FSI203

AXA: Rearchitecting with serverless to accelerate innovation

Vincent Hiribarren
Technical Leader
AXA Group Operations

Xavier Loup
Principal Solutions Architect
AWS

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

- AXA TEX connected car pipeline project
- Issues of first IaaS-oriented platform
- Rework to serverless platform
- Advantages and lessons learned



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Who we are

Multinational insurance firm that engages in property-casualty insurance, life and savings, and asset management

Present in 57 countries, with 160,000 employees and distributors, to serve our 108 million clients

AXA GO

AXA Group Operations
Emerging technologies
and data division

Explores and scales value of data and emerging technologies with potential to disrupt current insurance business model

Telematics team

6 years in business

Flagship project – automotive telematics, which combine data science and data processing chain

Team-developed and maintained solution (data science, network, mobile, backend, and so on)

TEX project – Telematics Exchange platform

Business rational

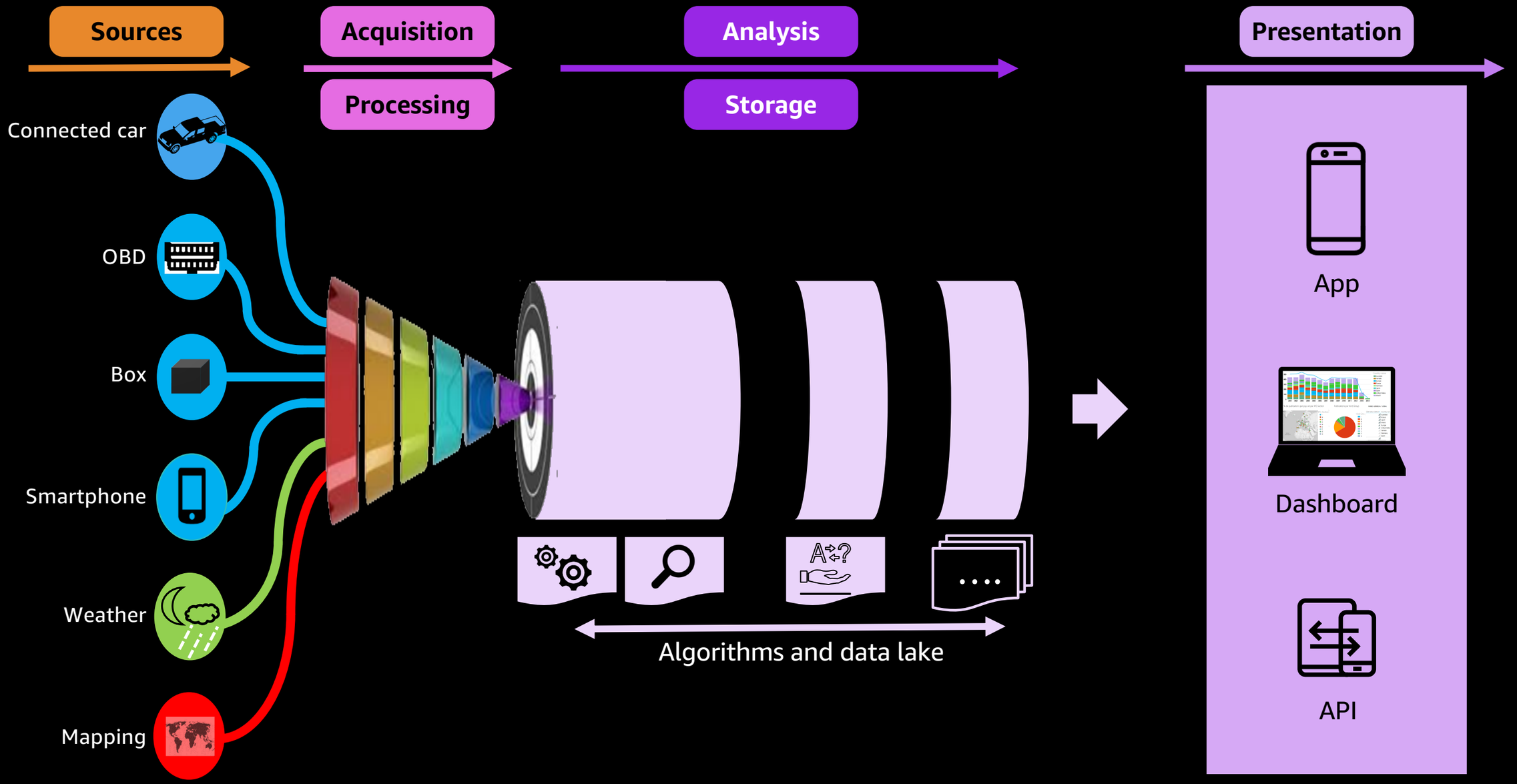
- Develop data-driven motor insurance policies
- Pay per mile, per day, per risk profile, with periodic premium adjustment
- Provide immediate “post-trip” risk feedback to help customer drive safer

