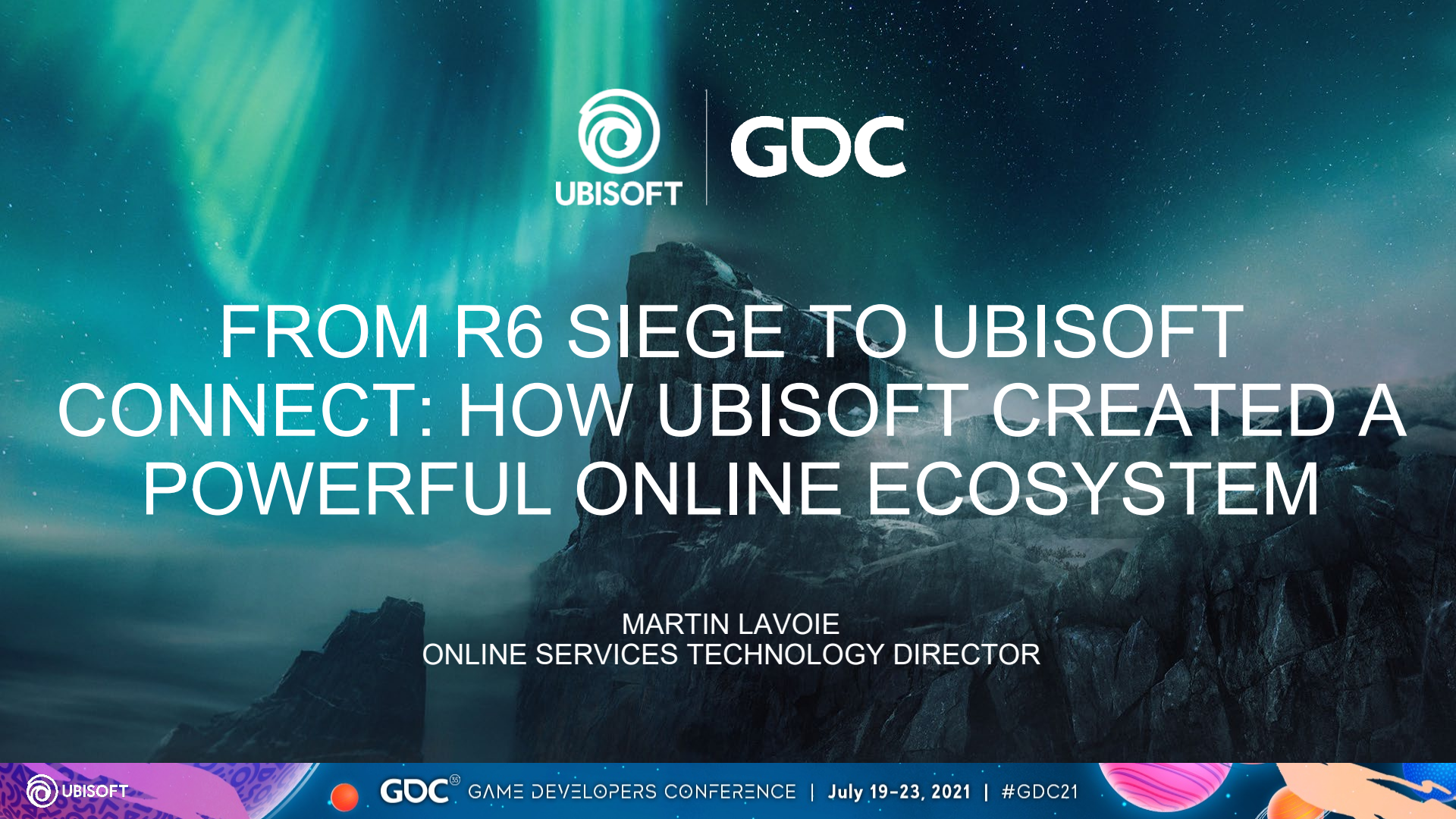


FROM R6 SIEGE TO UBISOFT CONNECT: HOW UBISOFT CREATED A POWERFUL ONLINE ECOSYSTEM

MARTIN LAVOIE
ONLINE SERVICES TECHNOLOGY DIRECTOR


SUMMARY
PART 1
WHY BUILD A PLATFORM
PART 2
HOW WE'VE DONE IT
PART 3
WHAT IT LOOKS LIKE NOW

PART 1

WHY BUILD A PLATFORM ?

