

Nintendo Music を 支える技術

Google
Cloud
Next

Tokyo

Proprietary



はじめに

本発表において、
登壇者の撮影はお控えくださいますようお願いいたします。
スライド自体の撮影は問題ございません。

灘友 良太

ニンテンドーシステムズ
システム開発部
チーフ



相馬 啓佑

ニンテンドーシステムズ
システム開発部
チーフ



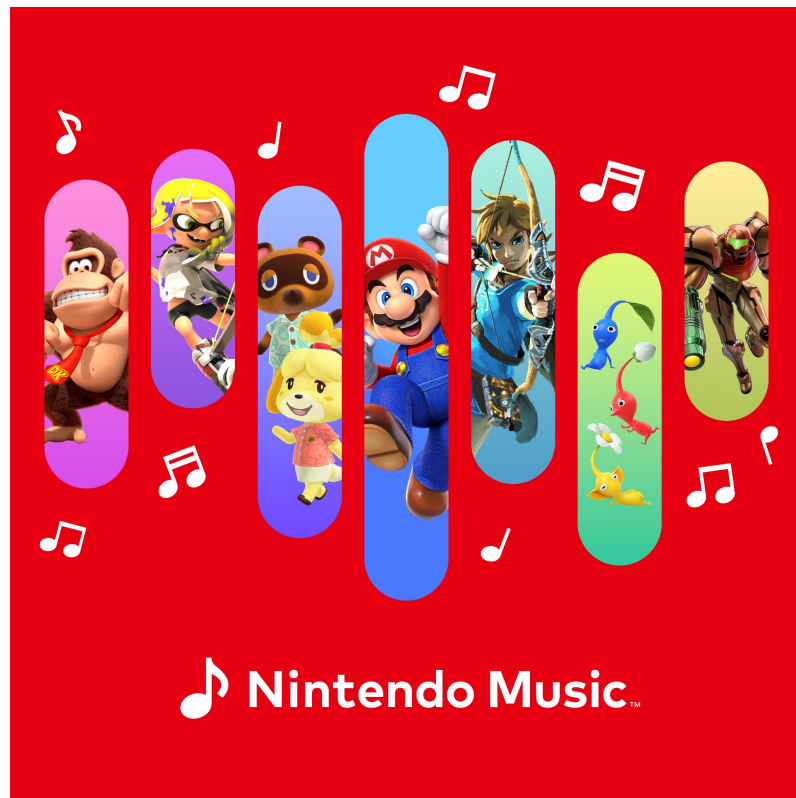
本日のアジェンダ

- 01 Nintendo Music のご紹介
- 02 ニンテンドーシステムズ担当箇所
- 03 ゲーム音楽探索体験を支える技術
- 04 ゲーム音楽再生体験を支える技術
- 05 まとめ

01. Nintendo Music のご紹介

01. Nintendo Music のご紹介

- 任天堂が提供するゲーム音楽配信アプリ
 - Nintendo Switch Online 加入者限定
- **40 以上の国と地域・16 言語**に対応して配信
- さまざまな年代のゲーム音楽、
さまざまなテーマのプレイリストの提供



01. Nintendo Music のご紹介

- ゲーム音楽ならではの体験
 - 曲ごとに設定されている**画面写真**
 - 対応した曲をゲーム中の BGM のように途切れることなく流すことができる「**ながさチェンジ**」



02. ニンテンドーシステムズ担当箇所

02. ニンテンドーシステムズ担当箇所



DeNA

ニンテンドーシステムズ株式会社は
任天堂株式会社と株式会社ディー・エヌ・エーの合併会社です

02. ニンテンドーシステムズ担当箇所

フロントエンド開発

- Android / iOS アプリ開発
- コンテンツ制作



バックエンド開発

- API の提供
- メディア配信
- メディア配信処理のワークフロー構築
- アプリとメディア処理をつなぐ SDK の提供
- etc...



**NINTENDO
SYSTEMS**

03. ゲーム音楽探索体験を支える技術

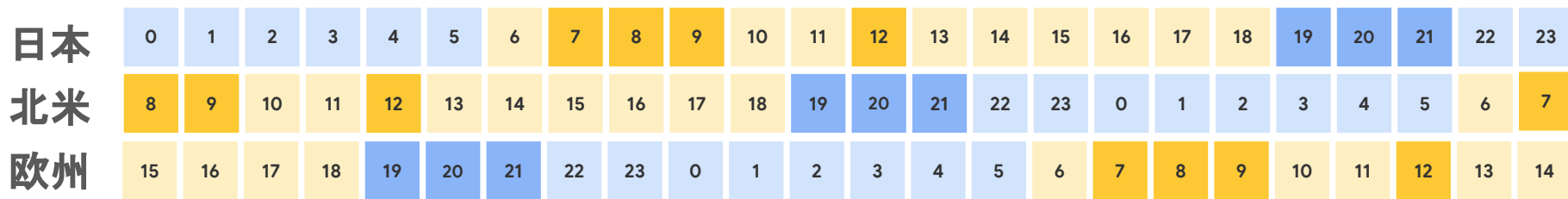
03. ゲーム音楽探索体験を支える技術

期待される特性

1. グローバルでのサービスを提供

→ どの時間帯でも常に安定して稼働する必要がある

図. 各タイムゾーンごとの稼働状況



昼間の時間帯

夜間の時間帯

各時間帯のピーク

03. ゲーム音楽探索体験を支える技術

期待される特性

1. グローバルでのサービスを提供
2. **突発的なアクセス増加に対応可能**
→ 事前準備のあるなしに関わらずサービス提供を継続する

03. ゲーム音楽探索体験を支える技術

期待される特性

1. グローバルでのサービスを提供
2. 突発的なアクセス増加に対応可能
3. **安定したユーザー体験の提供**
→ 快適な音楽体験のために、高速に安定したレスポンスを

03. ゲーム音楽探索体験を支える技術

期待される特性

1. グローバルでのサービスを提供
2. 突発的なアクセス増加に対応可能
3. 安定したユーザー体験の提供



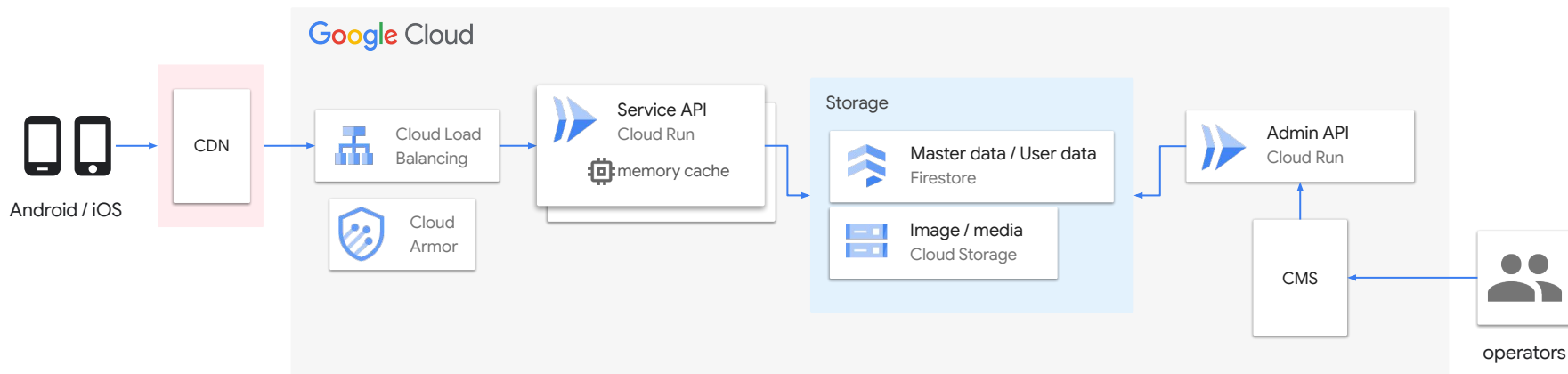
今回のプロジェクトでは ...