Where we are today

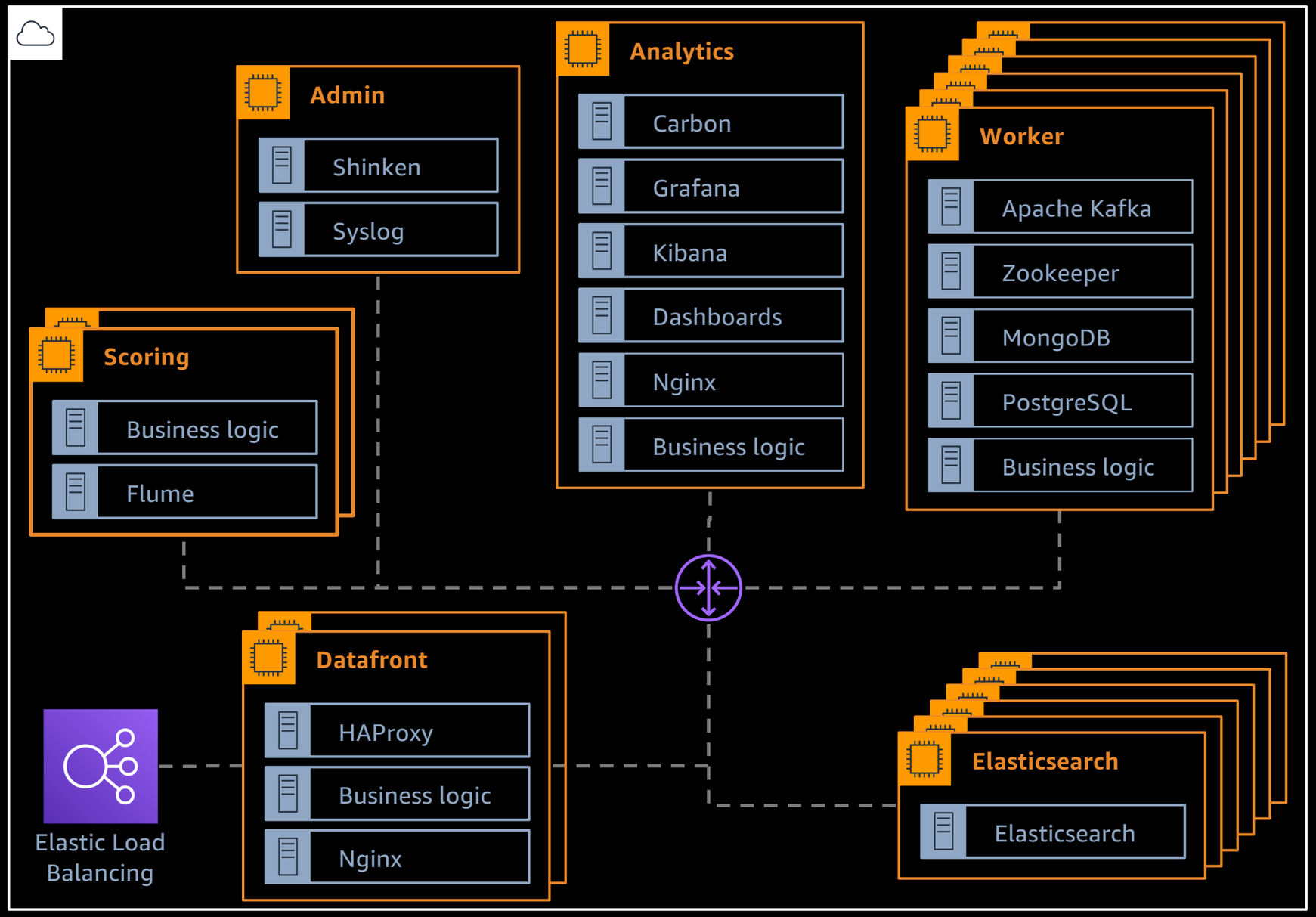
- Live connection with black boxes, smartphones, and native car connectivity as GPS sources
- 5 offers on EU market
- 40,000+ customers
- 60,000+ vehicle trips per day



