

JANOG58 ライトニングトーク

モバイル通信のUX改善 ～NWとAppの相互理解に向けて～



2026/07/16

NTTドコモビジネス イノベーションセンター 高木 雅

モバイル通信のUXとは？

単独指標

- スループット
- 遅延 (RTT)
- ジッター
- パケロス率
- TTFB: Time To First Byte

複合指標

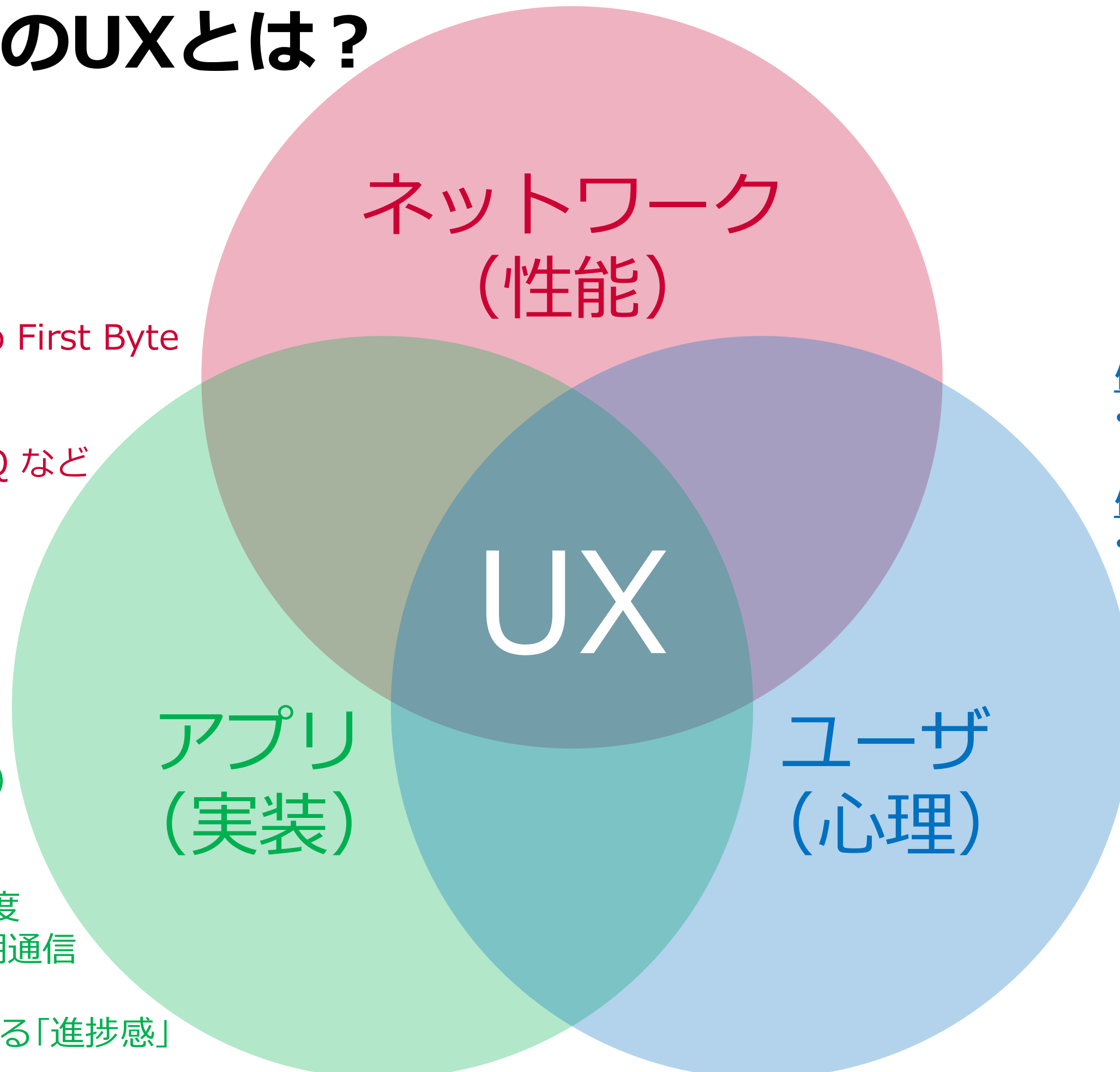
- OpenSignal CQ など

サーバ側

- APIの応答遅延
- APIの個数 (集約)
- FQDNの個数 (集約)