Google Cloud を採用

- 実行基盤に Cloud Run
- 起動速度・スケーラビリティに期待

03. ゲーム音楽探索体験を支える技術

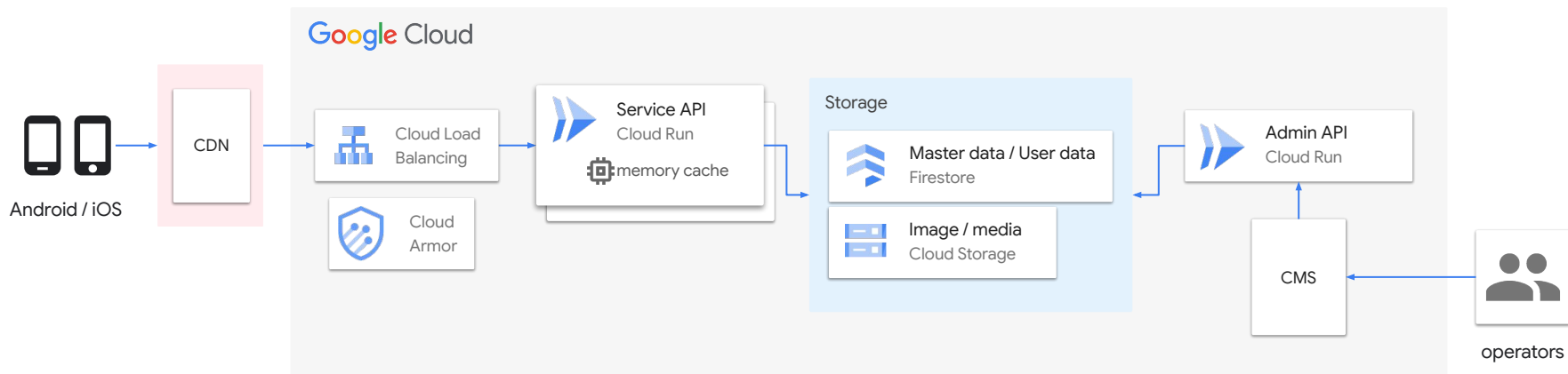
サーバー構成



保守工数削減 スパイク耐性	Cloud Run の採用	マネージド且つスケールが容易 コンテナのライフサイクルが短く復元力が高い
	Go 言語の採用	コンテナの迅速なスピンナップによるスケールアウトの高速・安定化
	Firestore の採用	Datastore モードで利用、マネージド且つスケールが容易

03. ゲーム音楽探索体験を支える技術

サーバー構成



安定した サービスの提供	Cloud Load Balancing の採用	IP Anycast、および TLS 終端
	CDN の活用	マスターデータ・アセットのキャッシュ 経路最適化
	Cloud Run 上の memory cache	永続化層へのアクセス低減