TEX – High-level functional view



TEX as we built it in 2015 (v1)



1 AWS account, multiple environments

Over 50 Amazon EC2 instances in total

Designed for portability to AXA data center

- Deployment with Ansible
- Limited AWS technologies used
- Infrastructure with Terraform

Efforts to manage

- Setup of monitoring tools
- Regular fix / security updates needed
- Difficult to add new components
- Deployment complex and slow
- HSM and IPSec links to manage due to strong security constraints

Why we needed to change?



Less system engineering

Amazon EC2 management requires regular system updates, advanced monitoring and major operational challenges



Easier scaling

Scalability of our architecture is possible, but both manual and difficult

Lack of scalability of some components due to hasty decisions



More security

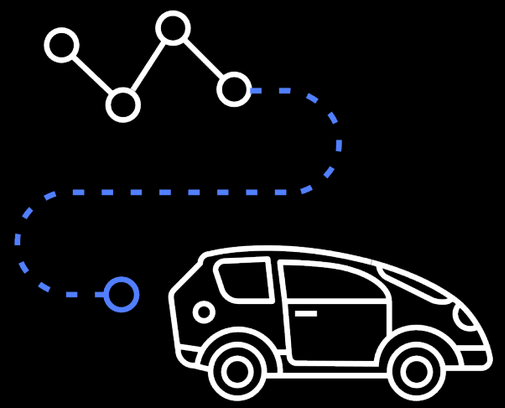
We are subject to strong constraints that are difficult for our team to manage



Focus on features

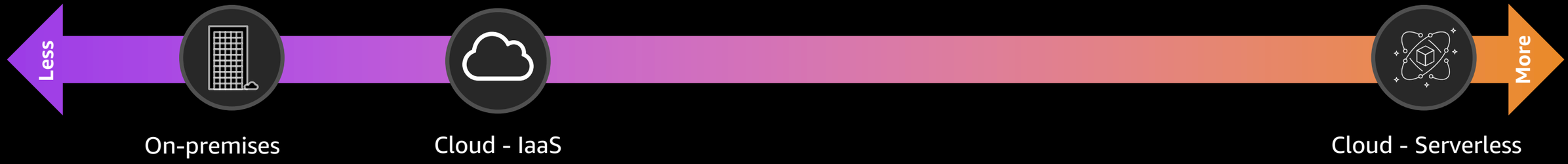
We want to spend more time developing functionalities than managing a system

TEX v2



Rearchitecting with serverless to address the limitations of TEX v1

AWS operational responsibility models



	On-premises	Cloud - IaaS			Cloud - Serverless
Compute	Virtual machine	Amazon EC2	AWS Elastic Beanstalk	AWS Fargate	AWS Lambda
Messaging / event streaming	ESBs	Kafka on Amazon EC2	Amazon Managed Streaming for Apache Kafka (Amazon MSK)	Amazon Kinesis	Amazon SQS / Amazon SNS
Relational database	PostgreSQL	PostgreSQL on Amazon EC2	Amazon RDS	Amazon Aurora	Aurora Serverless
Document-oriented database	MongoDB	MongoDB on Amazon EC2		Amazon DocumentDB	Amazon DynamoDB

TEX v1

Moving to serverless – Compute



TEX v1

Container **and** Lambda

Usage

- Communication with telematics boxes via UDP
- Long batches

Benefits

- Flexibility of containers
- Serverless infrastructure

10%

Usage

Asynchronous processing of car data stream

Benefits

- High scalability
- Cost optimization
- Integration with other AWS services (e.g., Kinesis)

90%

Moving to serverless – Messaging / Streaming



TEX v1

Streaming up to 1,000 car events per second

Usage

- Car event streaming
- Multiple consumers
- Need to preserve order

Benefits

- Easy to scale by increasing number of shards
- Possibility to batch events to reduce costs