アプリ側

- API通信の回数と頻度
- Background/非同期通信
- オフラインモード
- アニメーションによる「進捗感」



例1

- レジの前でコード決済アプリがなかなか開かないと焦る

例2

- カーナビアプリが不調だとミッションクリティカル&運転中はトラブル対応が困難

NW分析：モバイル電測ツールの内製

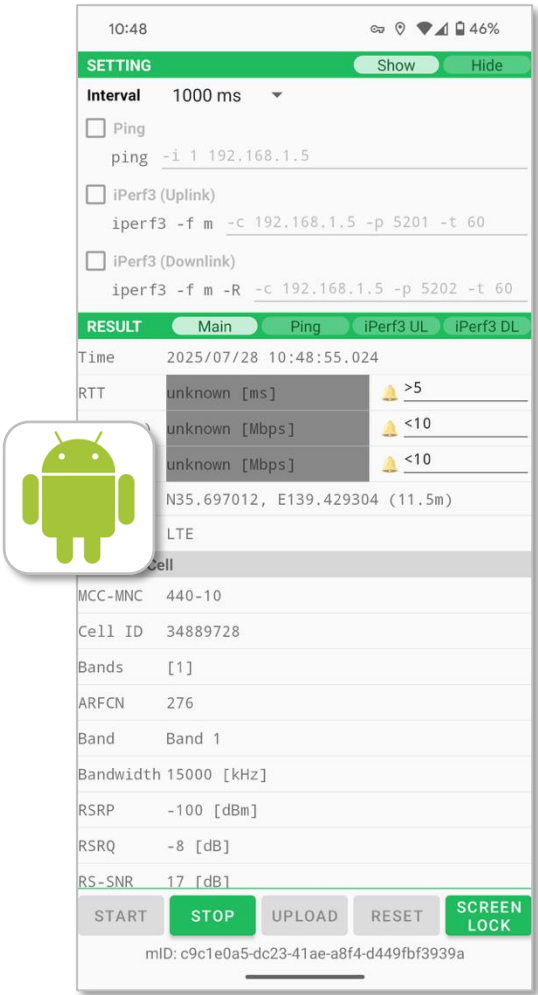
- 市販のAndroidスマホで約20種類（Galaxyは80種類）の指標を計測
- プッシュ通知により電測サービスを遠隔起動 / 停止 / Config投入
- 電測データは準リアルタイムでGoogleスプレッドシートに蓄積
- Googleマップで地図プロットして可視化
- 携帯4社を比較するとdocomo網の遅延が大きい傾向を確認

