aws DEV DAY
ONLINE JAPAN

DEV DAY

20-22.10.2020

In Partnership with intel

aws DEV DAY
ONLINE JAPAN

C - 7

AWS AppSync Advanced Design Pattern

Atsuya Nunomura

Senior Solutions Architect
Amazon Web Services Japan K.K.

In Partnership with intel

Who am I ?

布村 純也 (Atsuya, Nunomura)

- **Role :**

  Senior Solutions Architect, Enterprise

- **Background**

  Web/Mobile Apps、API・Web Service 開発、Apollo/GraphQL

- **好きな AWS のサービス :**

  AWS Amplify　　AWS AppSync　　AWS Lambda



Agenda

1. GraphQL & AWS AppSync のおさらい
2. AWS AppSync アーキテクチャ
3. AppSync ユースケース/デザインパターン
4. まとめ

本日お話する事

想定する参加者

・AWS AppSync を既に利用されていてる
・AWS AppSync の基本的な用語や機能を知っている

お話する内容

・AWS AppSync のユースケースに応じてどのように利用するか
・AWS AppSync におけるデザインパターン

※ 時間の都合上、主要なユースケース/デザインパターンのみ

aws DEV DAY
ONLINE JAPAN

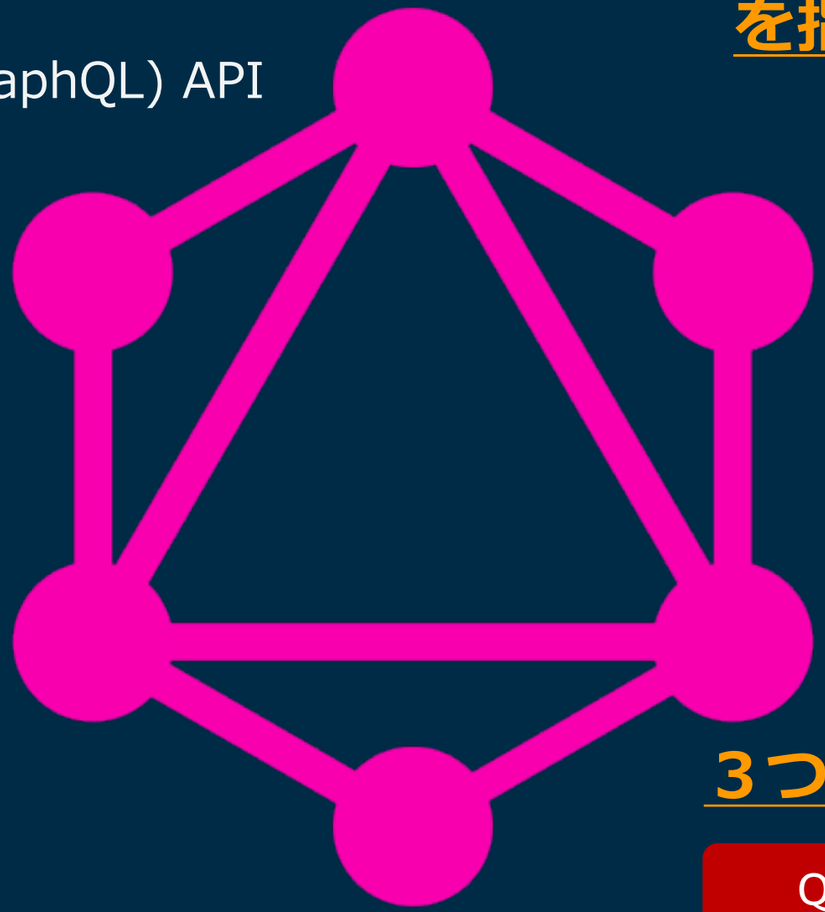
1. GraphQL & AWS AppSync のおさらい

In Partnership with intel

GraphQL とは！？

OSS QUERY 言語

A query language for your (GraphQL) API



型指定されたスキーマ

```
type Query {
    getTodos: [Todo]
}

type Todo {
    id: ID!
    name: String!
    description: String
    priority: Int
    duedate: String
}
```

クライアントがレスポンス形式を指定

```
type Query {
    getTodos: [Todo]
}

type Todo {
    id: ID!
    name: String!
    description: String
    priority: Int
    duedate: String
}
```

```
query {
  getTodos {
    id
    name
    priority
  }
}
```

３つの処理形態

Query | Mutation | Subscription

https://graphql.org/

In Partnership with intel

GraphQL定義

```
type Event {
    id: ID!
    name: String
    where: String
    when: String
    description: String
    comments: [Comment]
}

type Comment {
    commentId: String!
    eventId: ID!
    content: String!
    createdAt: String!
}
```

In Partnership with intel

GraphQL定義

Query : 取得

Schema

```
type Query {
    getEvent (id: ID!): Event
    listEvents (
        limit: Int,
        nextToken: String
    ): EventConnection
}
```

GraphQL定義

Mutation : 変更

Query : 取得

Schema

```
type Mutation {
    createEvent (
        name: String!,
        when: String!,
        where: String!,
        description: String!
    ): Event
    deleteEvent (id: ID!): Event
    commentOnEvent (
        eventId: ID!,
        content: String!,
        createdAt: String!
    ): Comment
}
```


In Partnership with intel

GraphQL定義

Subscription : 購入

Mutation : 変更

Query : 取得

Schema

```
type Subscription {
    onComments (eventId: String!): Comment
    @aws_subscribe(mutations:
["commentOnEvent"])
}
```

```
type Mutation {
commentOnEvent (
        eventId: ID!,
        content: String!,
        createdAt: String!
    ): Comment
}
```


In Partnership with intel

AWS AppSync

GraphQL
マネージド・サービス

アカウントの
Data sourceに接続

データ同期、リアルタイム、
オフライン機能

GraphQL ファサード

競合の検出と解決

セキュリティ
(API Key, IAM, Cognito, OIDC)