A platform is a set of software and a surrounding ecosystem of resources that helps you to grow your business. A platform enables growth through connection: its value comes not only from its own features, but from its ability to connect external tools, teams, data, and processes *

*https://blog.hubspot.com/marketing/software-platform

HISTORICAL CONTEXT

Acquisition

Technology acquired by Ubisoft in 2008 followed by the company acquisition in 2010

Technology

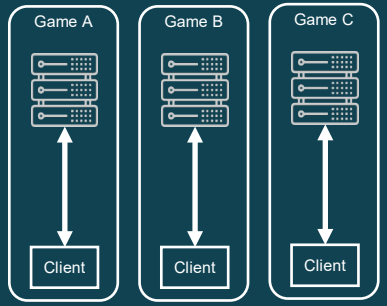
Application Server technology stack for writing scalable online services in Python. Provided with ready to use services.

Technology Group

Internalized as a new middleware offering within the existing central technology group

CHALLENGES

Keeping the technology middleware mindset brings more challenges than value



Multiple silos

Totally independent backend instances and isolated data

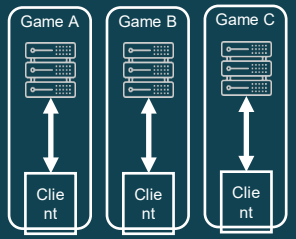
Snowflakes

Sharing a common piece of tech does not prevent ending up with snowflakes

Skillset

Developing services and operating the backend needs a lot of skills from game teams

CHALLENGES



Multiple silos

As many disjointed instances as games in live environments

Siloed Data and data formats across games
No global player ID

Per game integration of company wide tools, processes and policies
Customer support tools, Security & Regulation compliance

CHALLENGES

Snowflakes

Every game running different versions of the technology stack
Updating to latest version brings risk

Games teams would fork the service code to add new features

Data and APIs exposed differ from one game to another so data consumption outside of game is difficult

CHALLENGES

Skillset

Service deployment handled by game teams

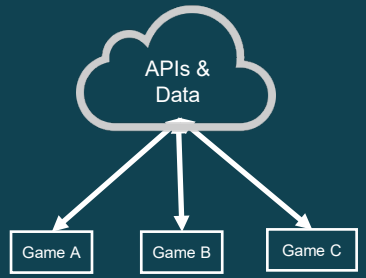
Proprietary technology ramp-up time
Need to master the technology stack internals

Effort spent on commodity rather than innovation

FROM TECHNOLOGY TO SERVICES

Build our own 1st party platform

2012



Global Platform

A modern multi-tenant and managed online services platform

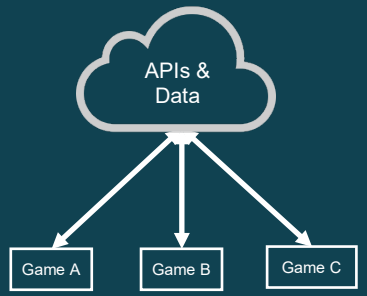
Standardization

Standard service APIs & Data formats

Climbing the stack

Let our game teams focus on innovation

FROM TECHNOLOGY TO SERVICES



Global Platform

Follow the Open API philosophy to easily allow all Ubisoft teams to create value for any or all games

Platform ugrades and extensions instantly benefits all games

Build with a modern micro-services architecture in mind

FROM TECHNOLOGY TO SERVICES

Standardization

Standardize commonly used services
Authentication, stats, leaderboards, secondary store

Integrating most standard service in our games is a requirement

Standard data and meta-data to ease data consumption by anyone on any game

FROM TECHNOLOGY TO SERVICES

Climbing the stack

Provide the basecamp, which should be commodity

Let the game teams focus on reaching the summit which is considered innovation

Allow for game teams to easily bring game specific innovative services to the platform

Using the most appropriate off-the-shelf language or technology to make a given service
This is a service implementation detail


PART 2
HOW WE'VE DONE IT
SECTION 1
MICRO-SERVICES FUNDAMENTALS
SECTION 2
PLATFORM FUNDAMENTALS
SECTION 3
ARCHITECTURE

MICRO-SERVICES FUNDAMENTALS

INDEPENDENTLY DEVELOPED & DEPLOYED

If you find the need to deploy services together,
you're doing something wrong.