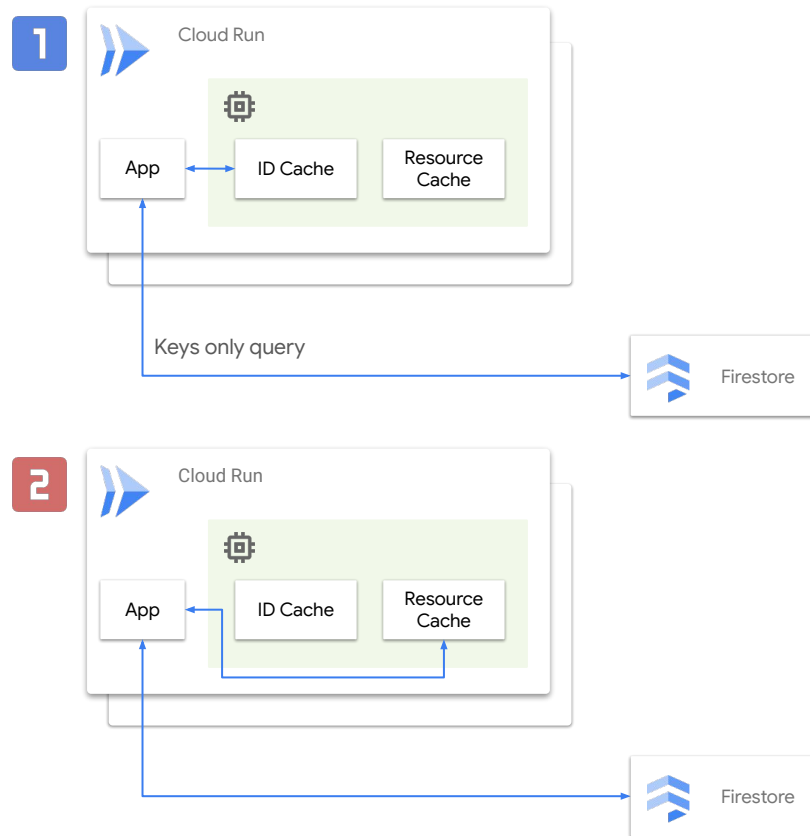
03. ゲーム音楽探索体験を支える技術

メモリキャッシュ戦略

- メモリキャッシュを使う理由
 - Hotspot 問題の解決
 - 高速化・コスト最適化
- 下記 2 つの流れでリソースを取得

1 取得したいリソースの ID を解決する

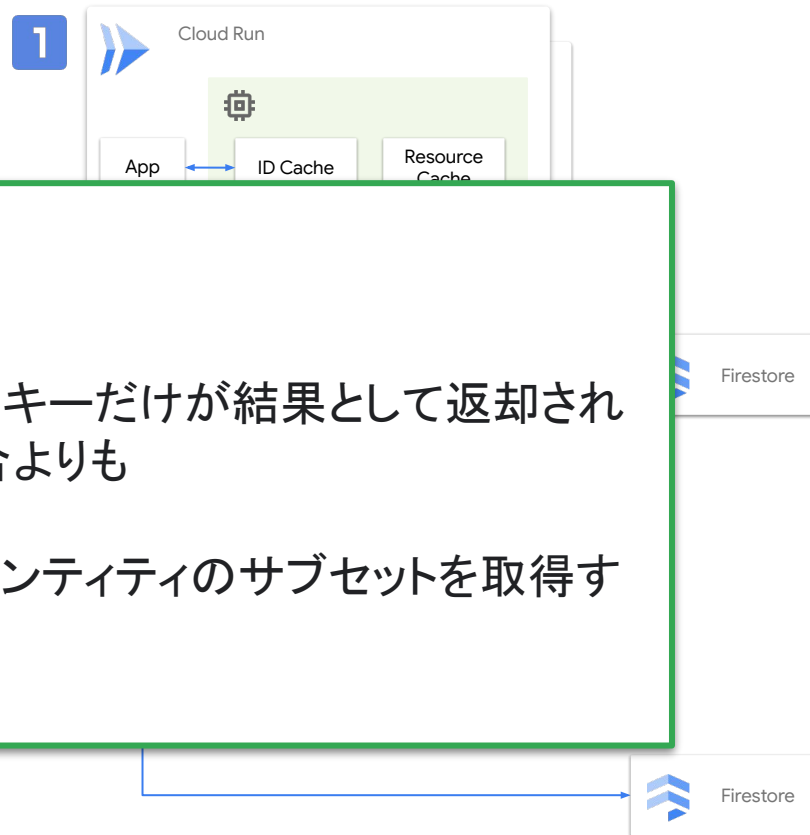
2 解決された ID からリソースを取得する



03. ゲーム音楽探索体験を支える技術

メモリキャッシュ戦略

- Keys Only Query ?
 - 射影クエリ的一种
 - エンティティ自体ではなく、エンティティのキーだけが結果として返却されるため、エンティティ全体を取得する場合よりもレイテンシとコストが低くなる
 - まずキーのみを取得し、その結果からエンティティのサブセットを取得する形

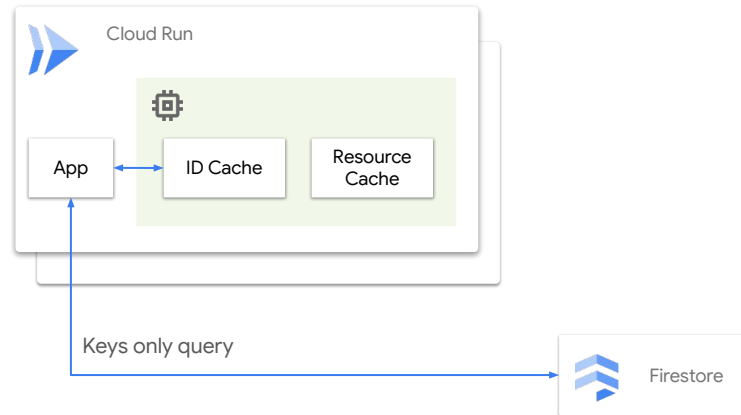


03. ゲーム音楽探索体験を支える技術

メモリキャッシュ戦略

1 取得したいリソースの ID を解決する

- 取得したいリソースの ID を解決して、解決された ID を「**ID Cache**」としてキャッシュ
- ID 解決には Keys Only Query を活用
- 最初から ID 指定でリソースを要求してくるケースについては本段階はスキップ

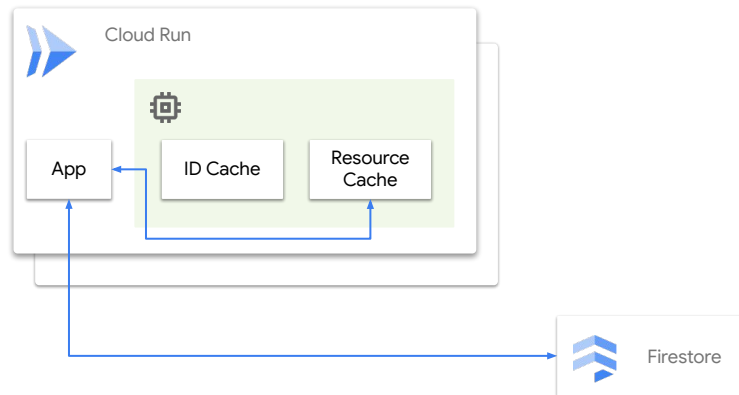


03. ゲーム音楽探索体験を支える技術

メモリキャッシュ戦略

2 解決された ID からリソースを取得する

- 第1段階で取得されたID、もしくはアプリから直接指定されたIDに紐づくリソースの取得を実施し、「**Resource Cache**」にキャッシュ
- キャッシュにないリソースのみ Firestore から直接取得



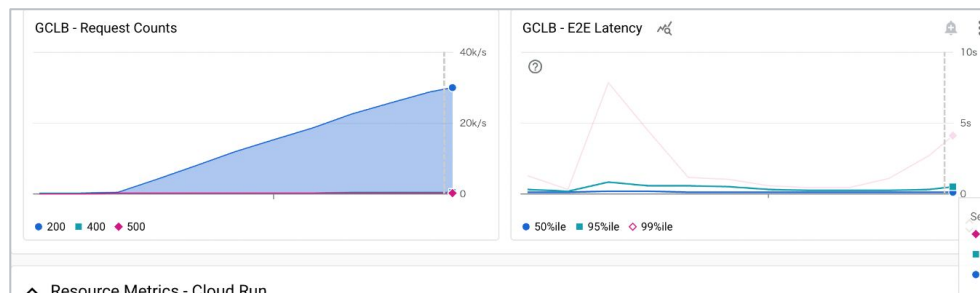
03. ゲーム音楽探索体験を支える技術

性能評価

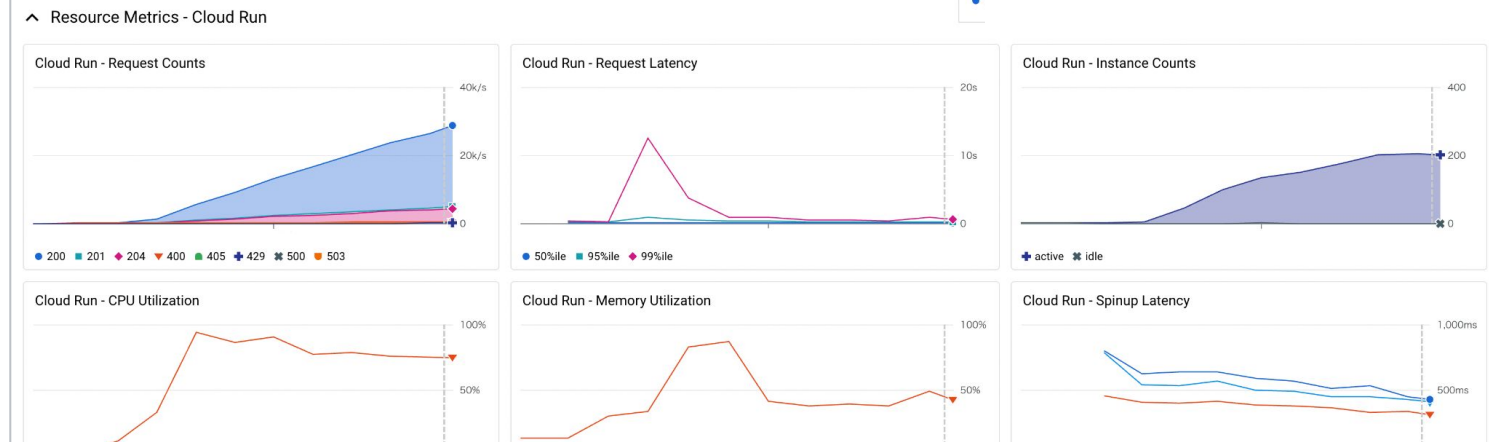
- 負荷テストで性能評価を実施
- 負荷テストには k6 を Google Kubernetes Engine 上で動かす形で実施
 - 並列実行する Pod 数を制御することで、負荷の増減を柔軟に制御可能
- 下記2つのケースで検証
 - **ローンチやコンテンツ追加を想定したスパイク的アクセス**
→ スケール性能・特性の確認と最適化が目的
 - **定常的にアクセスが来るロングランテスト**
→ 計算資源や同時接続数の最適化が目的