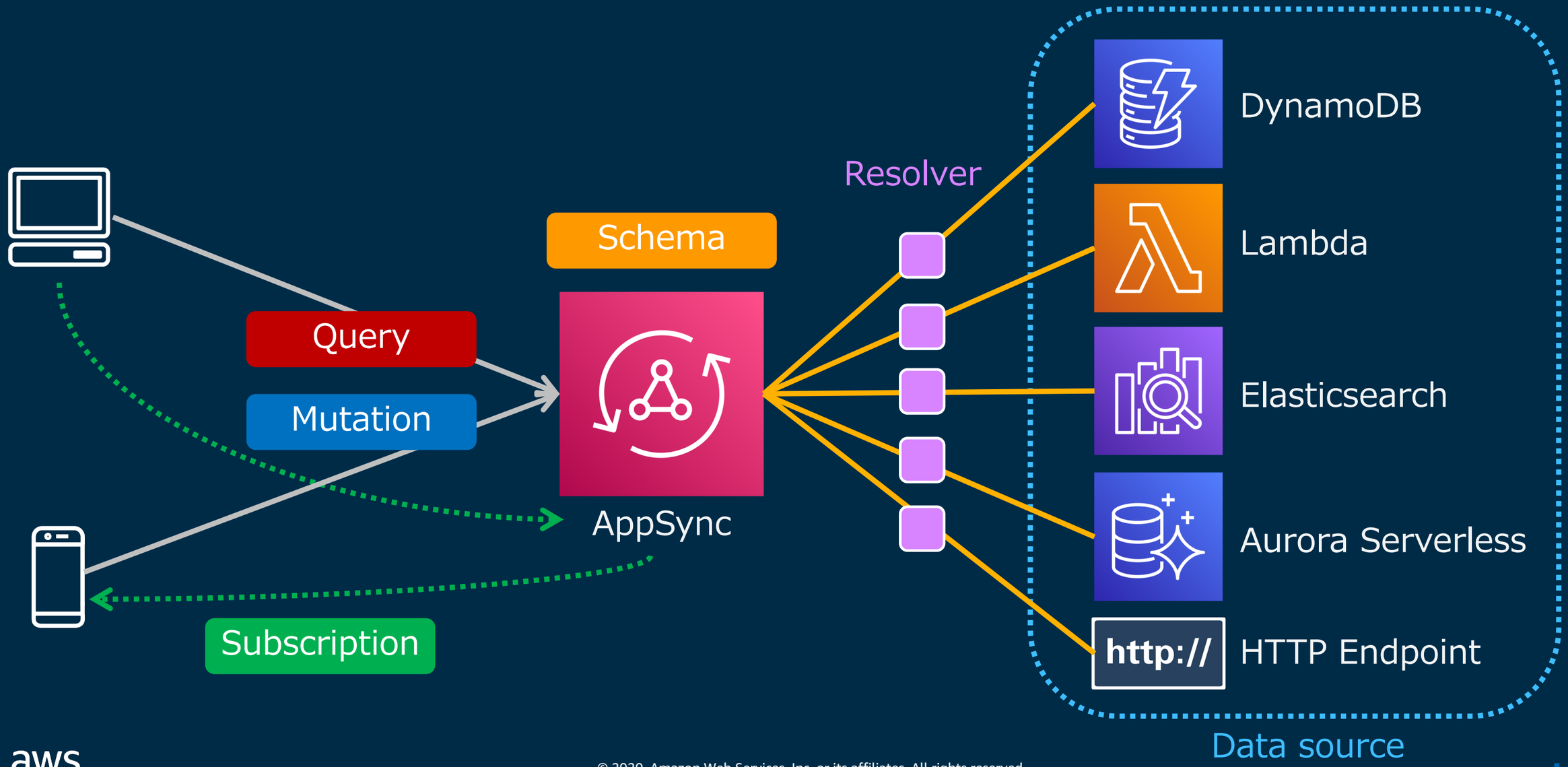
In Partnership with intel

aws DEV DAY
ONLINE JAPAN

2. AWS AppSync アーキテクチャ

In Partnership with intel

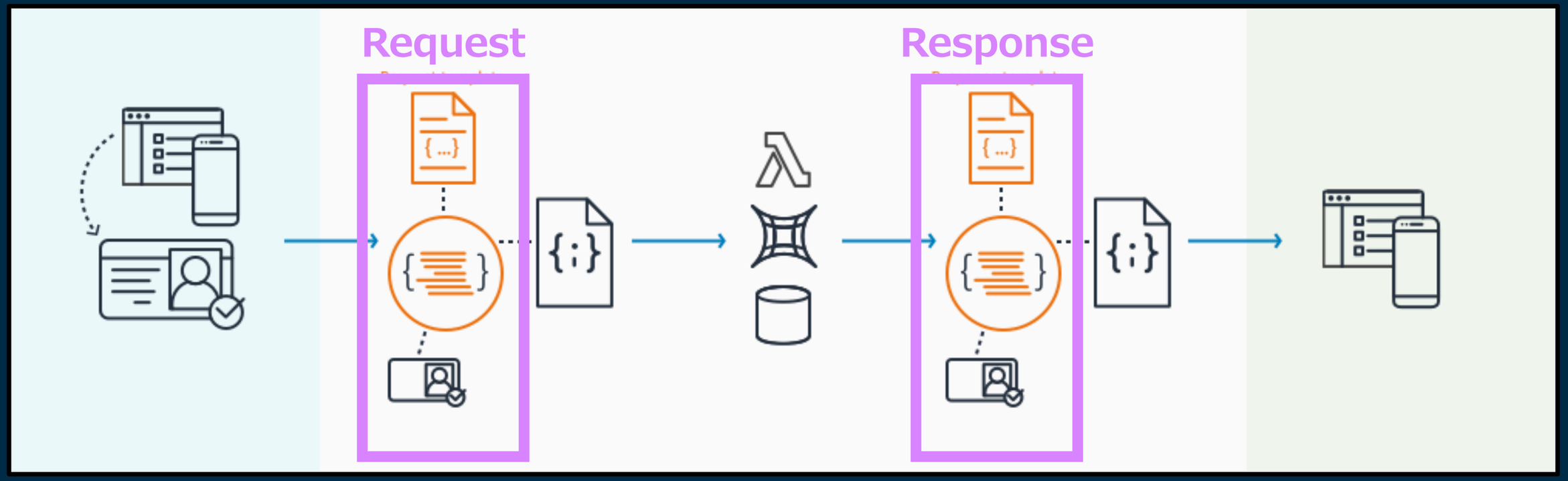
AWS AppSync 基本アーキテクチャ



In Partnership with intel

Resolver：2種類のMapping Template

- Request テンプレート（GraphQL Request → Data source命令）
- Responseテンプレート（Data source 応答 → GraphQL Response）



In Partnership with intel

Resolver：Request Mapping Template

必要となる処理ロジックを VTL (Velocity Template Language) で記載

```
#set( $friendIds = [] )
#set( $friendships = $ctx.prev.result.items )
#foreach( $tmp in $friendships )
    $util.qr( $friendIds.add( $tmp.friendId ) )
#end
{
   "version" : "2018-05-29",
   "operation" : "Scan",
   "filter" : {
    "expression": "contains( :flds, userId ) AND begins_with( registtime, :trgtMo )",
      "expressionValues" : {
         ":flds" : $util.dynamodb.toDynamoDBJson( $friendIds ),
```

Resolver：Response Mapping Template

Data source 応答後、クライアントへ返却する処理ロジックを記載

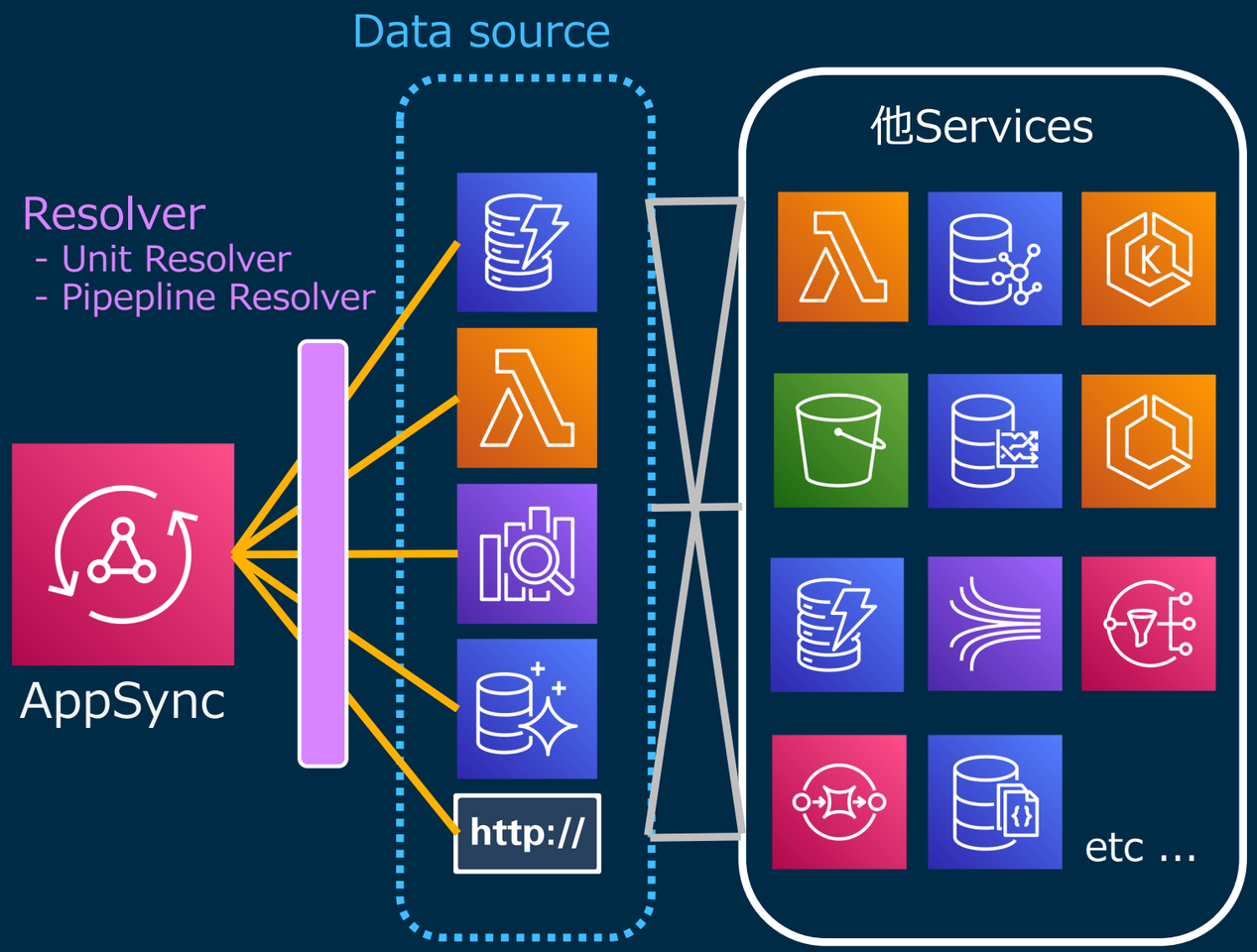
- デフォルトで返す場合：

```
$utils.toJson ( $context.result )
```