Usage

Message queue to compute driver score

Benefits

Automatic scalability

Moving to serverless: Databases

SQL
and
NoSQL

Data	Database TEX v1	Database TEX v2
Current trips	MongoDB on Amazon EC2	DynamoDB 
Driver scores / trips history	Elasticsearch on Amazon EC2	Aurora Serverless – PostgreSQL 
Cartography	PostgreSQL on Amazon EC2	

AWS operational responsibility models



	On-premises	Cloud - IaaS			Cloud - Serverless
Compute	Virtual machine	Amazon EC2	Elastic Beanstalk	Fargate	Lambda
Messaging / event streaming	ESBs	Kafka on Amazon EC2	Amazon MSK	Kinesis	Amazon SQS / Amazon SNS
Relational database	PostgreSQL	PostgreSQL on Amazon EC2	Amazon RDS	Aurora	Aurora Serverless
Document-oriented database	MongoDB	MongoDB on Amazon EC2		Amazon DocumentDB	DynamoDB

TEX v1

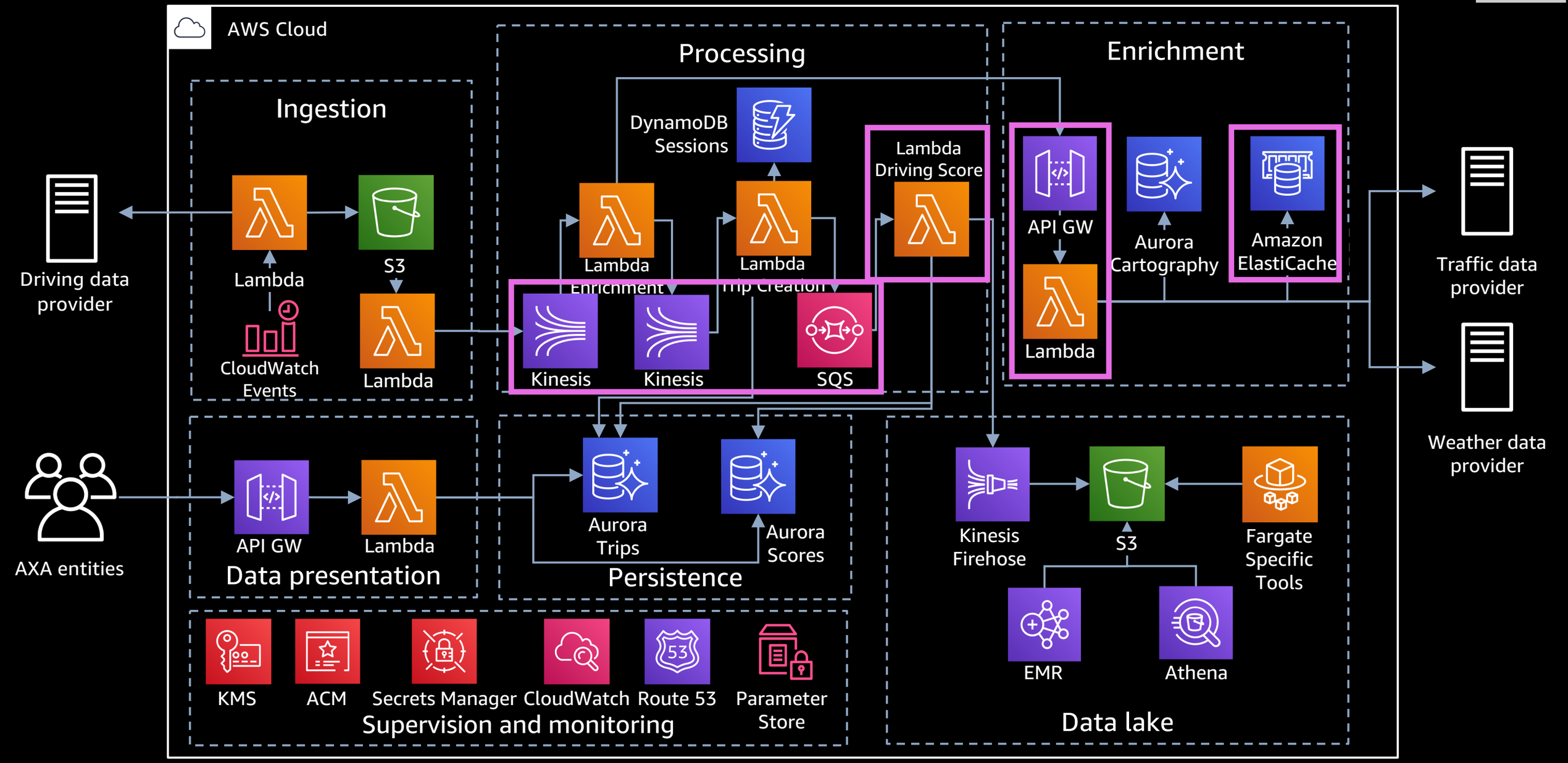
AWS operational responsibility models



	On-premises	Cloud - IaaS			Cloud - Serverless	
Compute	Virtual machine	Amazon EC2	Elastic Beanstalk		Fargate	Lambda
Messaging / event streaming	ESBs	Kafka on Amazon EC2	Amazon MSK		Kinesis	Amazon SQS / Amazon SNS
Relational database	PostgreSQL	PostgreSQL on Amazon EC2	Amazon RDS	Aurora	Aurora Serverless	
Document-oriented database	MongoDB	MongoDB on Amazon EC2		Amazon DocumentDB		DynamoDB