If you have to send out a warning before deploying a service,
you're doing something wrong.

If you need to run your game to test your service,
you're doing something wrong.

MICRO-SERVICES FUNDAMENTALS

THE CONTRACT IS THE API

Everything else is implementation details
Language, Database or MQ being used

Data is private and shielded by the API, no backdoors.
A schema change should not affect other services or systems

One service supporting multiple API versions
Breaking the contract is not an option to preserve autonomy

MICRO-SERVICES FUNDAMENTALS

KEEP IT SIMPLE

Stateless over Stateful when possible

Stateless services are much easier to deploy and scale

Standard communication protocol and service interaction

We standardized on HTTPS(/2) to make service usage as simple and interoperable as possible

A good API is easy to use and understand

We standardized on RESTful APIs. Resources oriented API leads to better clarity and consistency across services

PLATFORM FUNDAMENTALS

Sharing standard concepts and requirements between services is what makes a consistent and coherent platform

TOP LEVEL RESOURCES

Services bring value to the platform by exposing new resources under the same graph

ENTRY POINTS

Services share the same base URLs and expose resources on all entry points

AUTHENTICATION & AUTHORIZATION

Services honor the same authentication and authorization mechanisms

OTHER REQUIREMENTS

Services implement platform wide requirements for consistency

PLATFORM FUNDAMENTALS

TOP LEVEL RESOURCES

User
- User's real identity, Ubisoft Account
- Contains GDPR highly protected data (Real name, email, etc…)

Profile
- User's gamer identity
- Contains game data (Stats, inventory, etc…)
- One profile by 1st party platform (XBL, PSN, Ubisoft)
- Linked to the User
- Cross progression always attached on Ubisoft profiles

Ubisoft PSN

PLATFORM FUNDAMENTALS

TOP LEVEL RESOURCES

Application

- The application calling our APIs
- Mostly used for API management
- Can be Games, Web Sites, Tools, Services

Spaces

- Virtual storage area that contains game data
- Allows for game isolation and sharding
- One space per platform for any given game
- Using the same space for all platforms enables cross-progression

PLATFORM FUNDAMENTALS

TOP LEVEL RESOURCES

Some examples

Getting a game's list of virtual item that can be purchased

```
GET /v1/spaces/{sid}/offers
```

Getting a player inventory in a given game

```
GET /v1/profiles/{pid}/inventory?spaceId={sid}
```

Getting a player stats in a given game

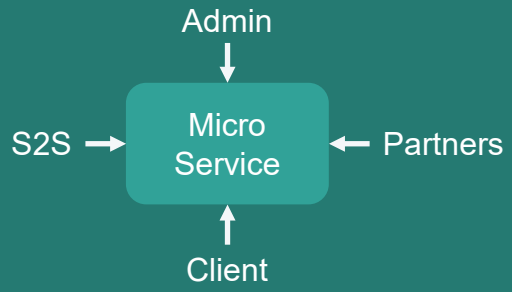
```
GET /v1/profiles/{pid}/stats?spaceId={sid}
```

Getting a leaderboard of a given game

```
GET /v1/spaces/{sid}/leaderboards/{lid}
```

PLATFORM FUNDAMENTALS

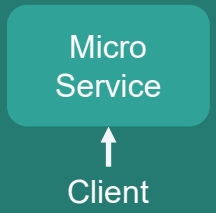
ENTRY POINTS



Serves Different Purposes
Distinct Base URLs
Distinct Authentication & Authorization
Distinct Security

PLATFORM FUNDAMENTALS

ENTRY POINTS



Player is calling the API through a game or web site
Basic Authentication or 1st Party tickets (e.g. XBL, PSN)
Authorized to its own or public data
Rate limited but not firewalled

Fetching own inventory or stats, Purchasing virtual item, Reading leaderbaords

PLATFORM FUNDAMENTALS

ENTRY POINTS

A Ubisoft Employee is calling the API through a tool
AD Authentication
Authorization given based on Employee role and spaces
Firewalled

Creating a new leaderboard, Configuring battlepass tiers, Adding items or changing prices for in-game store, Adjusting a player inventory

PLATFORM FUNDAMENTALS

ENTRY POINTS

S2S → Micro Service

Service to Service communication
Service Authentication
Authorized to given spaces
Exposing batch APIs

Dedicated server rewarding wining players at the end of a match,
Challenge service dropping a reward in a player inventory

PLATFORM FUNDAMENTALS

ENTRY POINTS

External Partners accessing player data on behalf of players
OAuth
Authorization given by players
Rate limited

Exposing a player stat or friends list to Discord, Community created web sites.