- データを結合する場合：

```
{
    "id" : ${context.data.id},
    "title" : "${context.data.theTitle}",
    "content" : "${context.data.body1} ${context.data.body2}"
}
```

AWS AppSync アーキテクチャ



AppSync を BFF としたアーキテクチャの場合
Resolver、Data source、Backend services の組み合わせは開発者の自由

開発における選択肢が多用で拡張性が非常に高い反面、Workload に沿ったアーキテクチャデザインが重要となる

In Partnership with intel

aws DEV DAY
ONLINE JAPAN

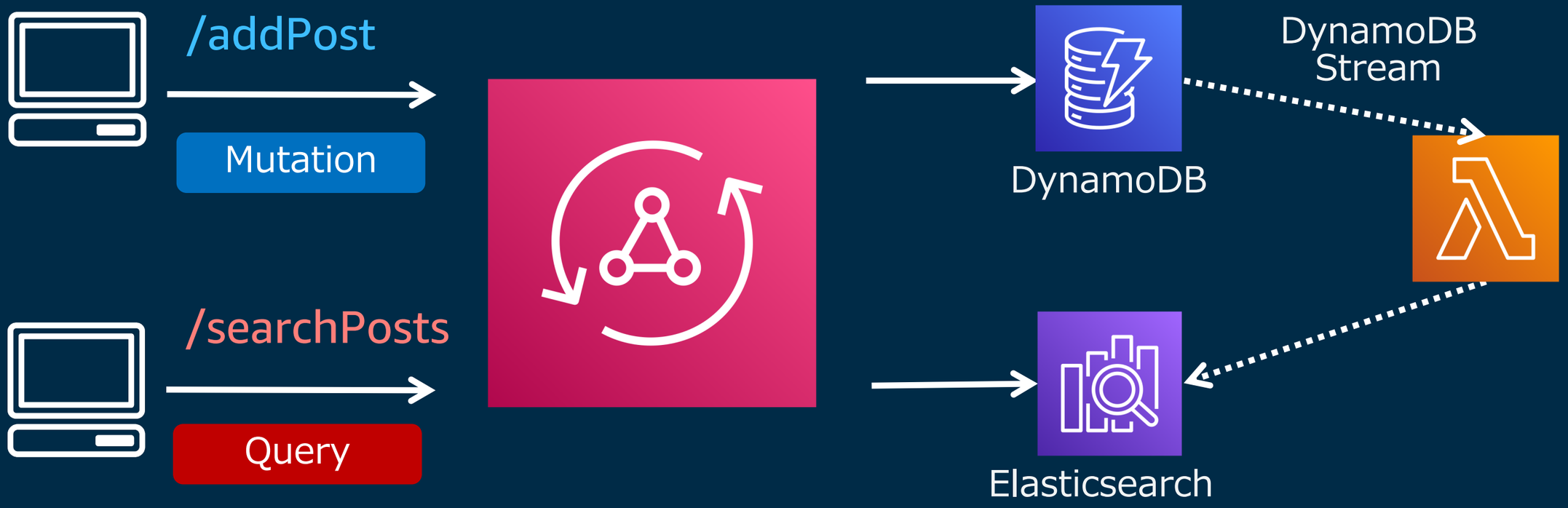
3. AppSync ユースケース/デザインパターン

In Partnership with intel

Multi-Data source / CQRS

AppSync の Data source として DynamoDB を利用する事は鉄板。 しかし
UI側でユーザ操作によるソート切替え、キーワード検索、ファジー検索機能等が
必要なWorkloadの場合、どの様にアーキテクチャを組むか。

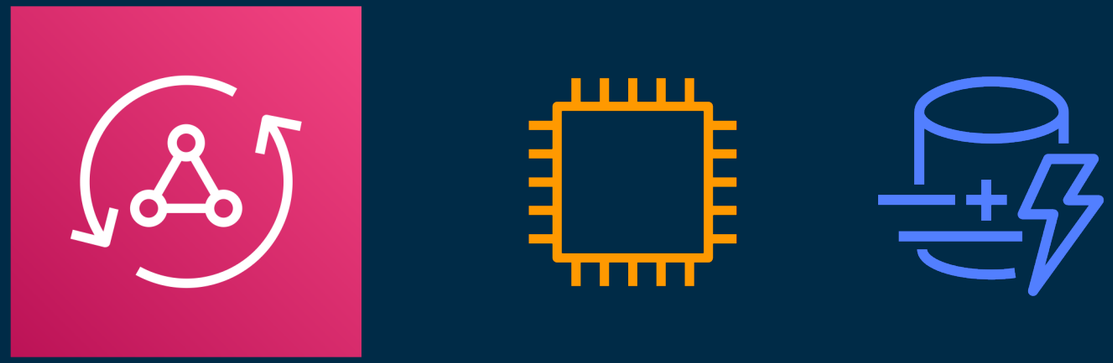
CQRS の実現



Data source を複数組み合わせ、それぞれの特性を活かした処理を実行する事で
高度な検索(キーワード検索、ファジー検索、地理空間検索)やソート切替えにも対応可能。

