your E for a sneaky stun! I tested it many times, it works!

Figure 1. Discord Windows Client UI showing "Hammer & Chisel" Guild

ii. Client

On the client side, Discord covers a vast array of platforms – Windows, Mac OS, Linux, Android, iOS and even 100% feature rich web application.

Discord client, arguably, provides the best chat experience compared to all other similar applications. Discord is built on the feeling of being part of a community, and it shows. Direct/private messages are essentially hidden on the top left portion of the screen as it is almost not meant to be used, while guilds are front and center, and get all the attention.

iii. Other components

Apart from the main client and server components, Discord provides additional features bundled either on the server or client.

Back in 2016, Discord announced **GameBridge API**. GameBridge API allows game developers to directly integrate Discord into their game.

Rich Presence provides detailed game stats and integrations, allowing members to not only view what game their friends are in, but also view what stage of the game they are in and their game stats. Additionally, with Rich Presence, friends can also spectate the game.

Discord Application enables developers to create RESTful web services based applications that provide customs features on guild servers. Applications can also act as administrators, automating the process of moderation.

Discord Bot is an additional feature provided through Discord application, that enables developers to create interactive bots that can communicate to users via both text and voice.

Webhooks are also provided on channel level, which facilitates easy posting through REST calls.

Discord RPC enables developers to control locally running Discord clients. It is also used by features such as GameBridge API.

3. TECHNICAL ANALYSIS

Being a propriety and relatively new product, there are very few technical and architectural description of Discord. The below mentioned content is obtained and inferred from the official Discord Engineering Blog (https://blog.discordapp.com/tagged/engineering), Discord developer documentation (https://discordapp.com/developers/docs/intro, and official Discord GitHub repository (https://github.com/discordapp).

i. PC Client(s)

PC Client for Discord is written on Electron framework. Electron is a web technology based application development platform that uses Chromium and Node.js to build native application experience for MacOS, Windows and Linux using HTML, CSS and JavaScript. Other applications that uses Electron are Slack and Atom editor.

Through Electron, Discord provides automated client updates via Squirrel Updater. Electron also enables Discord to provide native experiences like UI elements and notification services. Furthermore, Discord also uses Electron for crash reporting and content tracing. Additionally, Discord uses EventEmitter3, a high-performance event emitter as a replacement for the default Node.js EventEmitter.

Apart from being able to provide consistent experience throughout different platform (Linux, MacOS and Windows), Electron application also runs in a web browser. Due to this common codebase, Discord is not only able to support multi-platform but also streamline development, without needing resources for each individual platform.

ii. Discord RPC

Discord RPC is also bundled in the client. All Discord clients have an RPC server running on localhost that allows control over local Discord clients. Discord RPC is written in C++ and the source code is provided on Discord's Github repository.

Discord RPC is set up to process web socket connections and proxy API requests, and enables direct access and control of the client. Discord RPC communicates using JavaScript Object Notation (JSON).

iii. GameBridge SDK

The GameBridge Software Development Kit is a headless, optimized version of Discord RPC available to game developer as a Dynamic Link Library (DLL). Once started, it will spin up an RPC server and relay information about the port it is running on.

iv. Rich Presence

Rich Presence is a feature from Discord that allows game developers to surface unique, interesting, and actionable data inside a Discord user's profile when they play. It also enables features like spectating friend's game and joining friend's game party or server.

Rich Presence is available as an SDK for C, C++, Unity and Unreal game engine to game developers, and it is up to the game developer to implement it. Rich Presence also uses Discord RPC.

v. Mobile Clients

Discord again went for a multi-platform framework for their mobile client for iOS and Android. Discord mobile Client uses React Native. React Native allows development of native mobile apps using JavaScript and React, enabling rich mobile UI, rendering it indistinguishable from an app built using Objective-C or Java. Additionally, React Native allows for native code to co-exist - components written in Objective-C, Java, or Swift.

React Native runs JavaScript on a background thread and sends a minimal amount of code to main thread. Benchmarks show that there is little performance difference between this and native iOS apps which are written in Objective-C or Swift. Frontend developer at Discord, Fanghao Chen states that they also made use of a catalog of open source component for React Native – js.coach.

vi. Server

It is not completely public knowledge on what comprises of Discord Server. A few hints from Discord engineering blog speak about Google Cloud Platform, and use of Google Firebase Cloud Messaging service.