PLATFORM FUNDAMENTALS

AUTHENTICATION & AUTHORIZATION

Authentication

Centralized Authentication service providing JWT tickets

Tickets are mandatory for most APIs and Each service is responsible to validate them

Tickets contains type (Admin, Player, Service, Partner) and the ID of the caller

PLATFORM FUNDAMENTALS

AUTHENTICATION & AUTHORIZATION

Authorization

Centralized Role Based Access Control (RBAC) service

Privileges definition and validation is the responsibility of each service

Privileges are granted globally or per space for safe multi-tenancy

PLATFORM FUNDAMENTALS

OTHER REQUIREMENTS

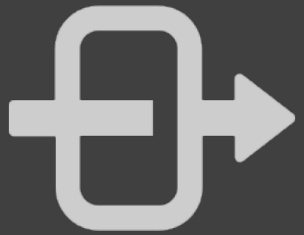
Centralized service to keep a standard audit trail of changes on the Admin entry point

Standard HTTP headers on all requests including Ticket, ApplicationID, SessionID

Standard application logs format

Rigourous API review

ARCHITECTURE



API GATEWAY

Allows for request routing that brings location independence, observability and API management for services

SERVICE NOTIFICATIONS

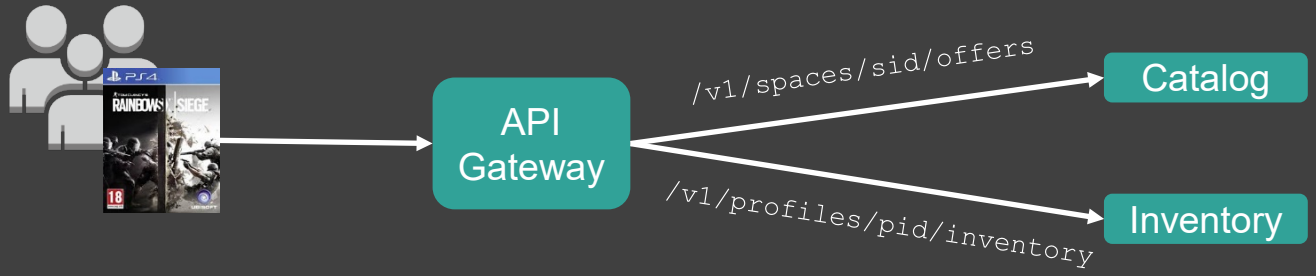
Allows for real-time asynchronous communication between services. This is key to open and extend services.

SERVERLESS

Serverless engine for containers allowing our service developers to focus on creating value