03. ゲーム音楽探索体験を支える技術

性能評価 ～スパイク的アクセス～

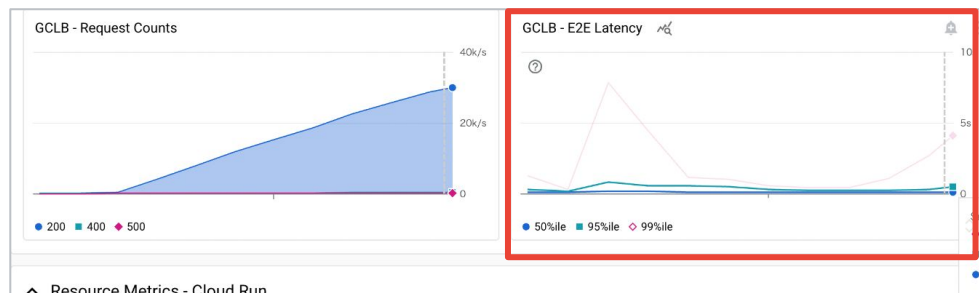


- Nintendo Switch Online 会員ほぼ全員が一気にやってくる想定を検証
 - 最大で秒間 30,000 リクエスト

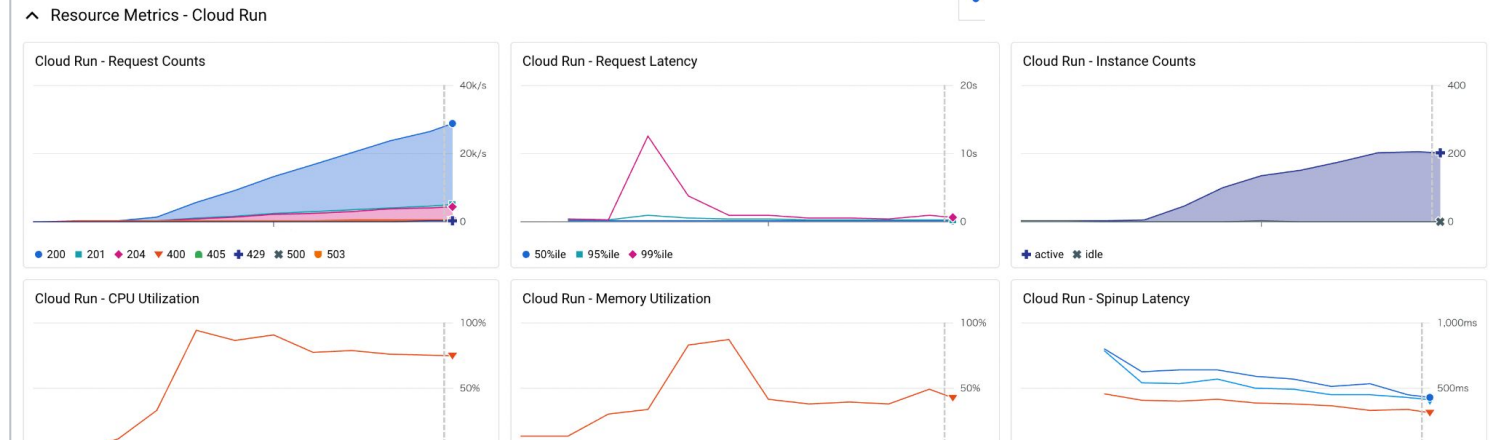


03. ゲーム音楽探索体験を支える技術

性能評価 ～スパイク的アクセス～

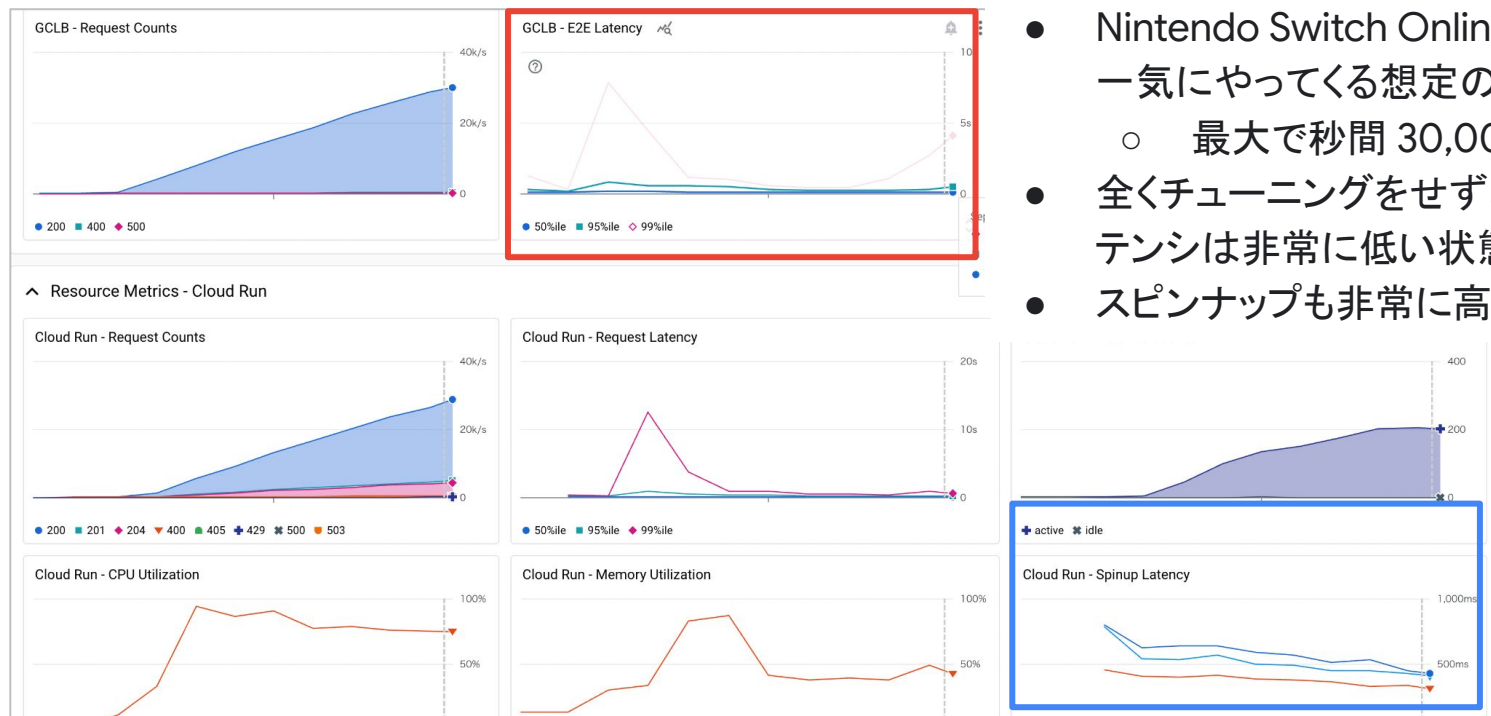


- Nintendo Switch Online 会員ほぼ全員が一気にやってくる想定を検証
 - 最大で秒間 30,000 リクエスト
- 全くチューニングをせずとも、95%ile の レイテンシは非常に低い状態