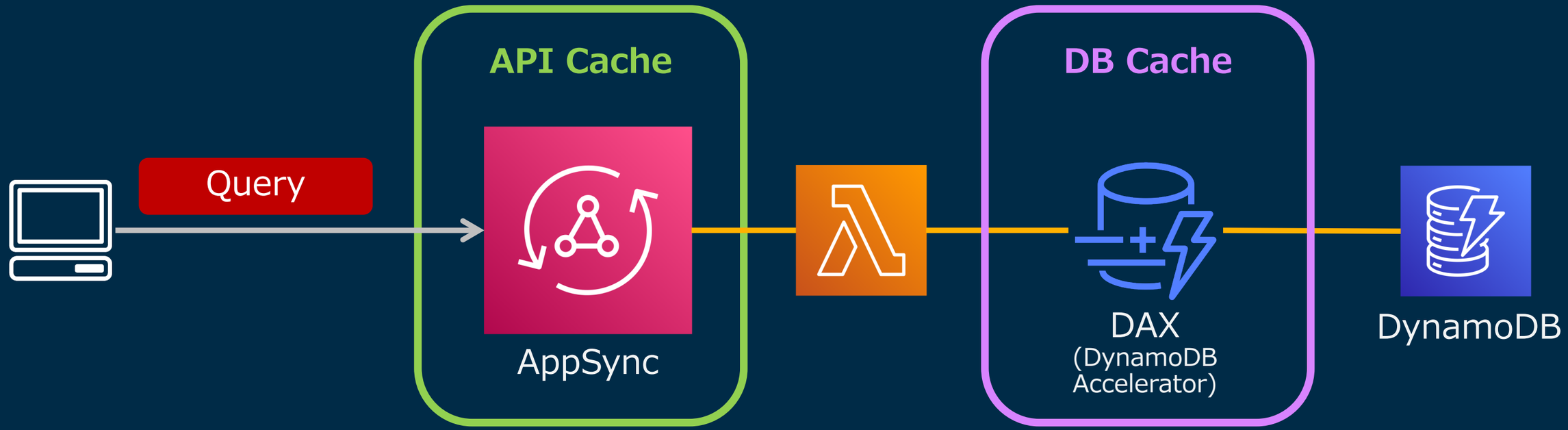
Cache Strategy

GraphQL API として Query が多い Workload で AppSync の Caching をどの様に実現するか、どの Layer でCache を実現するか？



Cache Layer 例

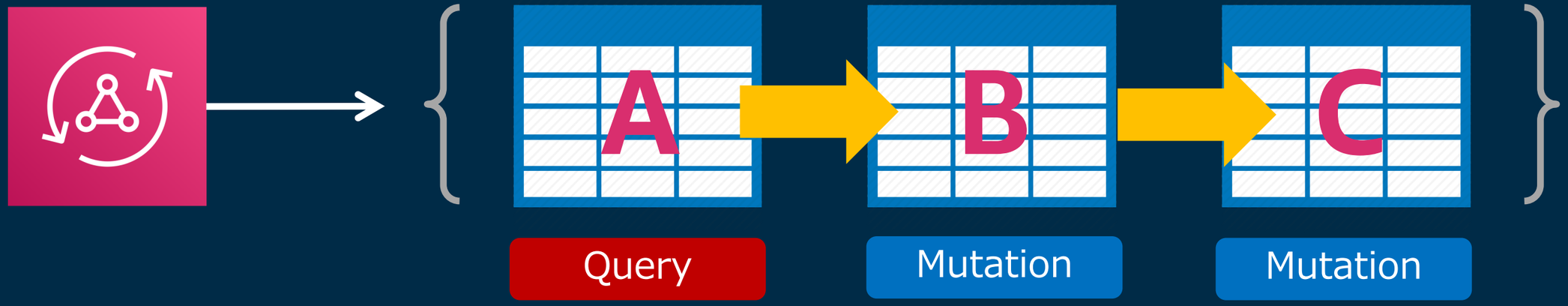
○ GraphQL API ベースで Caching 管理が可能
○ 全ての Data source に対応可能
○ 対象として Full Request、Resolver 単位での Caching 指定が可能
△ シンプルな Cache アーキテクチャで DAX ほど多機能でない



○ DAX の Full 機能が利用可能
○ 大規模 Workloadでの CU (Capacity Unit) の抑制によるコスト低減
○ Multi-AZ Node による Primary 昇格
△ Item, Query 別の Cache, Negative Cache, Write-Around による Cache 未反映等を意識した処理実装が必要