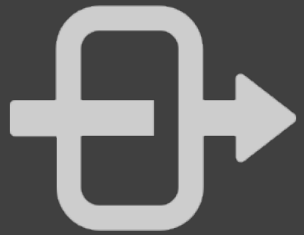
PLATFORM FUNDAMENTALS

API GATEWAY

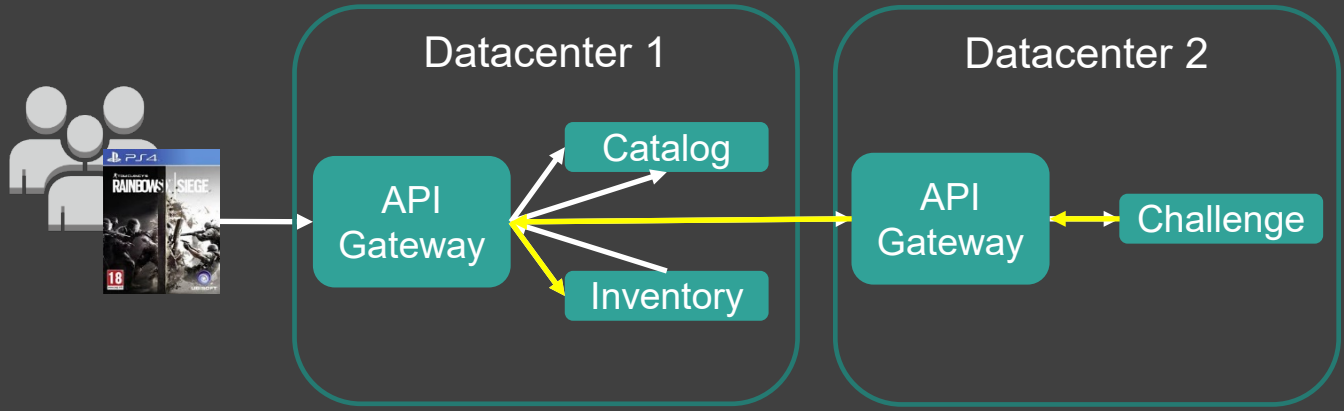


SSL offloading & certificate validation
Firewall rules & DDOS protection
API management (requests logging, throttling)
Path based routing

PLATFORM FUNDAMENTALS

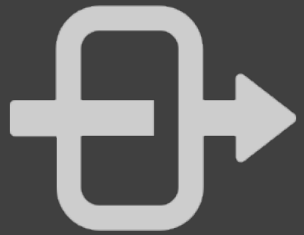


API GATEWAY



Datacenter awareness & routing
Moving services around does not break the API contract
Configuration service is used to avoid the extra hop
Gateway also used for service to service calls

PLATFORM FUNDAMENTALS



API GATEWAY

GLOBAL OBSERVABILITY

ApplicationID
SessionID
UserAgent
GameBuildID

API Gateway

ClientIP
RequestSize
Verb
URL
ServiceName
ServiceHostName
TransactionID

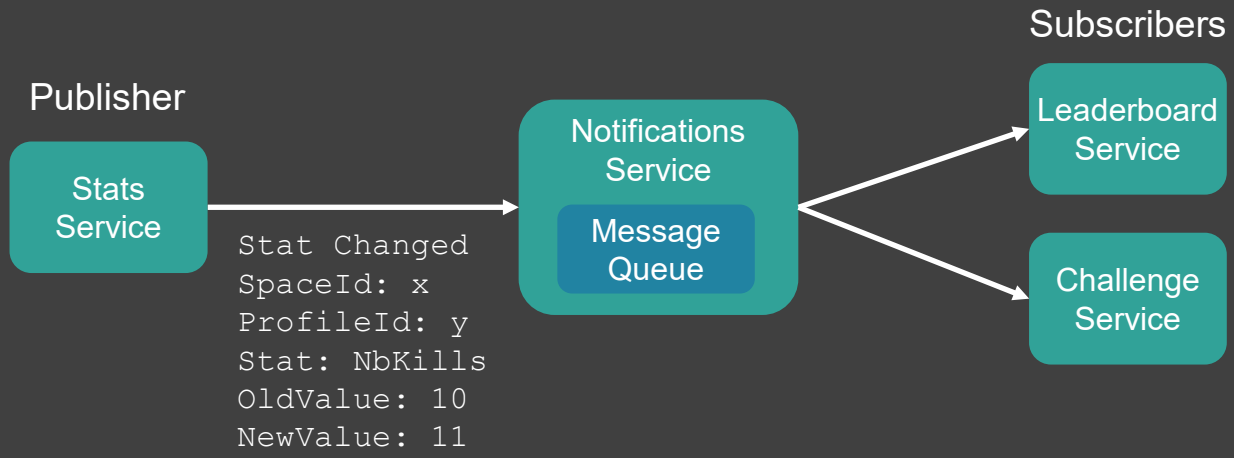
HTTP Result Code
Response Time
Response Size

Catalog

Service Quality monitoring & alerting
Session Tracing
Capacity Planning
Service Usage by Games
Troubleshooting

PLATFORM FUNDAMENTALS

SERVICE NOTIFICATIONS



All services must send notifications for every state change
Any service can register for any given notification for any space
Subscribers receive notification through a registered HTTP callback
Notifications are versionned since they are part of the Publisher contract

PLATFORM FUNDAMENTALS

SERVICE NOTIFICATIONS

Notification Service is the spinal cord of the platform

Brings asynchronous communication between services

Reduces hard dependencies between services

No change needed to the stats service to bring stats based challenges

Real-time state change propagation with notifications allows for easily creating new features/services on top of existing ones