03. ゲーム音楽探索体験を支える技術

性能評価 ～スパイク的アクセス～

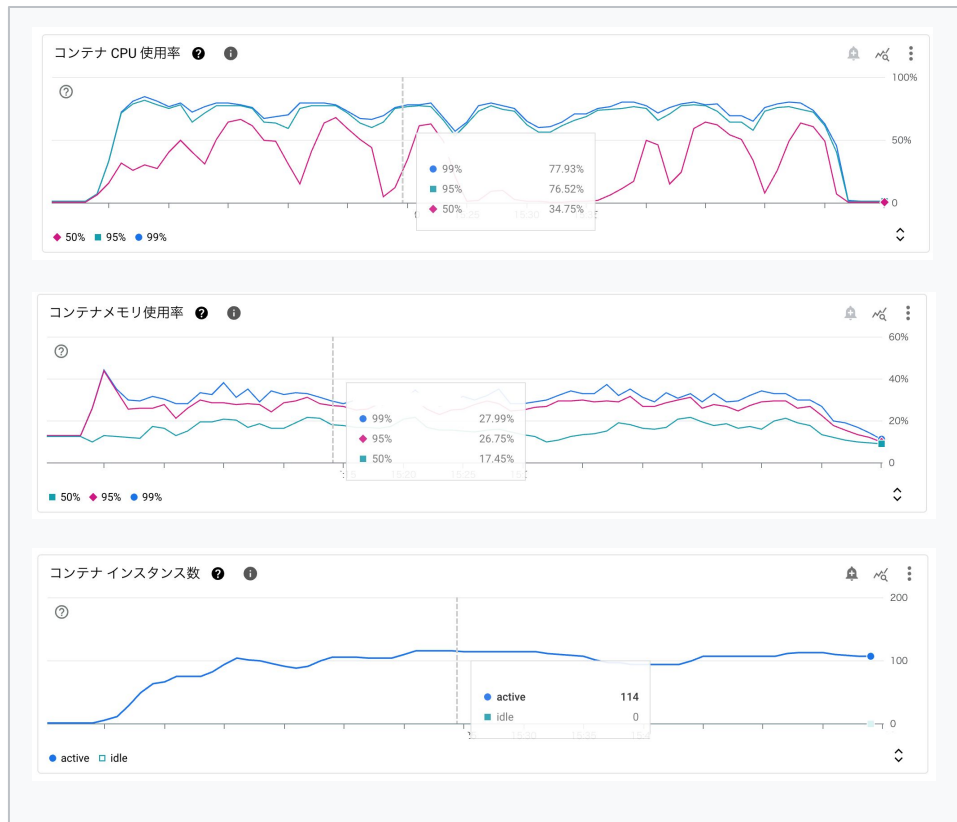


- Nintendo Switch Online 会員ほぼ全員が一気にやってくる想定を検証
 - 最大で秒間 30,000 リクエスト
- 全くチューニングをせずとも、95%ile の レイテンシは非常に低い状態
- スピンナップも非常に高速で安定している

03. ゲーム音楽探索体験を支える技術

性能評価 ～ロングラン テスト～

- 定常的にお客様がアクセスしてきている状況の検証
- CPU / メモリともに安定
- レイテンシについてもプロジェクト内で定めた SLO を満たすことができる水準



04. ゲーム音楽再生体験を支える技術

04. ゲーム音楽再生体験を支える技術

- 01 メディアチームの役割
- 02 各コンポーネントの紹介

Media Pipeline

Media Server

Media SDK

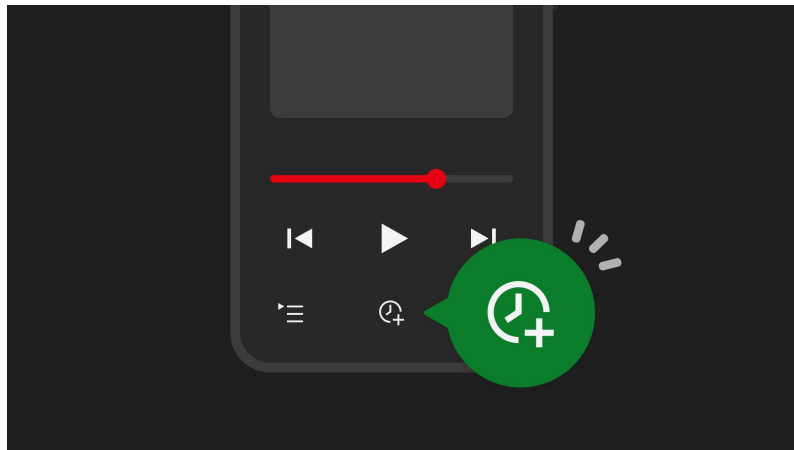
- 03 ながさチェンジ機能
- 04 再生を待たせない仕組み
- 05 QoE データの活用

04. ゲーム音楽再生体験を支える技術

メディアチームの役割

2つの再生体験を提供するチーム

- **ゲーム音楽らしい** 再生体験
 - ながさチェンジ機能
- **快適** な再生体験
 - 再生を待たせない仕組み
 - QoE データの活用



04. ゲーム音楽再生体験を支える技術

メディア配信を構成するコンポーネント

Media Pipeline

配信する用の
メディアの作成



Dataflow

Media Server

Pipeline で作成した
メディアの配信



Cloud Run

Media SDK

Server から配信された
メディアの再生



Android / iOS

04. ゲーム音楽再生体験を支える技術

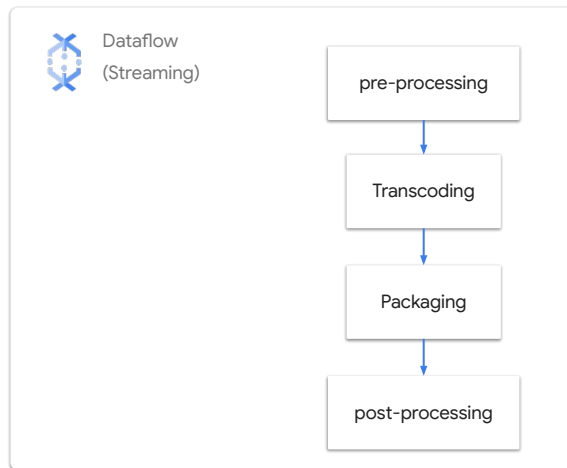
Media Pipeline

Transcoding : 音源を AAC ファイルに変換(ファイルサイズの削減)