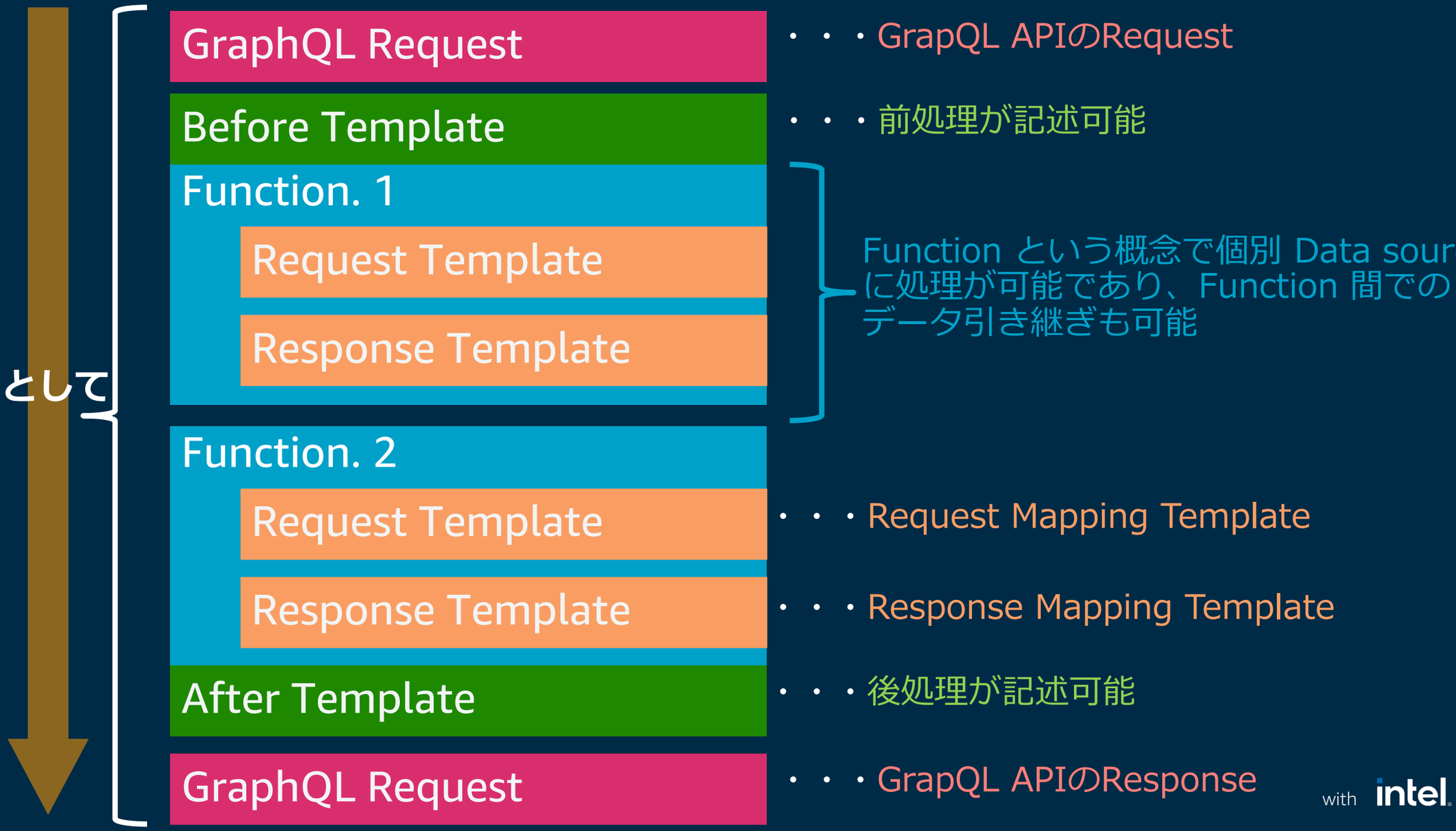
Multi-Operation / 複数処理 実行

AppSync の Resolver 処理として連続した複数処理が必要、又は前処理の結果により後続の処理が分岐・変更する一連の制御が必要な場合



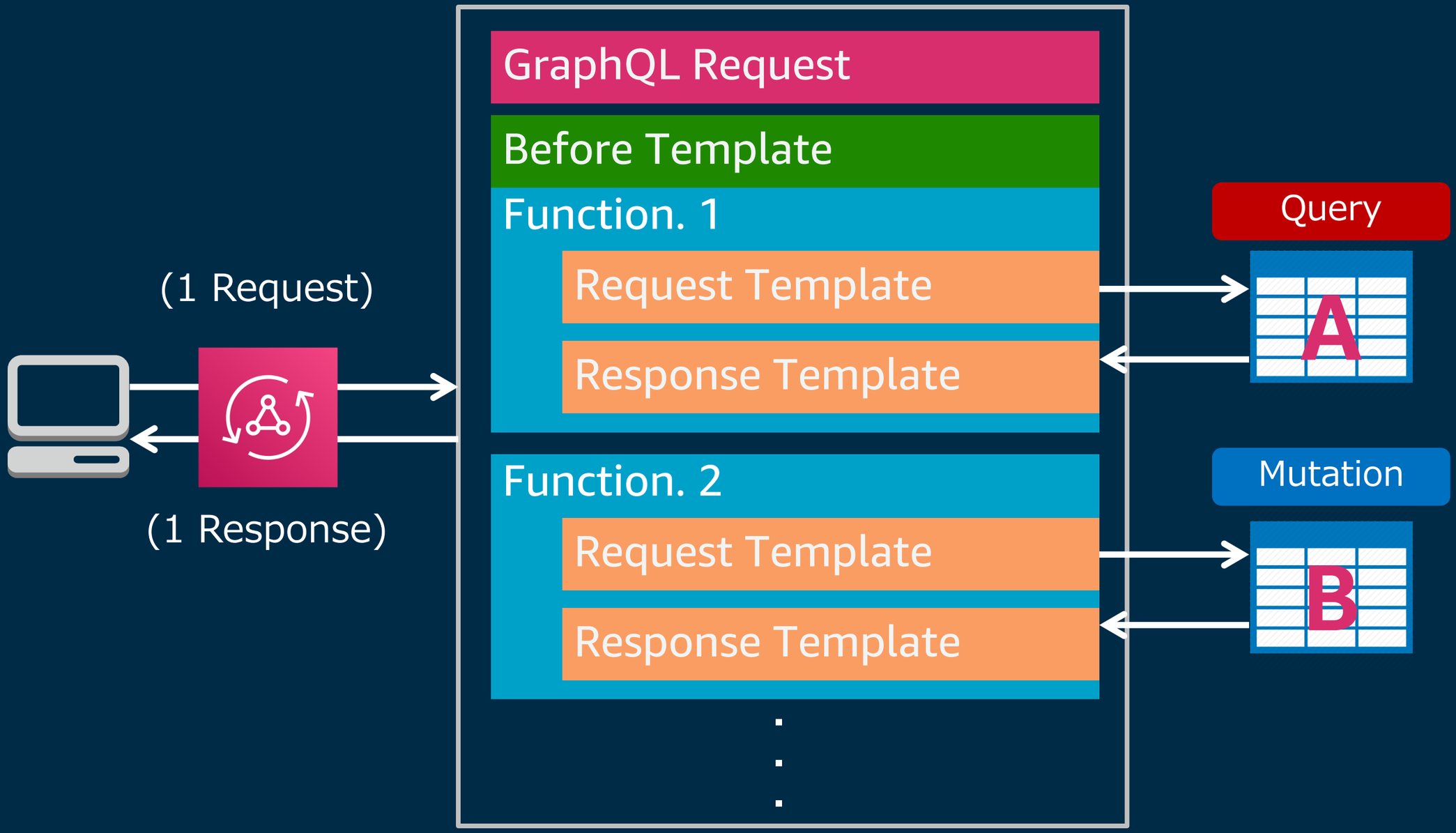
In Partnership with intel

Pipeline Resolver



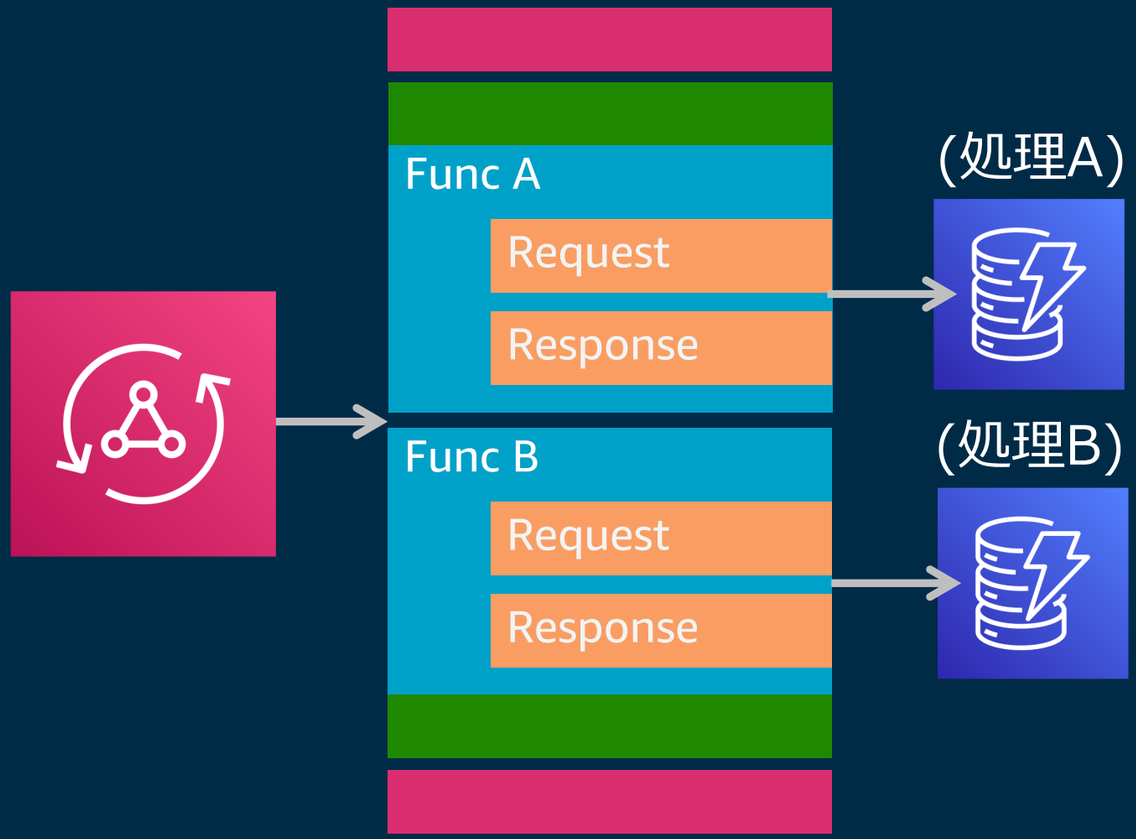
with intel

Pipeline Resolver



[Deep Dive] Pipeline Resolver と DynamoDB Stream の違いは？

Pipeline Resolver



DynamoDB Stream

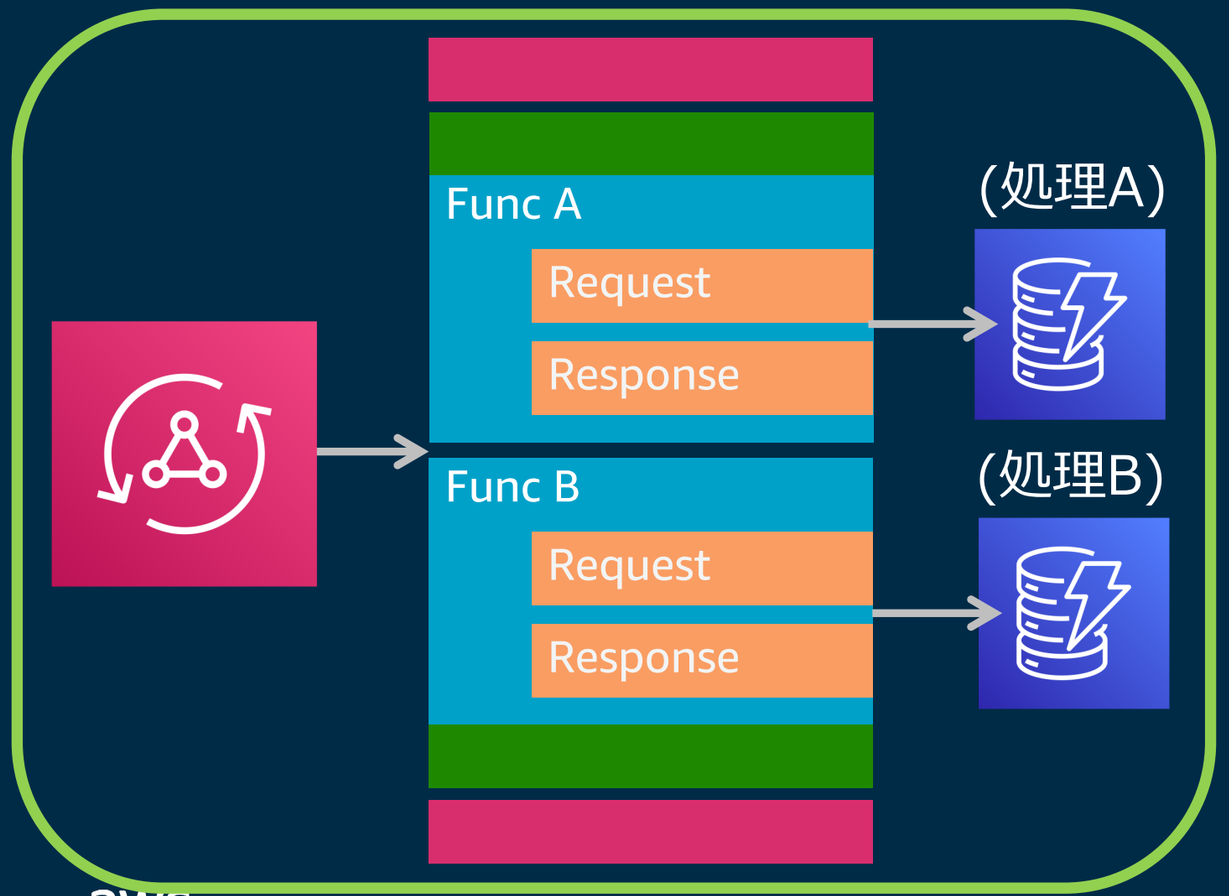
(Unit Resolver：DynamoDB)

非同期処理となるが処理Aの結果
で処理Bを変える事も可能

In Partnership with intel

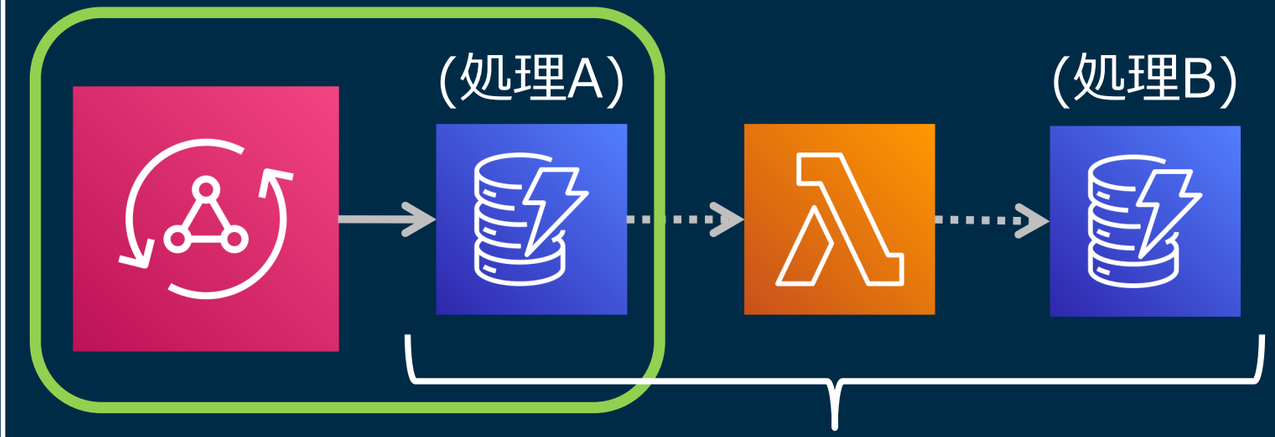