わずか数十msの違いが
UXに影響するものか？

可視化アプリ



Android 電測アプリ



計測指令・Config投入

バグ情報



Crashlytics
バグ情報収集

電測データ

電測データ



Apps Script
REST API



FCM
プッシュ通知



CloudRun
REST API

電測データ



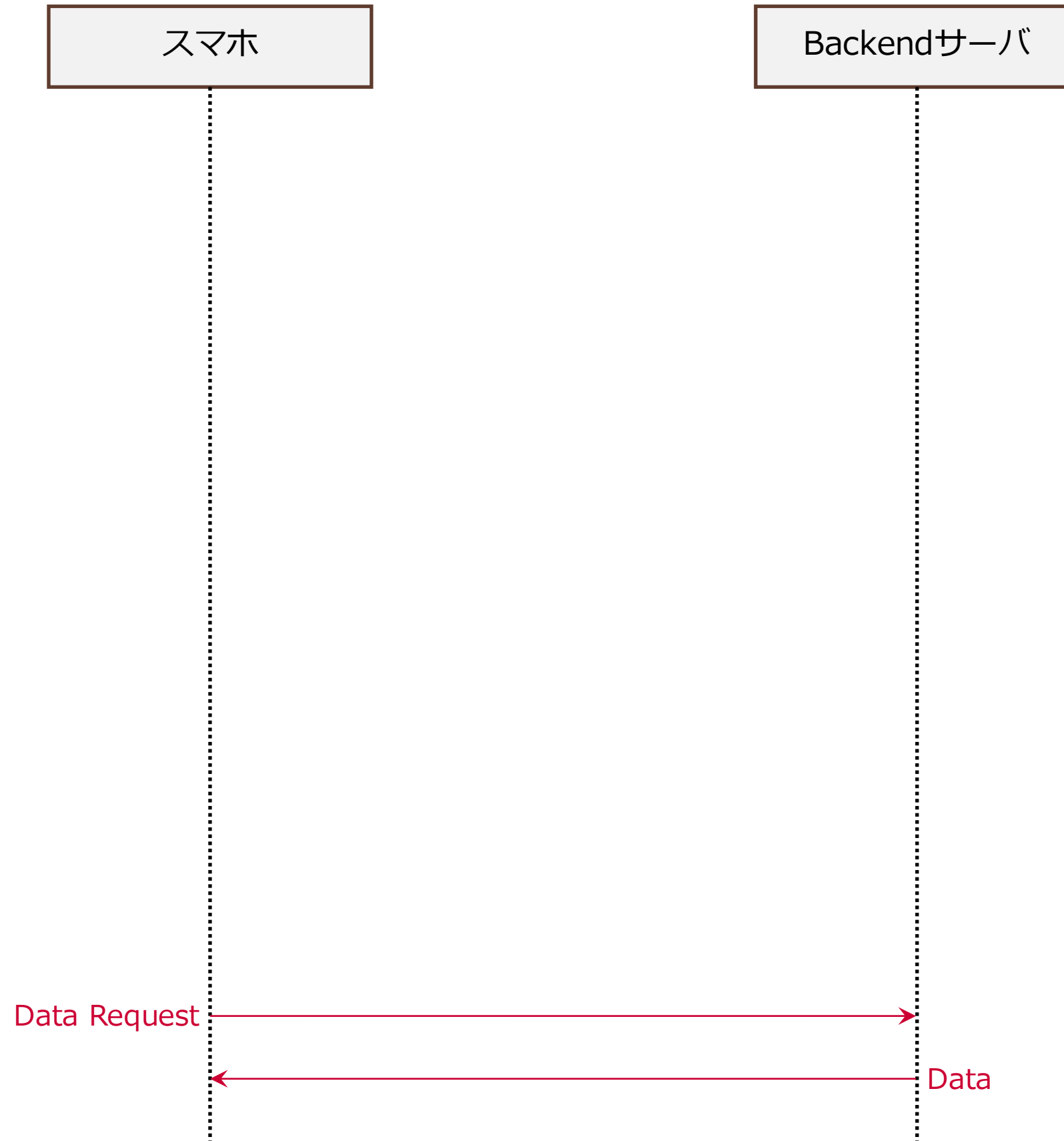
Apps Script
REST API

データ取得

Spreadsheet
データ蓄積・分析

App分析：通信遅延の考察 (1/5)