- ネットワーク環境に応じて音質を変更するために3つの音質を作成
 - High, Medium, Low

Packaging : 暗号化と配信用のファイルの作成

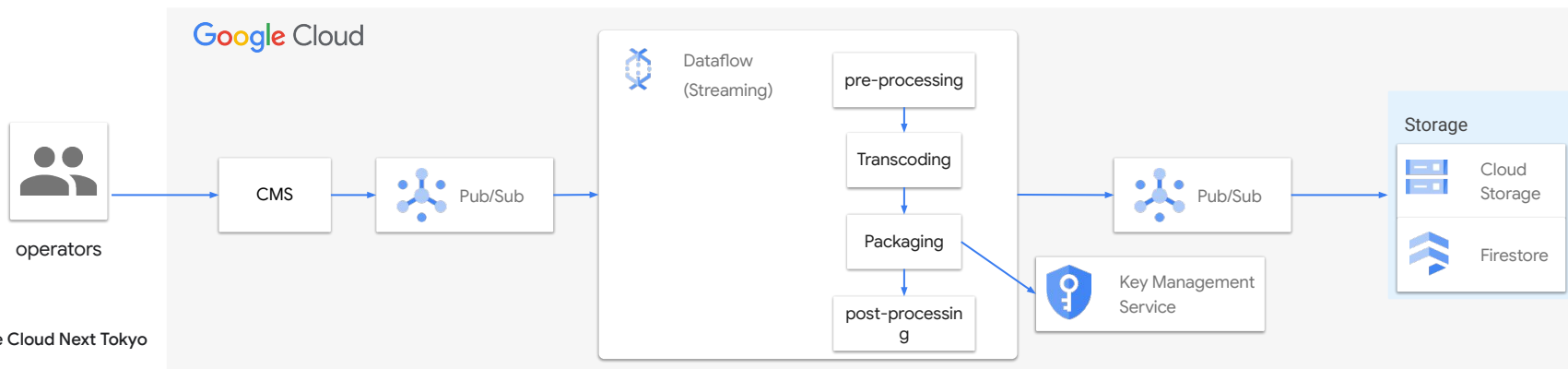
- DRM : FairPlay Streaming, Widevine
- フォーマット : HLS, MPEG-DASH
- 再生タイプ : Streaming, Download



04. ゲーム音楽再生体験を支える技術

Media Pipeline

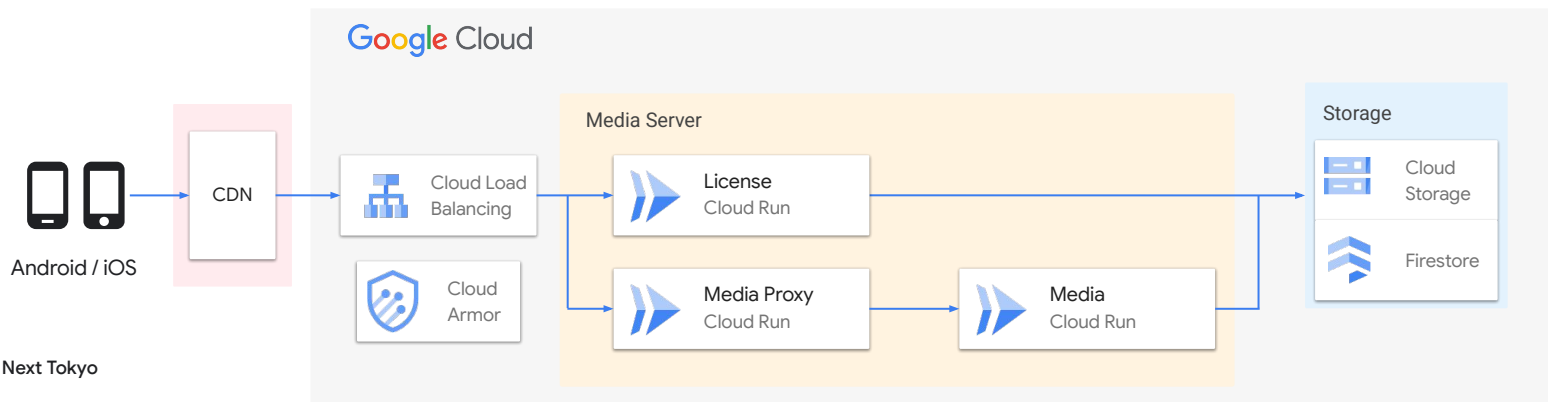
- Dataflow による高いスケーラビリティ
 - 10 時間分の楽曲を約 3 分で処理可能
- 独自の再生方式や暗号化方式に柔軟に対応できるように内製化



04. ゲーム音楽再生体験を支える技術

Media Server

- Media Proxy サーバー
 - 公開状態など配信の可否の確認
 - Media Pipeline で作成したファイルの配信
- License サーバー
 - データ復号のためのライセンスの発行

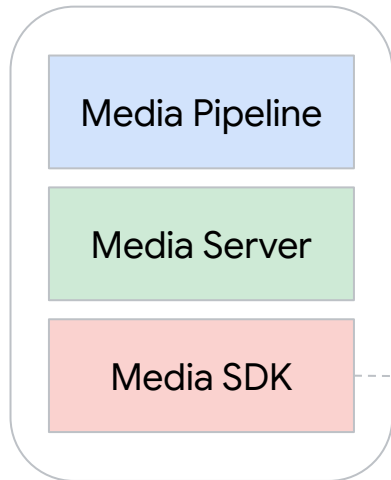


04. ゲーム音楽再生体験を支える技術

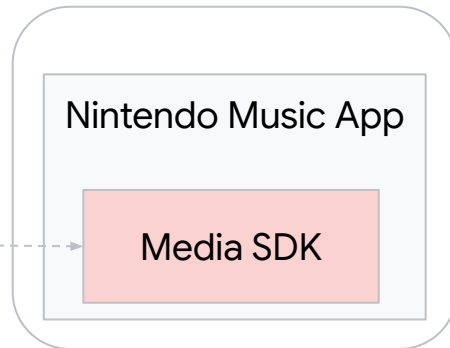
Media SDK

- 取得したファイルとライセンスを利用し楽曲を再生
- Media SDK はバックエンドとの関連性が強い**ためメディアチーム**で開発
- 各コンポーネントの動作確認用のアプリケーションの開発

メディアチーム



アプリチーム



提供



04. ゲーム音楽再生体験を支える技術 利用技術

- 各プラットフォームに最適な技術を採用

OS	言語	プレイヤー	配信フォーマット	暗号化方式
Android	Kotlin	media3	MPEG-DASH	Widevine
iOS	Swift	AVPlayer	HLS	FairPlay Streaming