Think IFTTT Triggers everywhere

PLATFORM FUNDAMENTALS

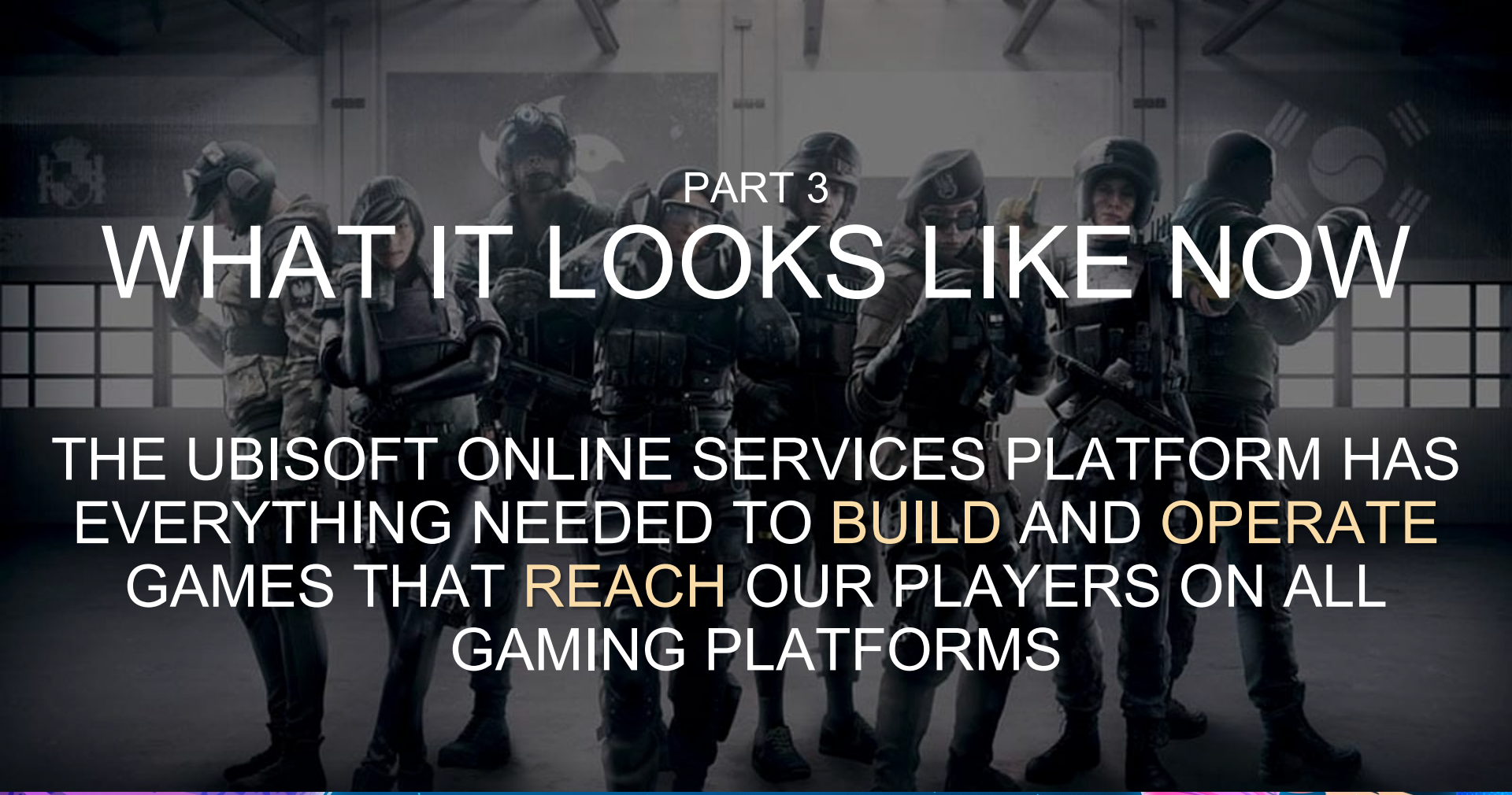
SERVERLESS

Service developer only need to care about implementing the containerized service

Using managed solutions for persistence (DB/MQs)