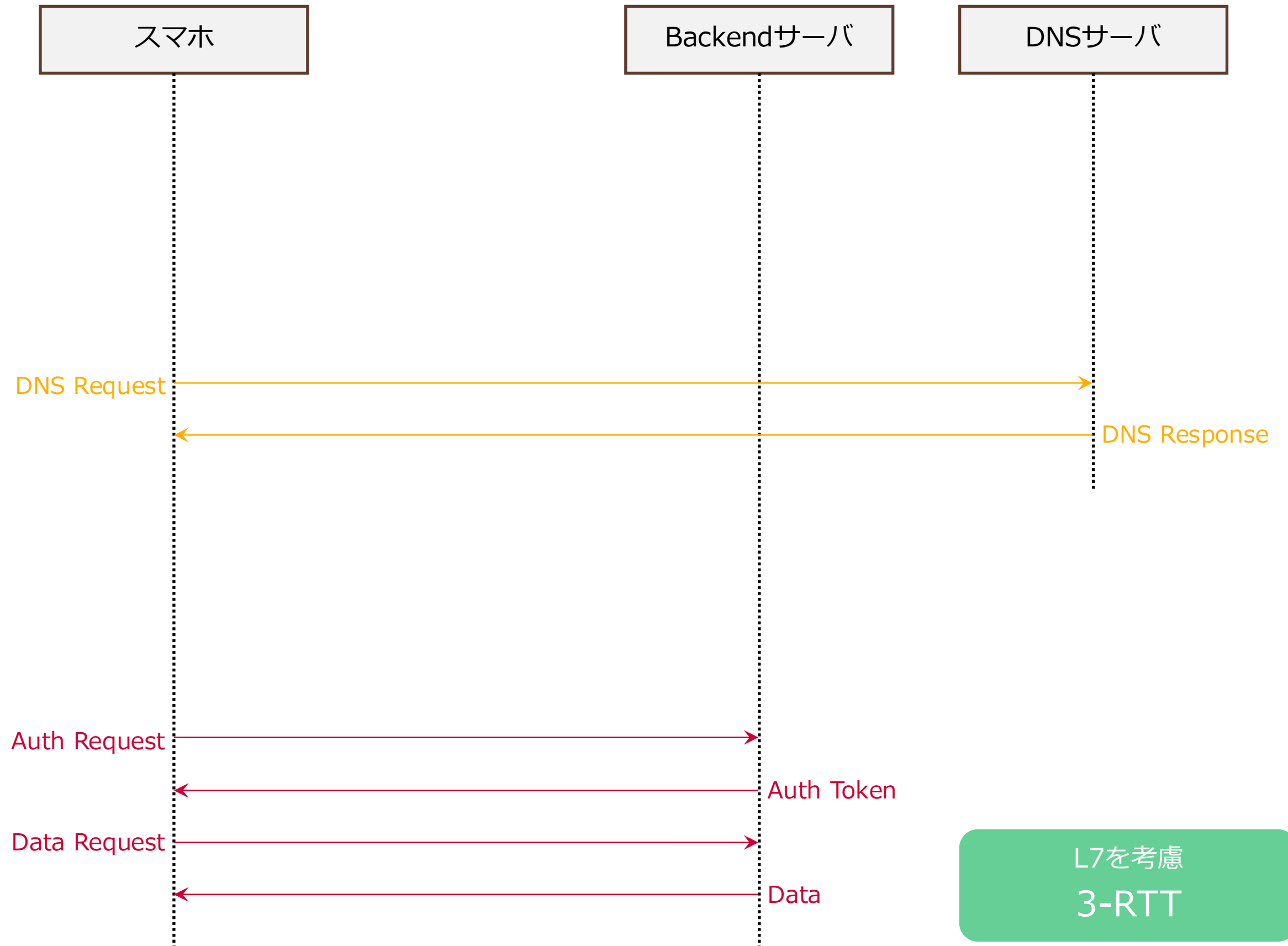
- アプリ受託開発の業務経験に基づき
アプリ起動直後のサーバ通信を考察
- DNS over TLS
 - Android 9 からデフォルト有効
 - iOS 14 からサポート
- TLSバージョン
 - 最新のTLS 1.3を仮定
 - TLS 1.2なら1-RTT増
 - QUICなら1-RTT減
- L7の認証処理 (Auth)
 - 認証APIでのToken発行 +
Tokenでの簡易認証が一般的
- CDNサーバの併用
 - FQDNもIPも異なるため、
DNS・TCP・TLSを再実行する必要
 - BackendサーバからDataが届くまで
アクセス先が分からない (直列処理)