Upon further research and studying the available source code provided by Discord, it was found that Discord uses Elixir and Erlang. Elixir is a dynamic, functional language designed for building scalable and maintainable applications. Erlang is a programming language used to build massively scalable soft real-time systems with requirements on high availability. It is used in telecoms, banking, e-commerce, computer telephony and instant messaging. Elixir leverages the Erlang VM for its known support for high concurrency, distribution and fault-tolerance.

For scaling to millions of push requests per minute, Discord uses Elixir's GenStage. GenStage is an Elixir behavior for exchanging events with back-pressure between Elixir processes, preventing system bottle-neck and overload. GenStage implementation on Discord uses producer and consumer architecture, where a push collector acts as a producer and the pusher to Firebase acts as consumer. This seems like a better solution than using dedicated queueing systems like Kafka which would end up taking more resources.

vii. Database

Before Discord started gaining huge userbase, it used a single MongoDB replica set to store everything on Discord. This was an intentional design decision, and a temporary one. The database was structured to be easily migrated when the company hit scaling/storage issues.

Studying user pattern and database uses, the Discord team came up with a list of requirements for their database of choice - Linear scalability, automatic failover, low maintenance, proven to work, predictable performance, not a blob store and open source.

Cassandra was found to be database that fulfilled the requirements as nodes could be added to scale it. Cassandra can also tolerate a loss of nodes without any impact on the application. Related data is stored contiguously on disk providing minimum seeks and easy distribution around the cluster. Large companies such as Netflix and Apple have thousands of Cassandra nodes.

While Cassandra was an apt choice for Discord, it is not without flaws. Cassandra has eventual consistency meaning that deletes on Cassandra is not immediate. Cassandra instead treats delete as "tombstone" writes. So, in a scenario where there are millions of deletes and eventually only a few data rows left, Cassandra must scan thorough all these tombstones, effectively slowing down reads.

Discord "solved" these issues with lowering tombstone lifespan and adding an additional query to track and ignore empty buckets.

viii. Indexing

Searching is a big problem when the set is billions of messages. Discord uses ElasticSearch to solve this issue. ElasticSearch is a search engine based on Lucene, which provides distributed, scalable full-text search via lazily indexing documents.

My previous experience with Solr, and then eventual migration to ElasticSearch at Dell Boomi, showed me the greater capability, performance and scalability of Elasticsearch. So, did the team at Discord. Although ElasticSearch supports single document indexing, it performs better with bulk, so Discord implemented a queue for bulk feeding.

ix. Voice

Discord uses VoIP and OPUS codec for relaying voice data. Opus is a free and open audio codec with unmatched performance for relaying speech over the internet.

Discord auto detects internet quality and adjust bit rate accordingly and dynamically switches server region to provide the best voice experience.

x. Extras

Discord comprise of many more features than the one listed above.

One of the many small things that user will not notice is how Discord handles 150 million images every day. To prevent image loss and IP address leakage of images posted on Discord, Discord automatically resizes images using Go and C++. Discord created its own image resizer on Go, aptly named Liliput, using C libraries and OpenCV for compression and FastHttp for concurrent HTTP communication. Liliput outperforms pillow-simd package for resizing which uses x86 SSE instructions.

4. ARCHITECTURE

From the prior studies, Figure 2 shows the architectural reconstruction of Discord. As all web services based applications, Discord follows a client server model. Each of this model however has custom components attached to it that enables for unique experiences only provide by Discord.

5. CONCLUSION

All the technologies and components used by Discord are well researched decisions and implementations. Discord has the best voice quality, making it well-loved by its user community. The performance numbers and service provided by Discord shows that the design decision made by Discord team is rational and fruitful. If there are any changes to be made to the components, Discord team has already been actively searching for newer solutions, as evident by their post on their engineering blog.