AppSyncの管理範囲の違い（= X-Ray 管理範囲の違い）

Pipeline Resolver



DynamoDB Stream

(Unit Resolver：DynamoDB)

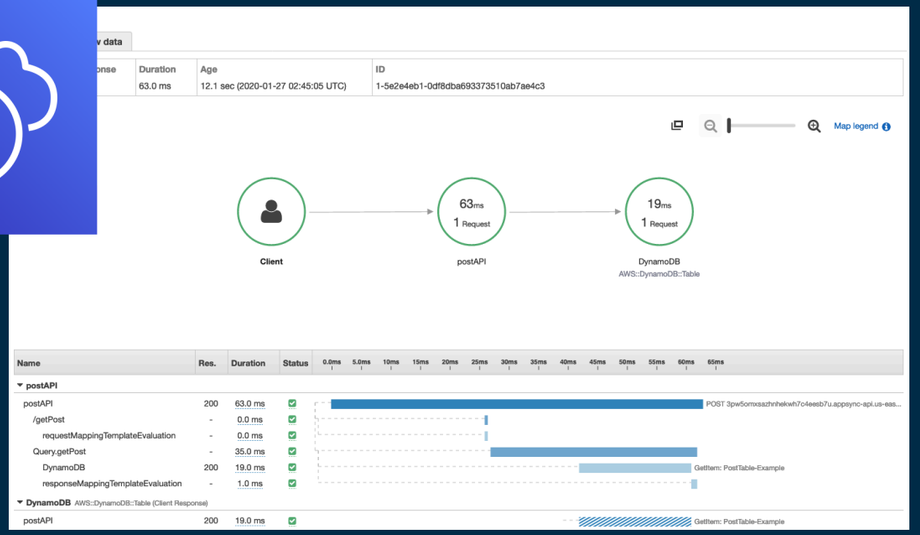


非同期処理となるが処理Aの結果
で処理Bを変える事も可能

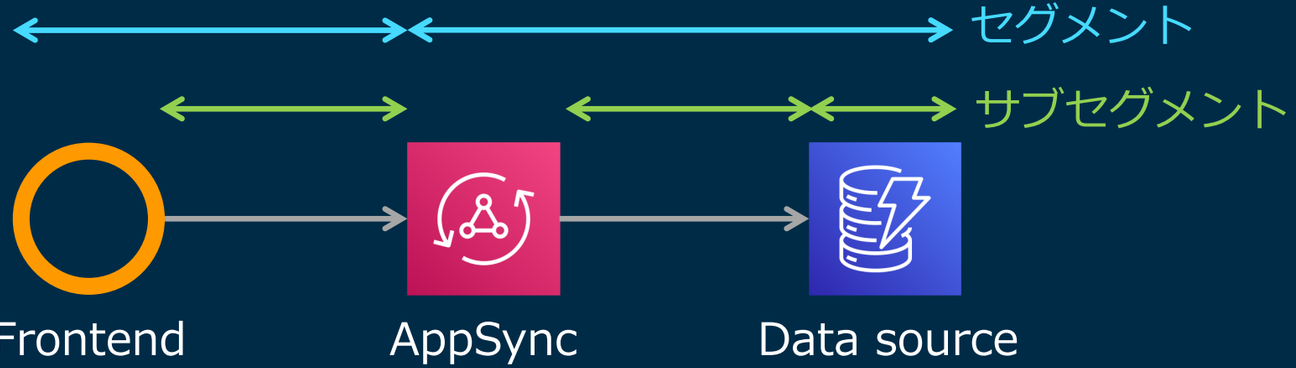
In Partnership with intel

X-Ray による GraphQL API のトレース

AppSync の API 設定で X-Ray を有効化



- Resolver 単位にセグメント、サブセグメント情報を記録
- AppSync の API Cache Hit も記録



In Partnership with intel

GraphQL Subscription Filtering - 1

対象者を絞って Subscription を受信する必要がある Workload。
例えば SNS 機能で同一部門で、且つ同一プロジェクトのメンバーのユーザのみに自身の投稿内容を通知させる機能。