ビジネスロジックのみ
1-RTT

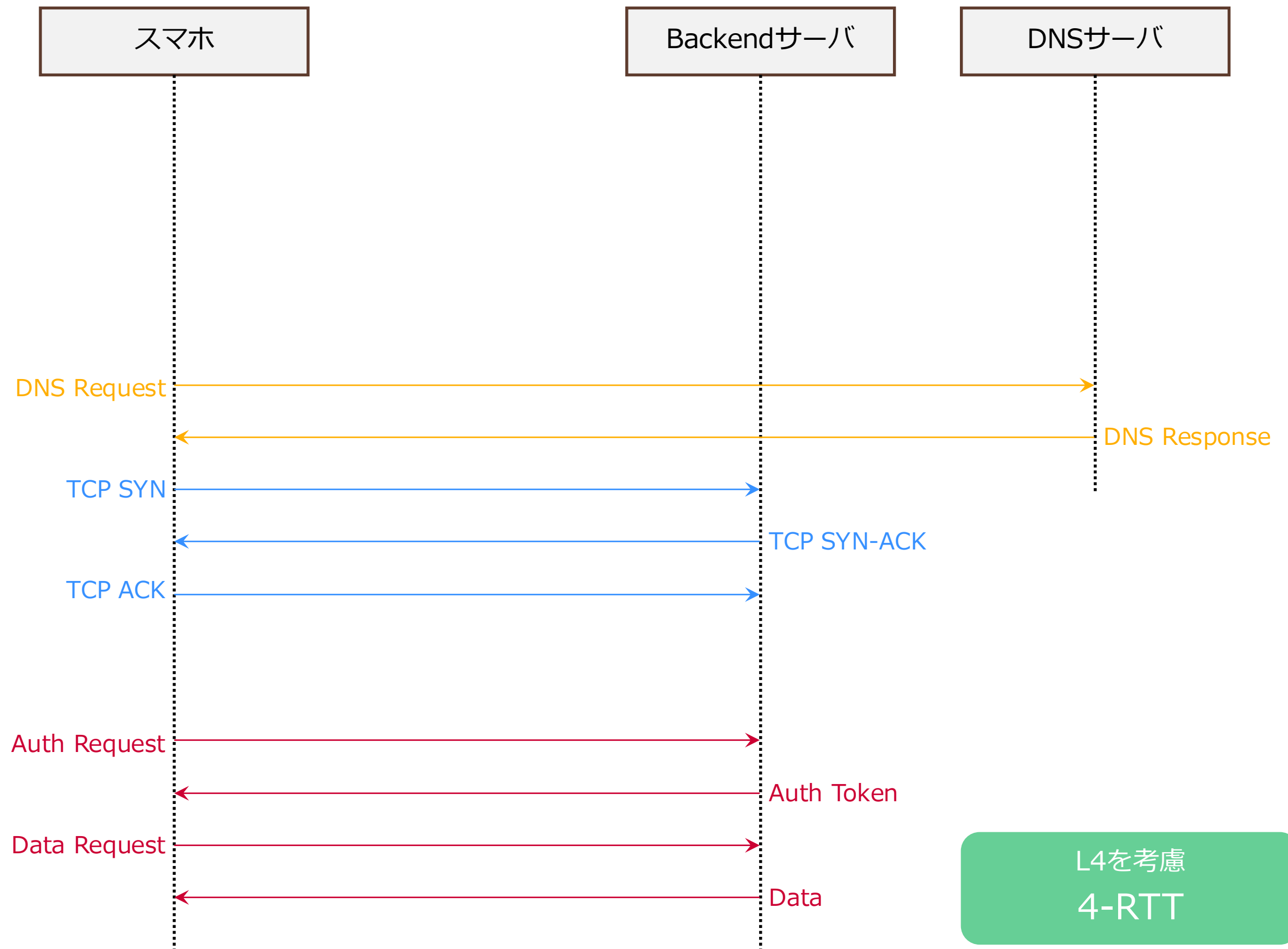
App分析：通信遅延の考察 (2/5)

- アプリ受託開発の業務経験に基づき
アプリ起動直後のサーバ通信を考察
- DNS over TLS
 - Android 9 からデフォルト有効
 - iOS 14 からサポート
- TLSバージョン
 - 最新のTLS 1.3を仮定
 - TLS 1.2なら1-RTT増
 - QUICなら1-RTT減
- L7の認証処理 (Auth)
 - 認証APIでのToken発行 +
Tokenでの簡易認証が一般的
- CDNサーバの併用
 - FQDNもIPも異なるため、
DNS・TCP・TLSを再実行する必要
 - BackendサーバからDataが届くまで
アクセス先が分からない (直列処理)