No infrastructure knowledge needed

CI/CD pipeline for managing and abstracting EKS/ECS clusters to easily deploy services on the platform without downtime



PART 3

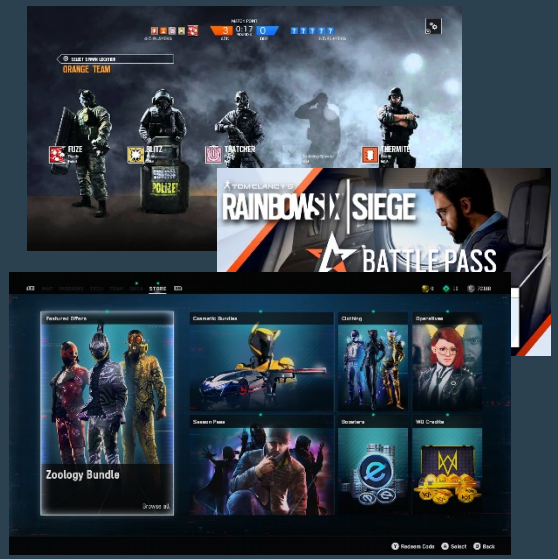
WHAT IT LOOKS LIKE NOW

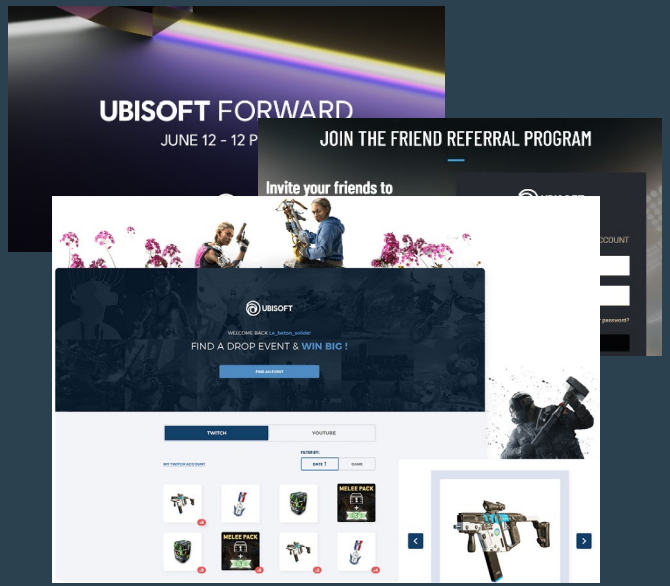
THE UBISOFT ONLINE SERVICES PLATFORM HAS EVERYTHING NEEDED TO BUILD AND OPERATE GAMES THAT REACH OUR PLAYERS ON ALL GAMING PLATFORMS