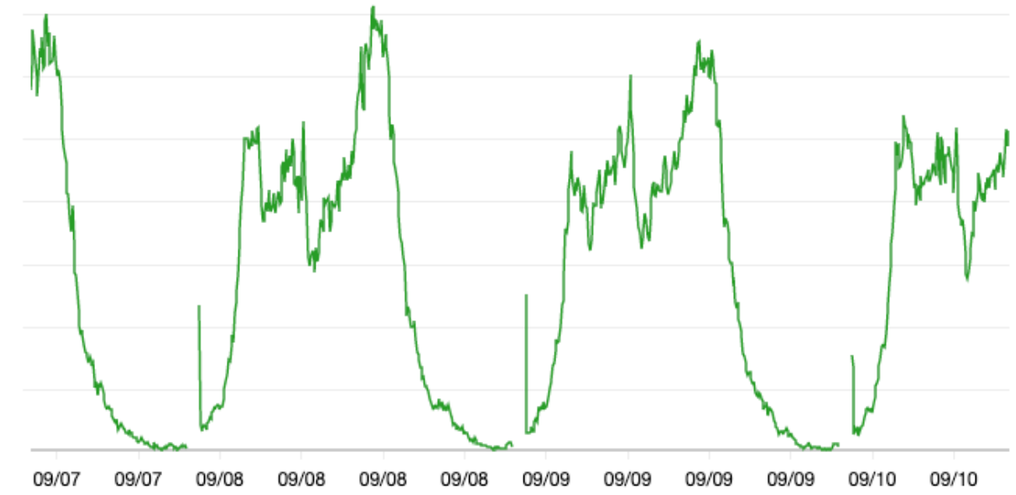
TEX v2

TEX v2 – Simplified processing pipeline

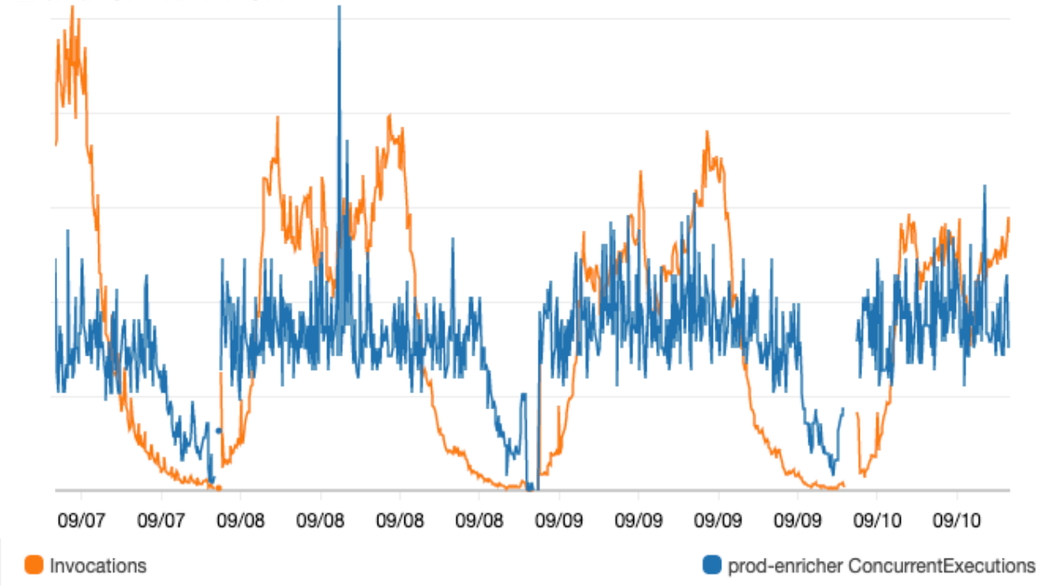


TEX v2 – Lambda activities

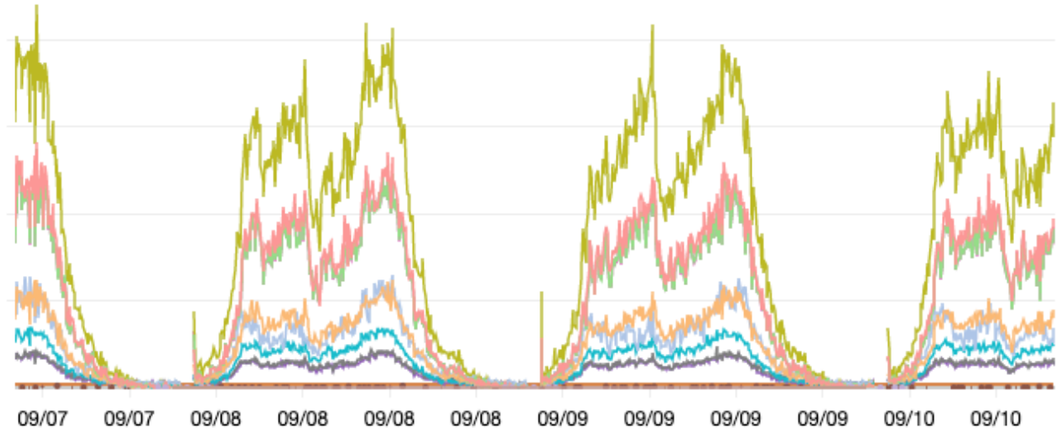
Kinesis incoming records



Data enrichment



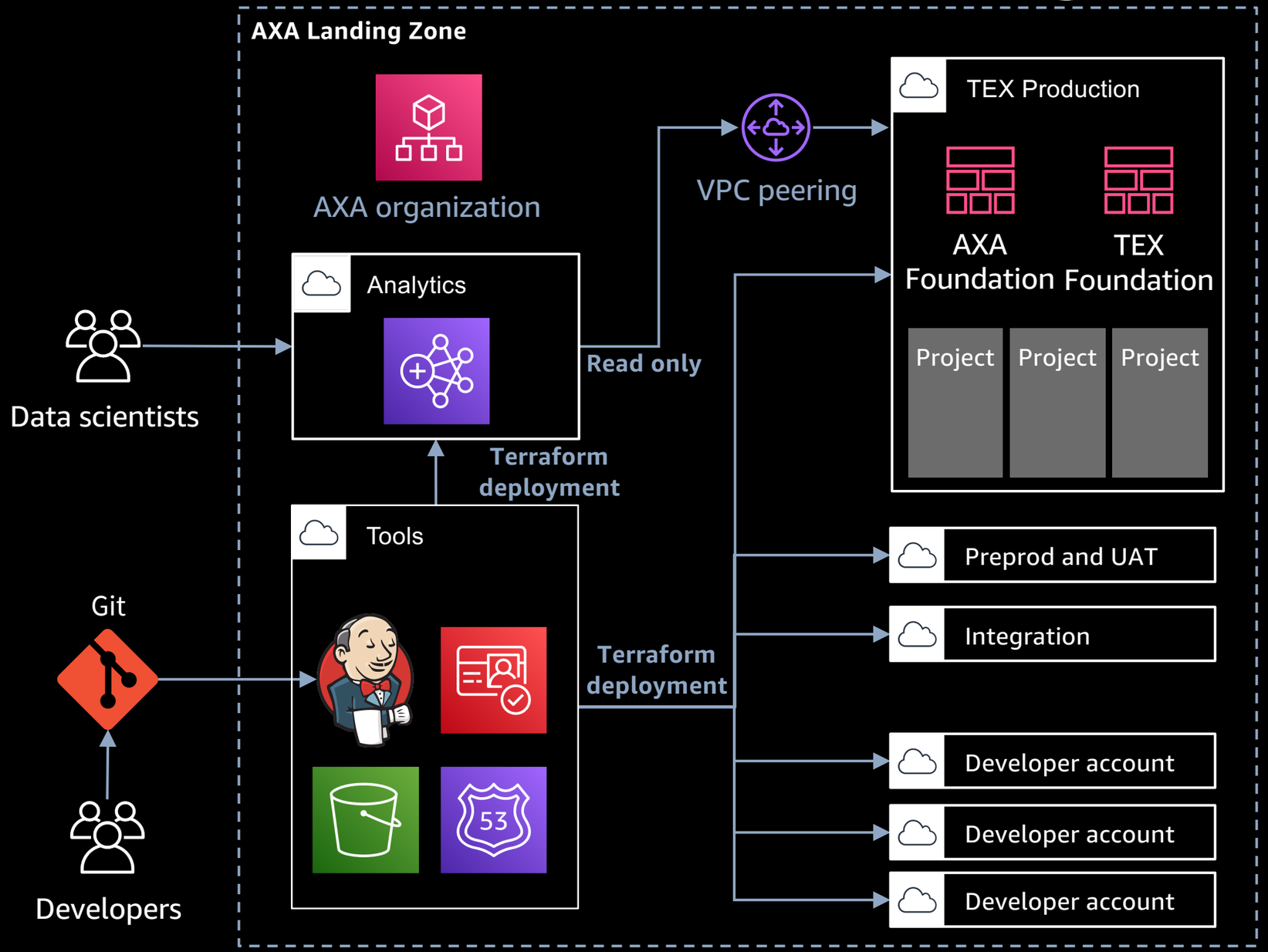
Lambda invocations



Business logic scaling effect

- Smoothly scale depending on road traffic
- Development environment – no traffic
  ≃$0 for Lambda
- Production environment – lower cost than fault-tolerant servers dedicated to business logic

TEX v2 – AWS Accounts organization



AXA Landing Zone

- Activation of monitoring and security tools, such as CloudTrail, Amazon GuardDuty, service control policies, and so on
- Authentication management
- Ensure compliance with Config
- Centralized billing

→ Presentation made at AWS re:Invent 2019

Prime accounts

One AWS account per variant – prod, preprod, development

Analytics account