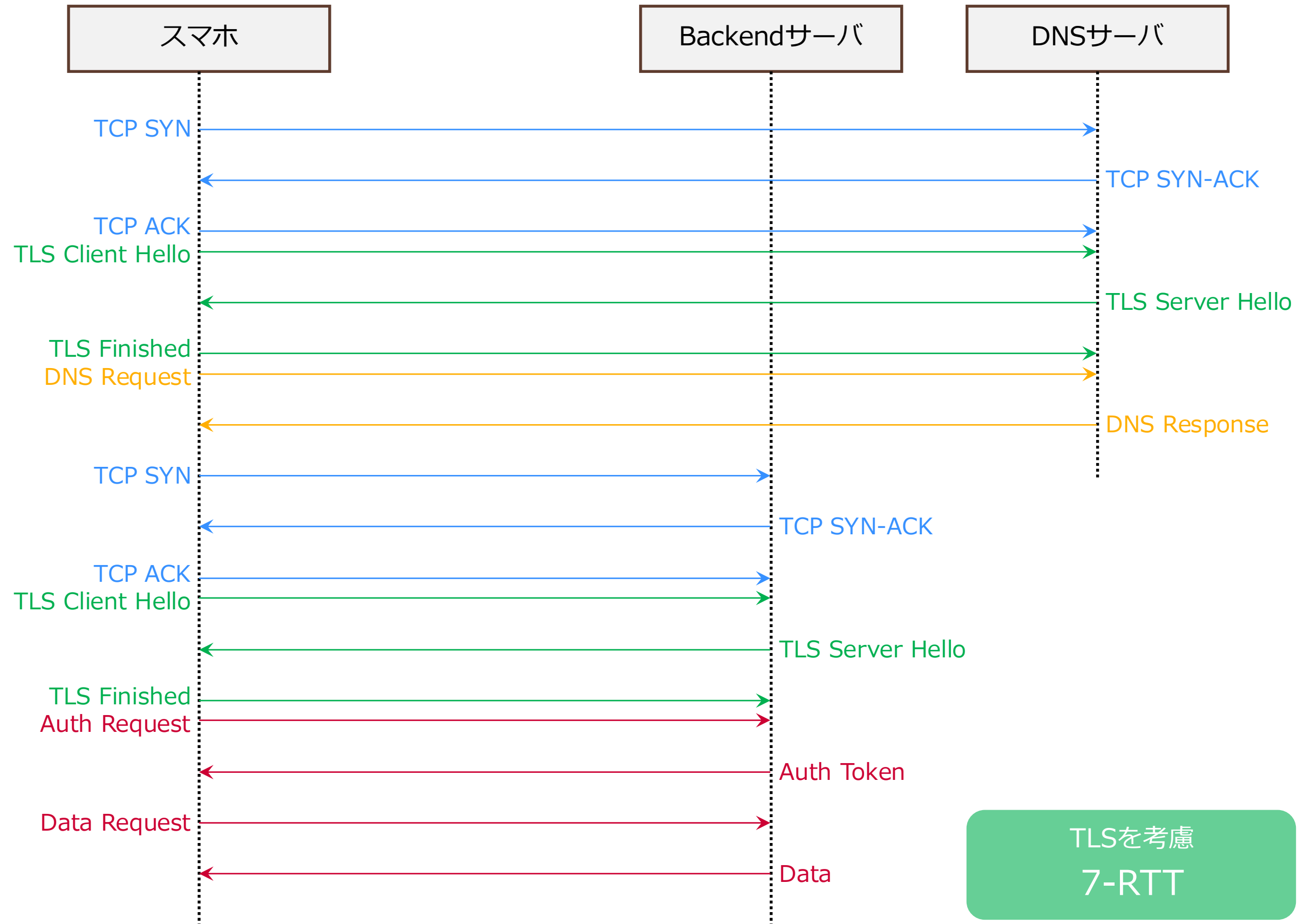
App分析：通信遅延の考察 (3/5)