04. ゲーム音楽再生体験を支える技術

「ながさチェンジ」機能

ゲーム中のように途切れることなくずっと聴いていたい



- 求められる機能
 - 継ぎ目なく再生し続けられる仕組み
 - お客様が再生時間を設定できること

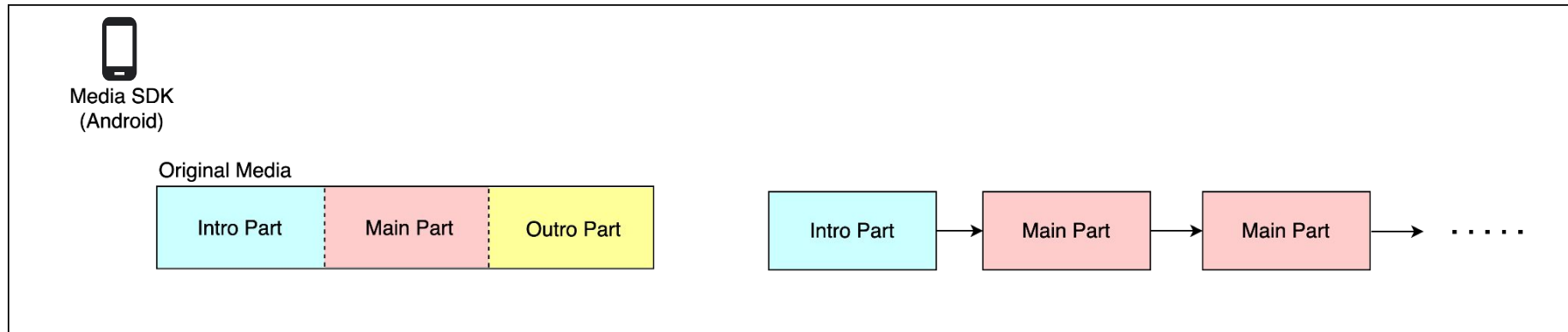
OS ごとに異なる再生方式で「ながさチェンジ」を実現



04. ゲーム音楽再生体験を支える技術

「ながさチェンジ」機能 - Android

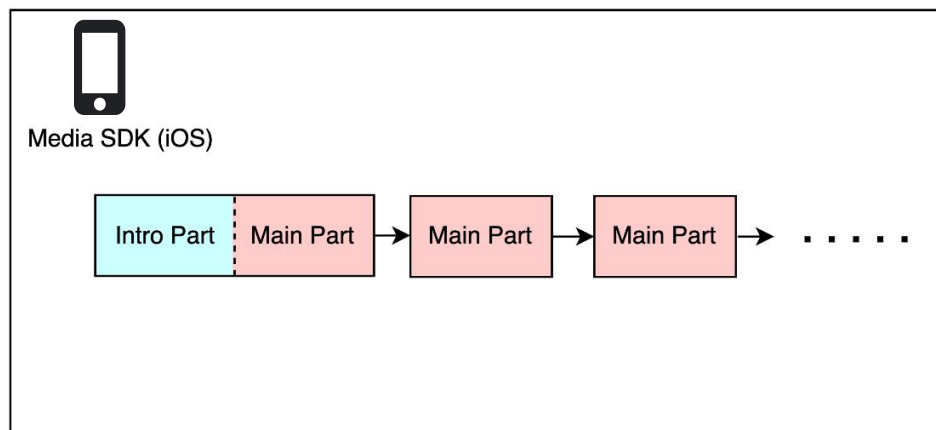
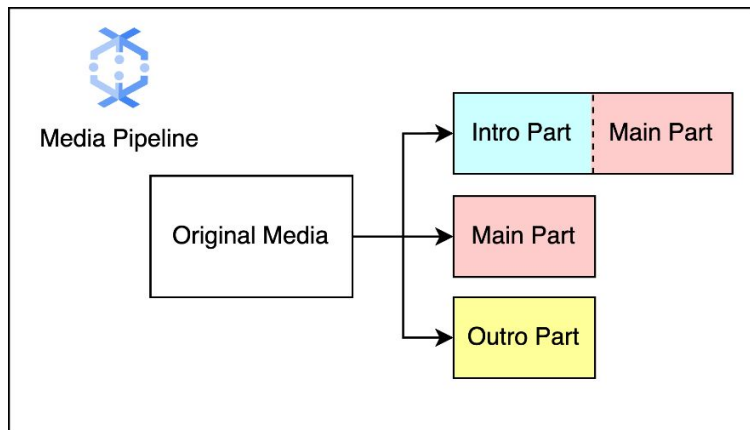
- ClippingMediaSource を利用して曲を 3 つに論理分割 (Intro, Main, Outro)
- ConcatenatingMediaSource2 を利用してループするように繋ぎ合わせる



04. ゲーム音楽再生体験を支える技術

「ながさチェンジ」機能 - iOS

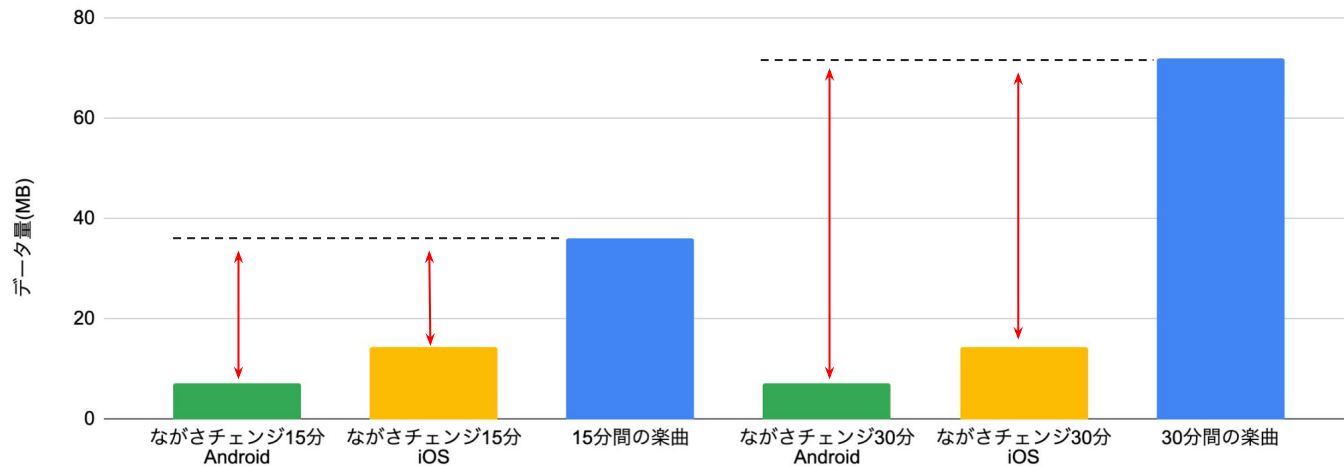
- Media Pipeline で Intro+Main, Main, Outro パートごとの楽曲を作成
- AVQueuePlayer でループするようにパート繋ぎ合わせ再生し続ける
- Intro が短い場合はノイズが発生するため Intro + Main を利用