Subscription Filtering：Subscription引数

Subscription 引数を使用し実現可能

Subscription 先の Mutation 処理の返す型に含まれるフィールドを
フィルタリングする事で実現

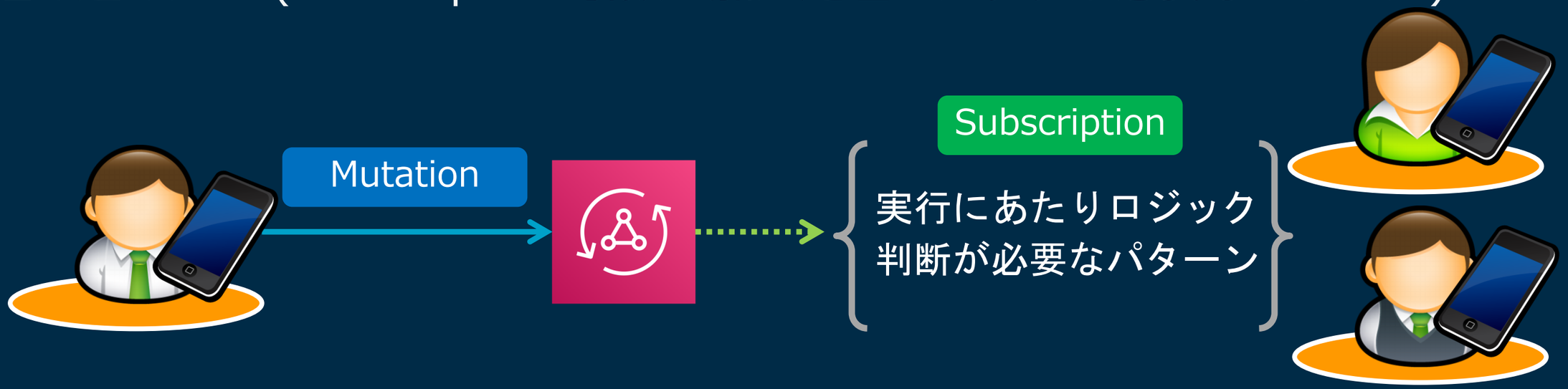
```
type Mutation {
    createPost( input: CreatePostInput! ): Project
}
type Subscription {
    onCreatePost( prjId: String ): Project
    @aws_subscribe( mutations: ["createPost"] )
}
```

※ Mutation 実行時 Selection Set (応答)に prjId を含めておく必要がある

GraphQL Subscription Filtering - 2

対象者を絞って Subscription を受信する必要がある Workload。

Query、Mutation と同様に Subscription でも接続認可するきめ細かな処理を組み込みたい (Subscription 引数に加え処理ロジックが必要なパターン)



Subscription Filtering：Subscription引数 + α

Subscription 引数 + Resolver(Data source:None) をアタッチ

SubscriptionにData source:None の空 Resolver をアタッチし、
Request/Response Mapping Template にて判定・アクセス制御を行う
事で実現可能

例) Cognito ユーザがグループ “Group1” に属していない場合、Subscription が拒否される動作

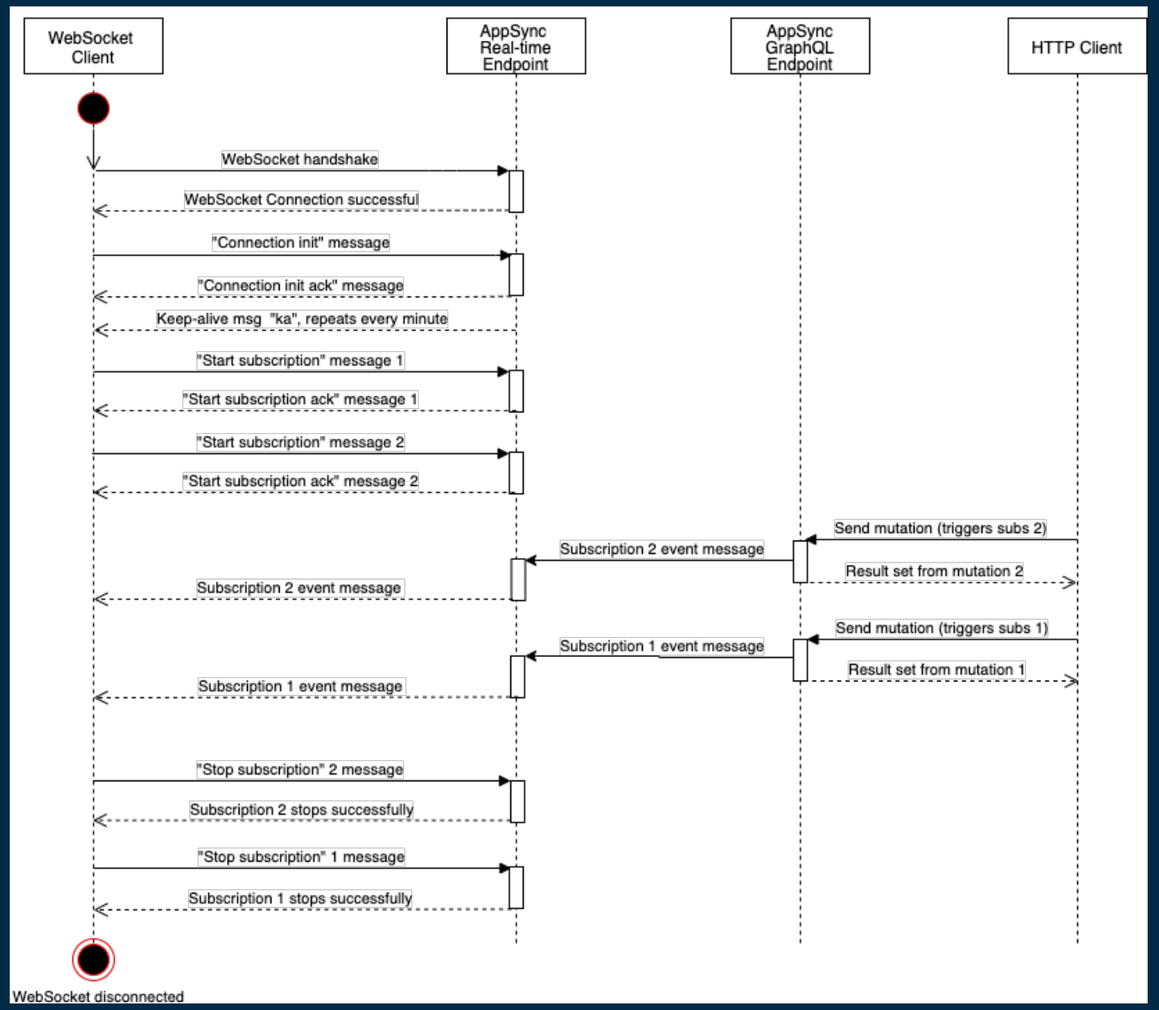
```
#if ( !$context.identity.claims['cognito:groups'].contains("Group1") )
    $util.unauthorized()
#end
```