- アプリ受託開発の業務経験に基づき
アプリ起動直後のサーバ通信を考察
- DNS over TLS
 - Android 9 からデフォルト有効
 - iOS 14 からサポート
- TLSバージョン
 - 最新のTLS 1.3を仮定
 - TLS 1.2なら1-RTT増
 - QUICなら1-RTT減
- L7の認証処理 (Auth)
 - 認証APIでのToken発行 +
Tokenでの簡易認証が一般的
- CDNサーバの併用
 - FQDNもIPも異なるため、
DNS・TCP・TLSを再実行する必要
 - BackendサーバからDataが届くまで
アクセス先が分からない (直列処理)



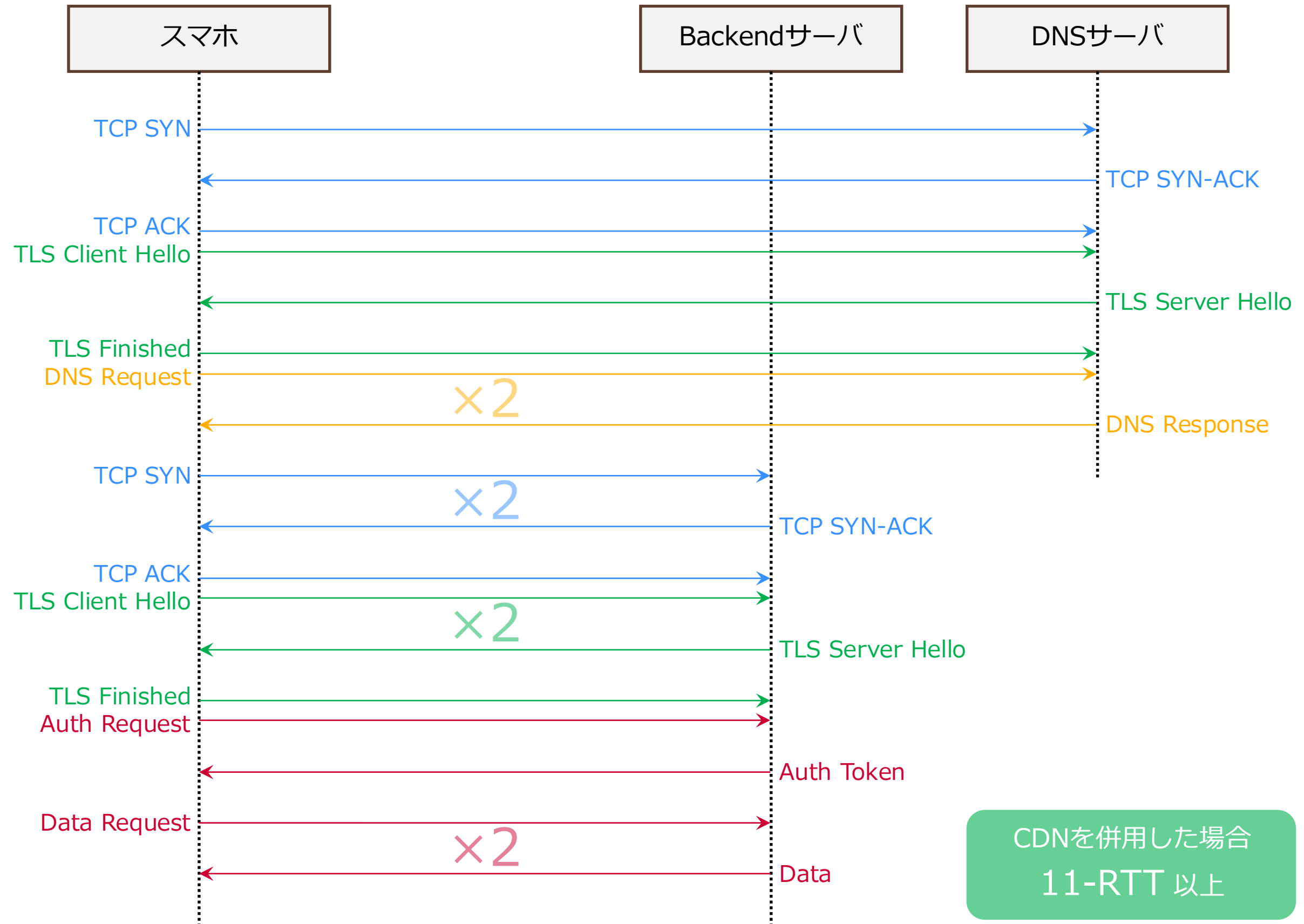
App分析：通信遅延の考察 (4/5)