04. ゲーム音楽再生体験を支える技術

「ながさチェンジ」機能

- 長時間の再生でも一定のデータ量で再生可能
- Media SDK でのロジックの隠蔽によりアプリ開発が効率化



ながさチェンジ再生時の通信量比較

04. ゲーム音楽再生体験を支える技術

再生を待たせない仕組み

さまざまな年代のゲーム音楽を気軽に聞いてもらいたい
さまざまなプレイリストから新しいゲーム音楽を発見してもらいたい



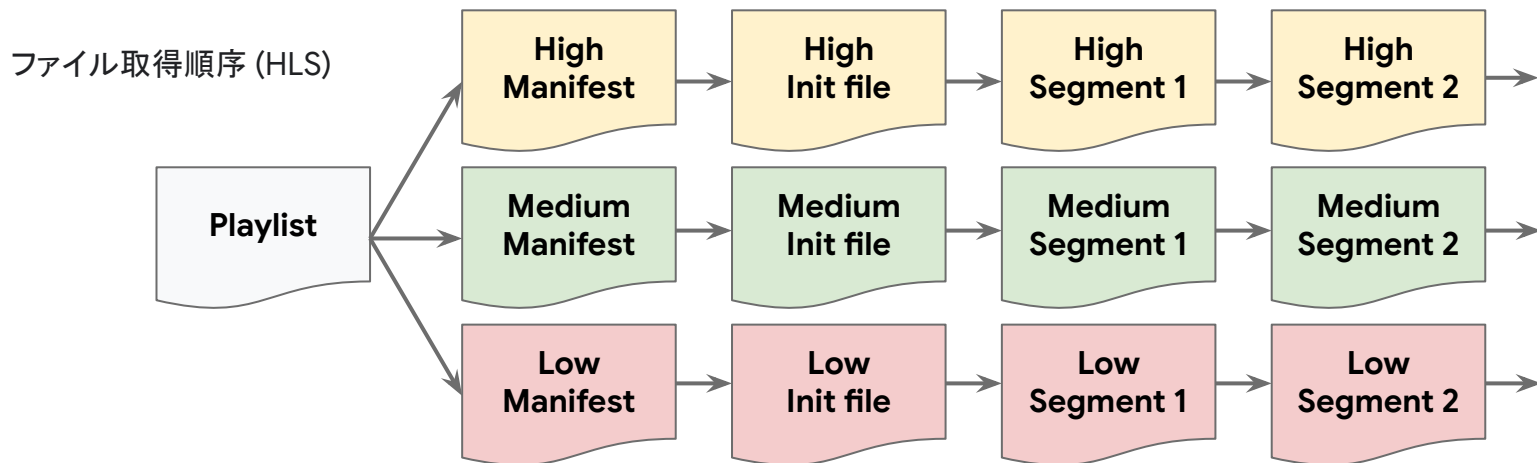
再生を待たせないことが重要

- Media Server のキャッシュとプリフェッチ
- Media SDK のプリフェッチ
- データ復号処理の改善

04. ゲーム音楽再生体験を支える技術

Media Server のキャッシュとプリフェッチ

- CDN + on memory cache による高速化 (API 側の方式と同じ)
- お客様が取得するファイルを**事前に CDN に配置**

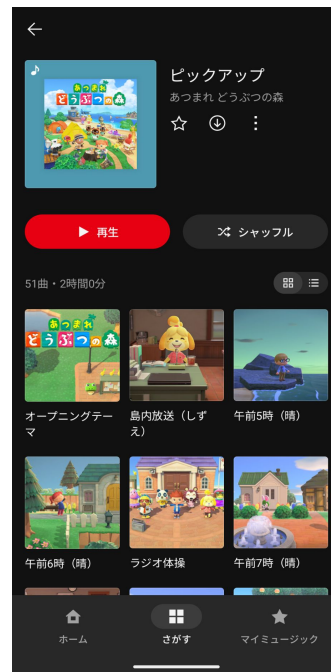


04. ゲーム音楽再生体験を支える技術

Media SDK のプリフェッチ

再生に必要な準備をお客様の**再生操作より前**に行う仕組み

- プリフェッチで行うこと
 - プレイヤー インスタンスの準備
 - ライセンスの準備
 - メディアファイルの準備
- プリフェッチ対象になる楽曲
 - 画面内に表示されている楽曲
 - キューに入っている楽曲



04. ゲーム音楽再生体験を支える技術

データ復号処理の改善 - Clear Lead

暗号化データはライセンス取得と復号処理により再生開始が遅くなる課題がある

- 楽曲冒頭の暗号化方式を変更し再生開始を高速化 (Clear Lead)
 - Android: **楽曲冒頭のみ暗号化なし**
 - iOS: **楽曲冒頭のみ共通鍵で暗号化**

0秒目から Key が必要

Clear lead なし	segment 1 (key A)	segment 2 (key A)	segment 3 (key A)	...
---------------	----------------------	----------------------	----------------------	-----

再生開始時は Key は不要 (復号も不要)

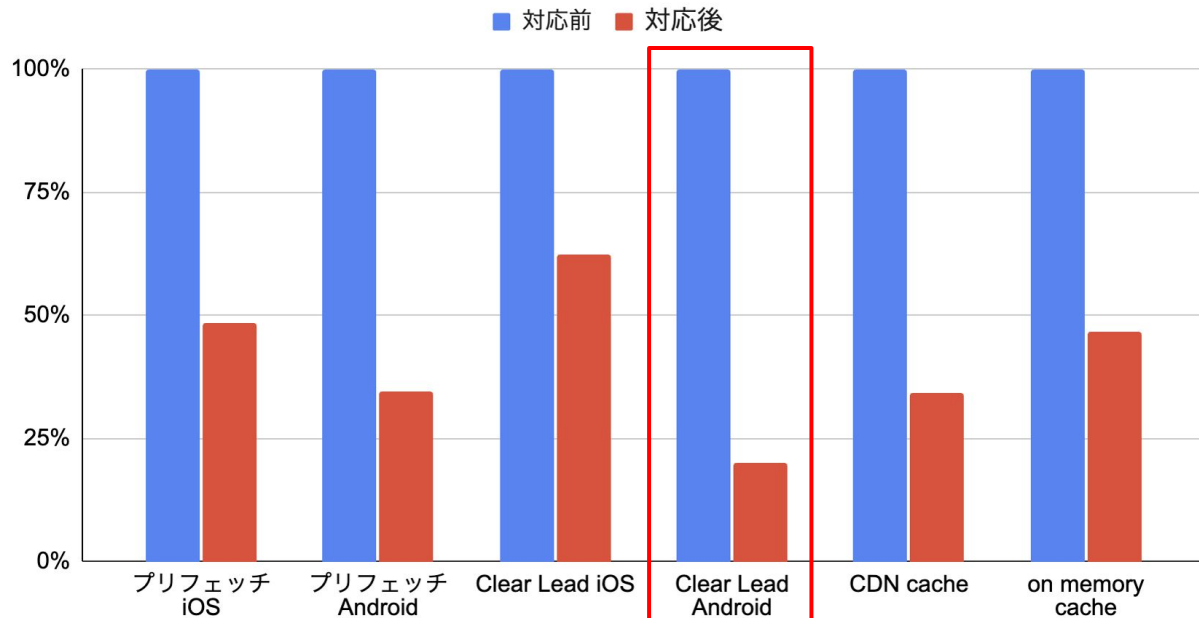
Android Clear lead	segment 1 (非暗号化)	segment 2 (key A)	segment 3 (key A)	...
--------------------	---------------------	----------------------	----------------------	-----

共通鍵取得済みの場合、再生開始時に Key の取得が不要

iOS Clear lead	segment 1 (共通鍵)	segment 2 (key A)	segment 3 (key A)	...
----------------	--------------------	----------------------	----------------------	-----

04. ゲーム音楽再生体験を支える技術 再生開始高速化の結果

総合して**約 3 倍**再生開始が高速化



再生開始高速化の結果

04. ゲーム音楽再生体験を支える技術

QoE データの活用

リリース後もお客様の再生体験を**継続的に改善**して行くための仕組み

- データ保持の要件や取得対象データのカスタマイズ性のために独自に取得
- BigQuery で収集・分析



04. ゲーム音楽再生体験を支える技術

QoE データの活用

- 主な指標
 - 再生開始までの時間(平均値、中央値)
 - Media SDK のエラー発生率
 - 再バッファリングにより再生が停止した時間と頻度
- 週次、アプリリリース、OS バージョンアップのタイミングで確認



お客様の再生体験を継続的にチェック

05. まとめ

05. まとめ

- **Cloud Run を中心としたサーバレスな環境でのシステム構築**
 - 完全にサーバレスな環境で運用負荷も低い状態で運営が可能に
 - CDN や memory cache も活用し、
高いパフォーマンスを発揮できることを確認
- **メディア配信における技術紹介**
 - Dataflow を用いたパイプライン処理の構築
 - メディアまわりを SDK 提供も含めて一貫して対応
 - ゲーム音楽ならではの体験提供、お客様を待たせない仕組みの構築



**ニンテンドーシステムズでは
一緒に働く仲間を募集中です！**