Real-time WebSocket Client

AppSync SDK、Amplify Library が用意されているが、IoT デバイス OS等の理由で開発言語が SDK、Library に対応していない Workloadの場合

(任意の開発言語での AppSync 利用)

Real-time WebSocket Client



https://docs.aws.amazon.com/appsync/latest/devguide/real-time-websocket-client.html

AppSync におけるQuery、Mutation は HTTPS で処理が可能

Query　Mutation

```
https://
 xxx.appsync-api.[region].amazonaws.com/graphql
```

Subscription については MQTT から Pure WebSocket に変更後、Handshake の Workflow仕様を公開しており任意の開発言語での実装が可能

Subscription

```
wss://
 xxx.appsync-realtime-api.[region].amazonaws.com/graphql
```

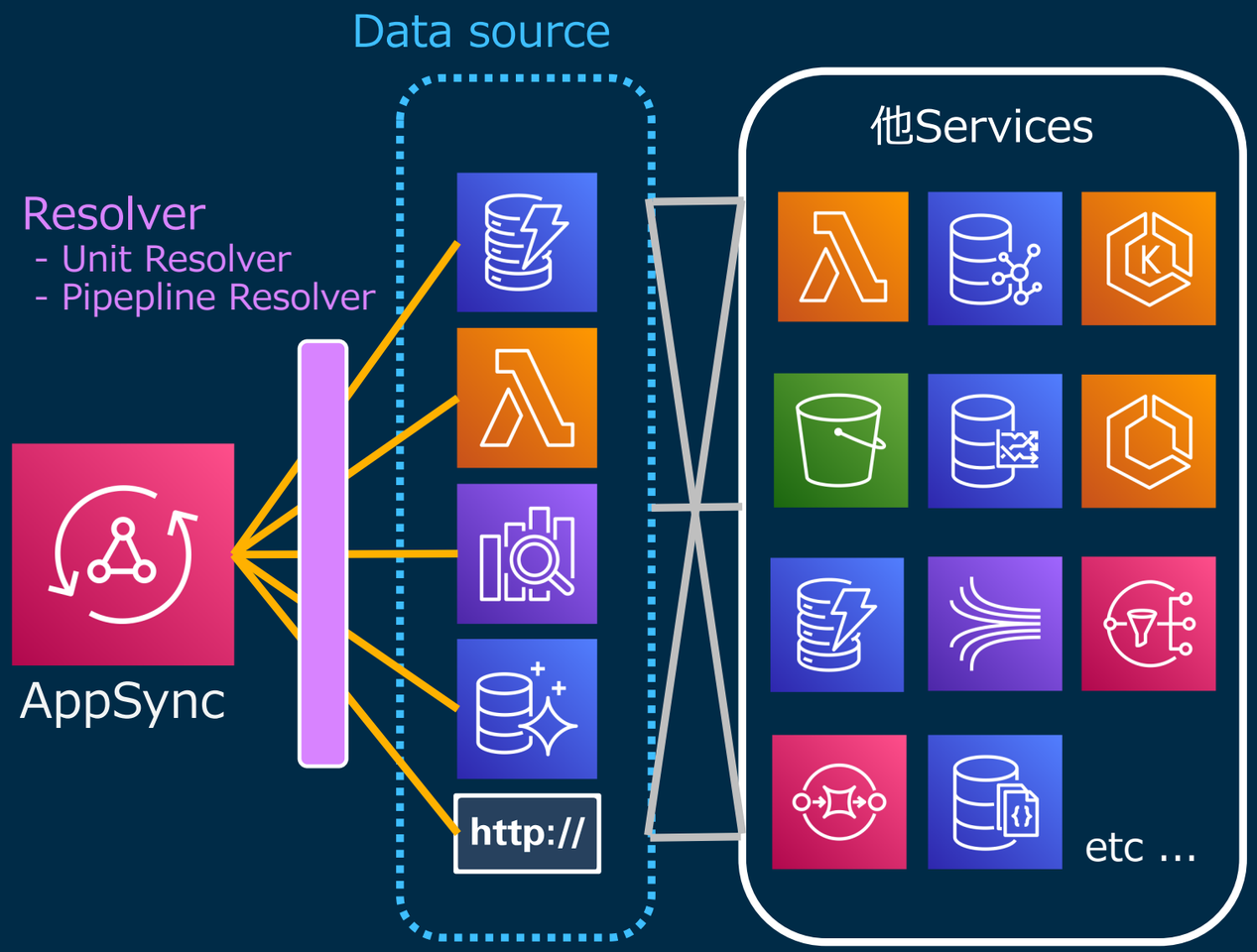

In Partnership with intel

aws DEV DAY
ONLINE JAPAN

4. まとめ

In Partnership with intel

再掲：AWS AppSync アーキテクチャ



AppSync を BFF としたアーキテクチャの場合
Resolver、Data source、Backend services の組み合わせは開発者の自由

開発における選択肢が多用で拡張性が非常に高い反面、Workload に沿ったアーキテクチャデザインが重要となる

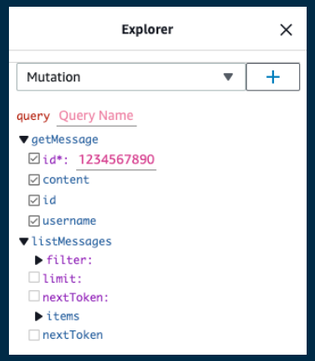
In Partnership with intel

AWS AppSync 最近の主要 Updates

AWS WAF 統合

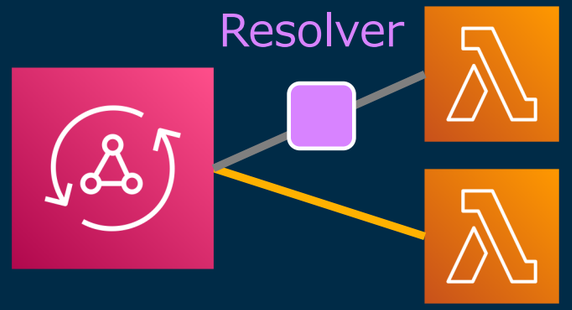
L7 脆弱性対策

GraphiQL Explorer



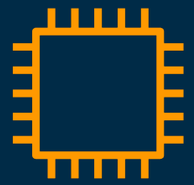
Console 上での GraphQL QUERY プロトタイピングの簡素化

Direct Lambda Resolver



Resolver(VTL) を介さず Lambda Data source の実行

Cache インスタンス追加



Cache.12xlarge

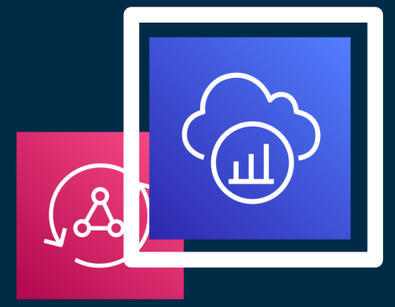
(48 vCPU / 317.77 GiB RAM)

Real-time Subscription

```
wss://xxx.appsync-realtime-api.
[region].amazonaws.com/graphql
```

任意の開発言語での Pure WebSocket Subscription の実装

AWS X-Ray 統合



In Partnership with intel

Happy coding with AWS AppSync

In Partnership with intel

aws DEV DAY ONLINE JAPAN

Thank you!

Atsuya Nunomura
Senior Solutions Architect
Amazon Web Services Japan K.K.

In Partnership with intel

aws DEV DAY
ONLINE JAPAN

Please complete the session survey in the mobile app.

In Partnership with intel