6. REFRENCES

- Discord Engineering blog - https://blog.discordapp.com/tagged/engineering
- Discord Developer Documentation - https://discordapp.com/developers/docs/intro
- Discord Github Repository - https://discordapp.com/developers/docs/intro
- "Apps Built on Electron" - https://electronjs.org/apps?q=discord
- Roose, Kevin (August 15, 2017). "This Was the Alt-Right's Favorite Chat App. Then Came Charlottesville". The New York Times. Archived from the original on August 19, 2017. - https://www.nytimes.com/2017/08/15/technology/discord-chat-app-alt-right.html

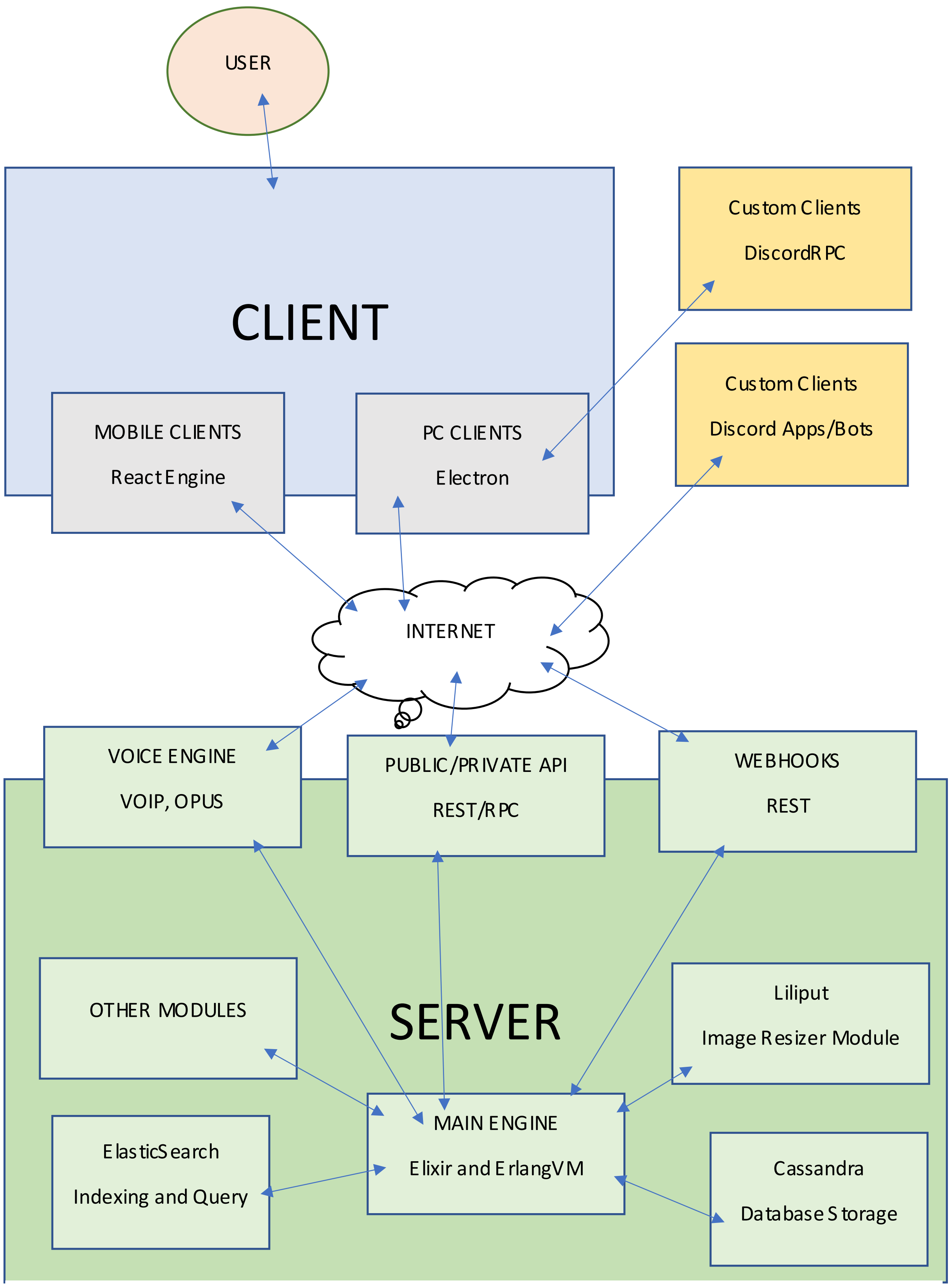


Figure 2. Discord Architecture Reconstruction