Build great live experiences

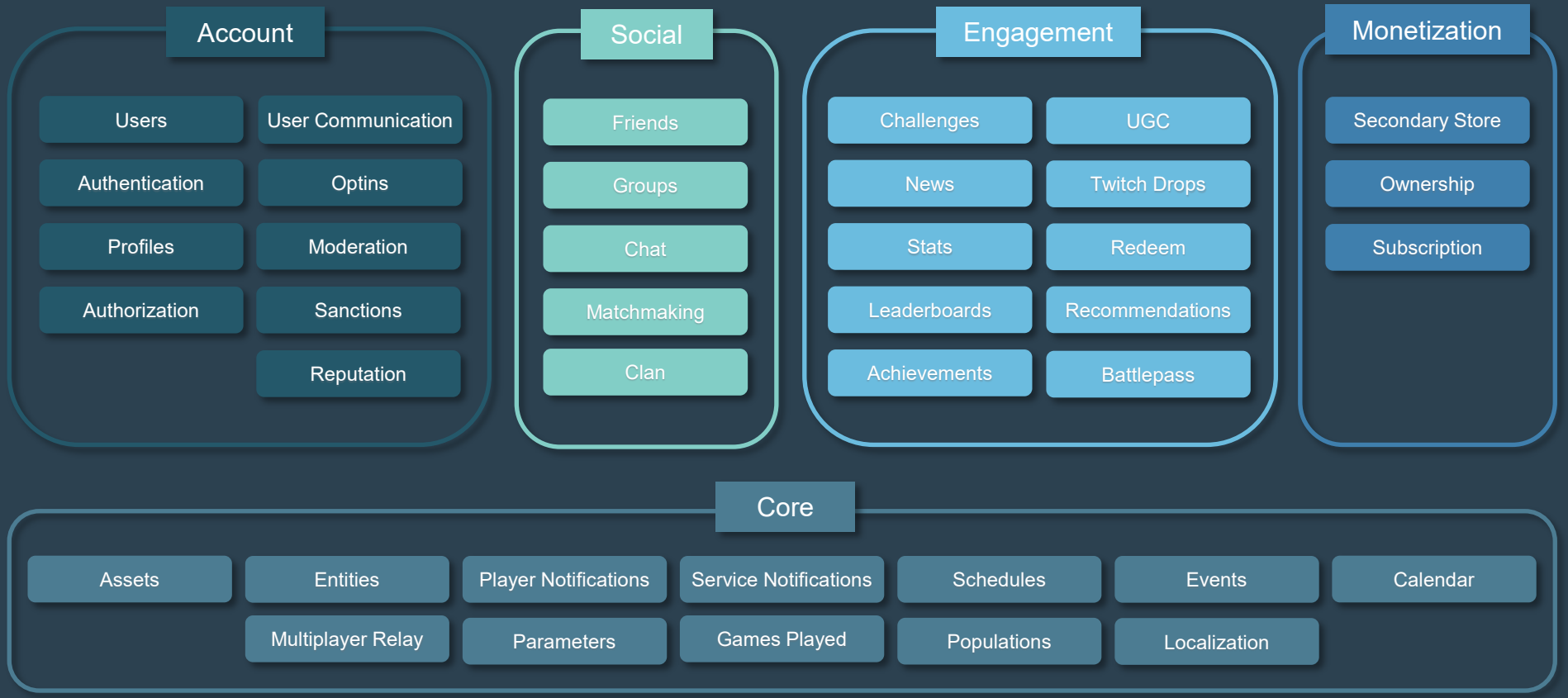
Manage Live Operations

Reach our Players





Rich set of global services enabling production teams to focus on their game



Game Streaming Solutions

Share Play for Journalists

The Ubisoft Connect application lets our demoist share unreleased version of our games to journalist through game streaming during our Ubisoft Forward virtual events

Cloud Pipeline for development

We are also leveraging our streaming capabilities to allow for a more efficient remote work for our employees and partners in this pandemic times.

Services and Technology for Multiplayer Gameplay

Gameplay Replication

Transport protocol, Object Duplication and Session Management

Game Server deployment

Geographically distributed and cloud agnostic game server deployment services


100+ Microservices
Integrated in 200+ Games running on all gaming devices including mobile and streaming
Serving 1500+ Application Clients
Integrated with 38 Partners
Reaching 140M YAU

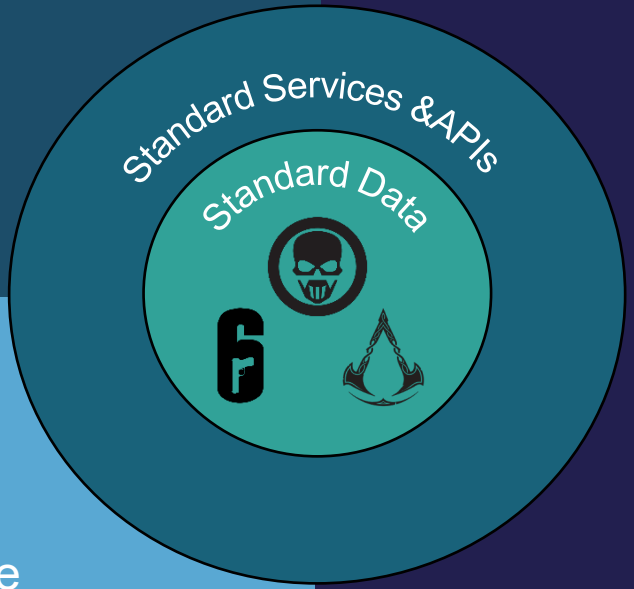
Build and Operated by a Geographically Distributed Team

Build Online Experiences

Games
Ubisoft Connect
Web Sites
Partner Extensions

Operate

Developer Portal
Service Development Pipeline
Customer Support Tool
CRM Tools
Campaign & Events



Reach


THANK YOU
UBISOFT