- Read-only access for the production
- VPC peering for Aurora access
- Monitored actions of data scientists

Tools account

- CI / CD (Jenkins)
- Artifact library (S3)
- DNS management

TEX v2 – Did we succeed in resolving our issues?



Less system engineering



We do not think about it anymore

Remaining tasks are on database performance with business logic



Easier scaling



The architecture and the infrastructure just scale

Some components may need manual tuning



More security



A full class of issues disappeared with the serverless migration



Focus on features



We spend less time in maintaining the infrastructure,

So we are more comfortable in adding new features

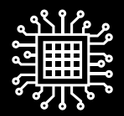
Lessons learned



Serverless still needs configuration

Aurora Serverless requires a VPC configuration

Kinesis shards and Aurora units need manual tuning



Techno maturity

Fewer features with Aurora Serverless than with Amazon RDS equivalents

Kinesis costs, even if not used



Hybrid architecture

Full serverless is not always possible, hybrid approach should be kept in mind and integrated in infrastructure as code



Control automatic scaling

Lambda can scale too much and be costly if bug in logic or architecture – set alarms and limits

Credits – TEX v2 team

Telematics Department Manager
Jérôme Lafon

Product Owner
Guillaume Dosser

Technical Leader
Vincent Hiribarren

CI / CD and Tooling
Rasmey Sareth

Backend Developers
Gautier Pradier
Julien Simon

Data Scientists
Alexandre Corradin
Arnaud De Bergh
Paul Martial

Mobile Developer
Erwan Masson

Git Logo by Jason Long from https://git-scm.com/downloads/logos is licensed under CC BY 3.0.
The Jenkins logo from the Jenkins project from https://jenkins.io/ is licensed under CC BY 3.0.

Thank you!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.