- アプリ受託開発の業務経験に基づき
アプリ起動直後のサーバ通信を考察
- DNS over TLS
 - Android 9 からデフォルト有効
 - iOS 14 からサポート
- TLSバージョン
 - 最新のTLS 1.3を仮定
 - TLS 1.2なら1-RTT増
 - QUICなら1-RTT減
- L7の認証処理 (Auth)
 - 認証APIでのToken発行 +
Tokenでの簡易認証が一般的
- CDNサーバの併用
 - FQDNもIPも異なるため、
DNS・TCP・TLSを再実行する必要
 - BackendサーバからDataが届くまで
アクセス先が分からない (直列処理)

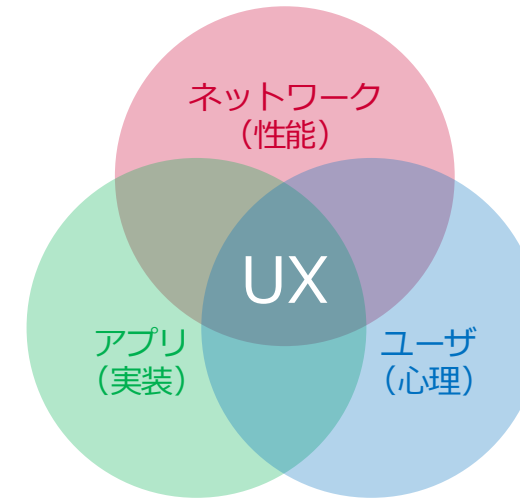


App分析：通信遅延の考察 (5/5)

- アプリ受託開発の業務経験に基づき
アプリ起動直後のサーバ通信を考察
- DNS over TLS
 - Android 9 からデフォルト有効
 - iOS 14 からサポート
- TLSバージョン
 - 最新のTLS 1.3を仮定
 - TLS 1.2なら1-RTT増
 - QUICなら1-RTT減
- L7の認証処理 (Auth)
 - 認証APIでのToken発行 +
Tokenでの簡易認証が一般的
- CDNサーバの併用
 - FQDNもIPも異なるため、
DNS・TCP・TLSを再実行する必要
 - BackendサーバからDataが届くまで
アクセス先が分からない (直列処理)



NWとAppの相互理解に向けて



- UX（ユーザ体験）は複合的に決まる

- ネットワーク性能 × アプリ実装 × ユーザ心理
- 通信事業者とアプリ開発者が手を取り合うべき

- 通信事業者の課題と対策

- 世の中「お行儀の良い」アプリばかりではない
 - そういう通信も円滑に流せないでNWの評判に影響
 - 他社アプリの仕様には直接手を出せないが…
→ アプリ開発者向けのイベント等で啓発しよう！
→ NWフレンドリーな通信ライブラリを制作・配布？
- アプリ開発に精通した人が少ない（8年間の勤務経験から）
 - 自社アプリでも内製せず、外注するケースが多い
 - 発注側に知識がないと、適切な指示を出せない
→ Android/iOSアプリ開発者を社内で育成しよう！
→ 自社アプリを内製してノウハウを蓄積しよう！

- アプリ開発者の課題と対策

- NWに精通した人が少ない（7年間の勤務経験から）
 - 「通信すると待たされる」「たまに失敗する」くらいの認識
 - APIコール1回で何往復の通信が走るか？など考えもしない
→ モバイルNWを勉強してライバルに差をつけよう！
- ユーザが劣悪な通信環境にいるケースは珍しくない
 - 今月の「ギガ」を使い切った
 - 混雑した時間帯/エリアにいる
- 劣悪な通信環境での動作テストは珍しい
 - アプリ開発中のテスト環境は社内Wi-Fiが基本
 - 開発中のAPIサーバに携帯網から接続できないケースも
→ リアルな通信環境でアプリの動作テストをしよう！