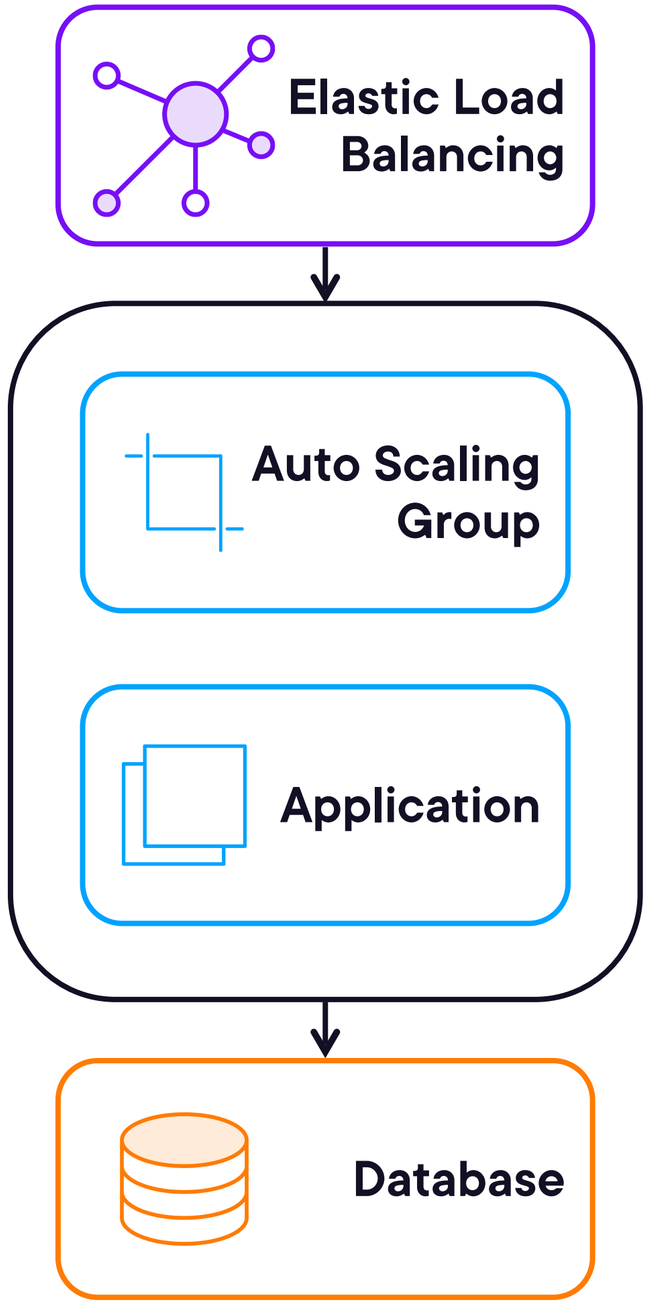
Introducing Pub Sub Systems

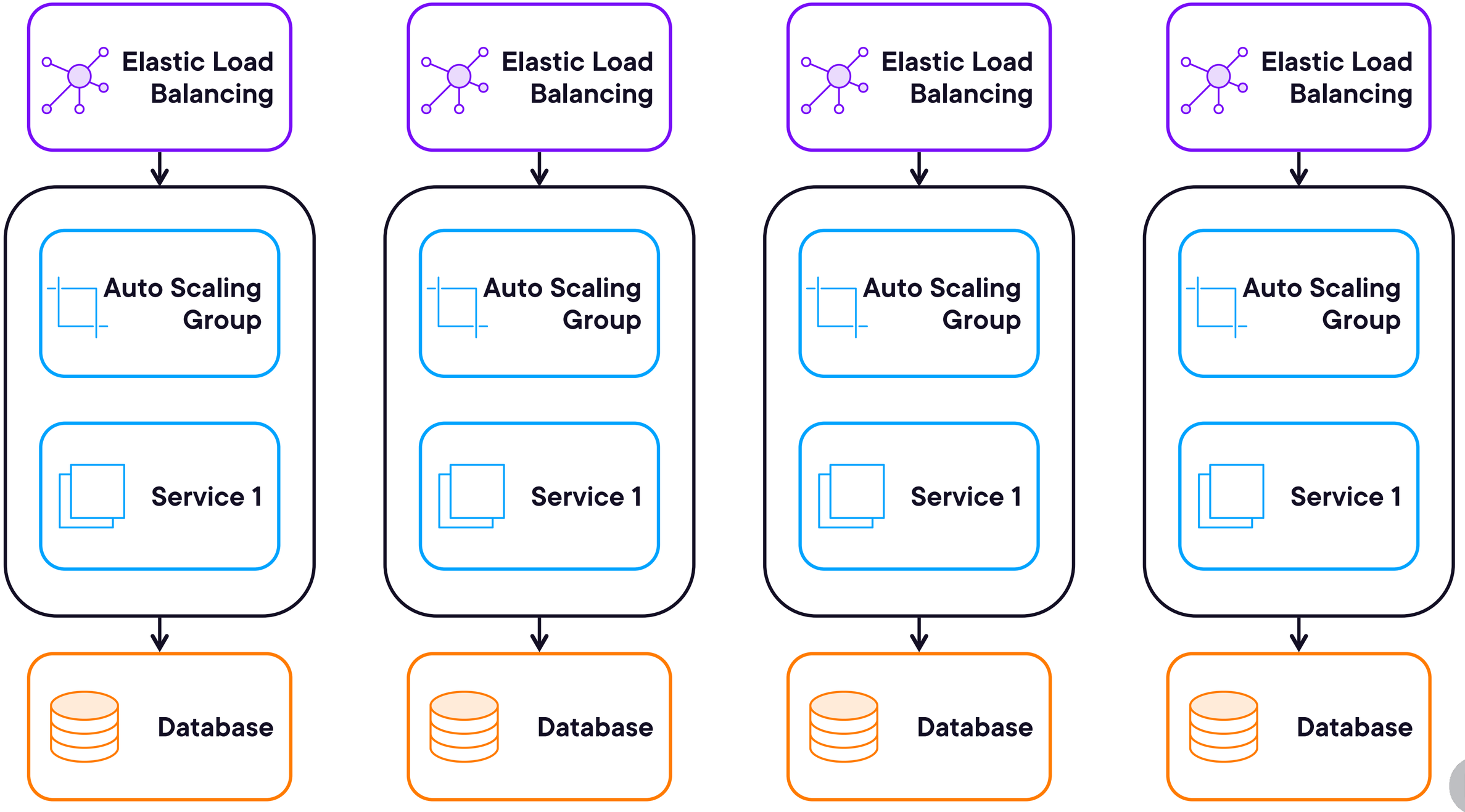
Meeting Pub-Sub Systems

Axel Sirota

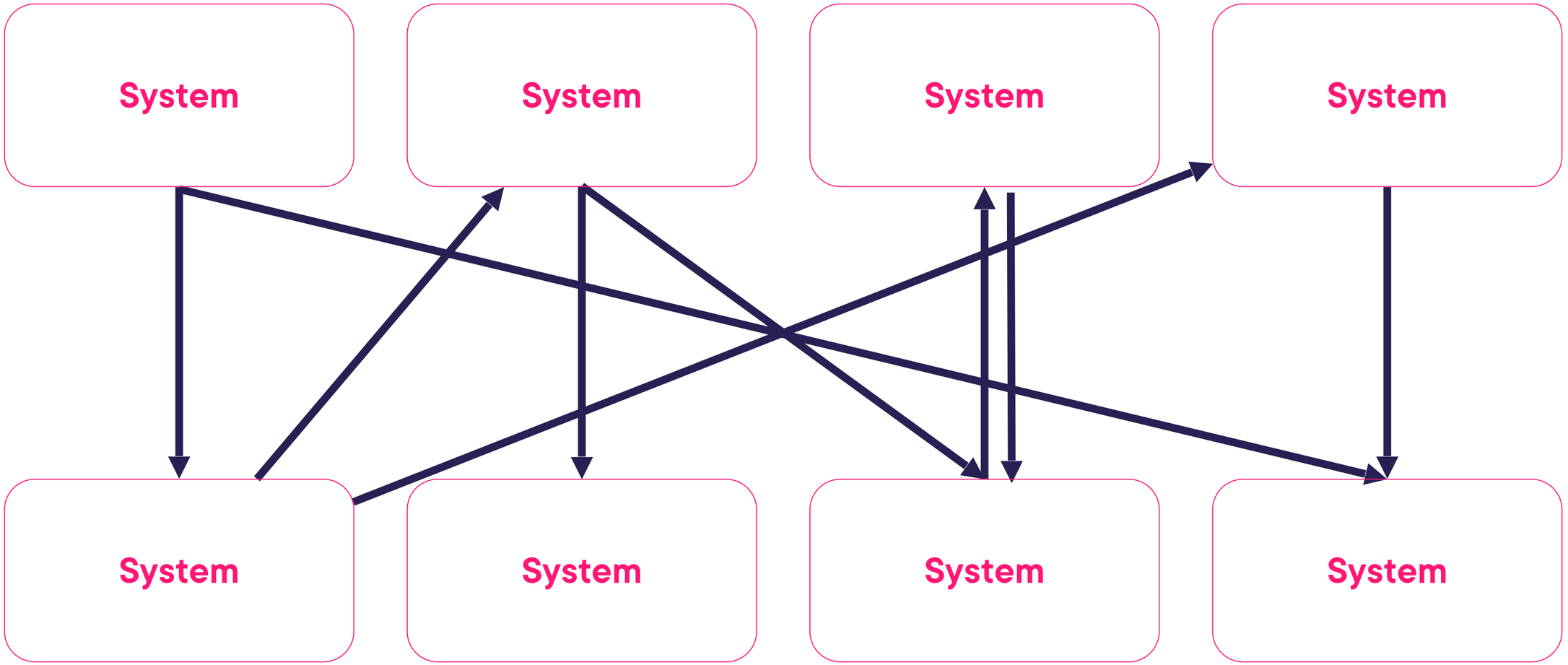
AI and Cloud Consultant

@AxelSirota


Elastic Load Balancing
Auto Scaling Group
Application
Database


Elastic Load Balancing
Auto Scaling Group
Service 1
Database
Elastic Load Balancing
Auto Scaling Group
Service 1
Database
Elastic Load Balancing
Auto Scaling Group
Service 1
Database
Elastic Load Balancing
Auto Scaling Group
Service 1
Database

Entangled System




Kafka


Items Service


UI Service

We Call:

The entity/app creating a message, a publisher or producer

The entity/app consuming messages from a channel as a consumer

The system where the channels live and handle this requests as an Event Bus or, more recently, streaming platforms

The channel where messages flow as channel or topic

Back to Definitions

We say the pub-sub system is reliable when you ensure there is no message loss

Has at most once processing when you ensure there is no message duplication

And has exactly-once processing when you only process a message once ensuring it wasn’t lost. Of course, the holy grail.

We Had Ton of Other Pub/Sub Systems Before:

RabbitMQ

Mulesoft

Redis

Amazon SQS

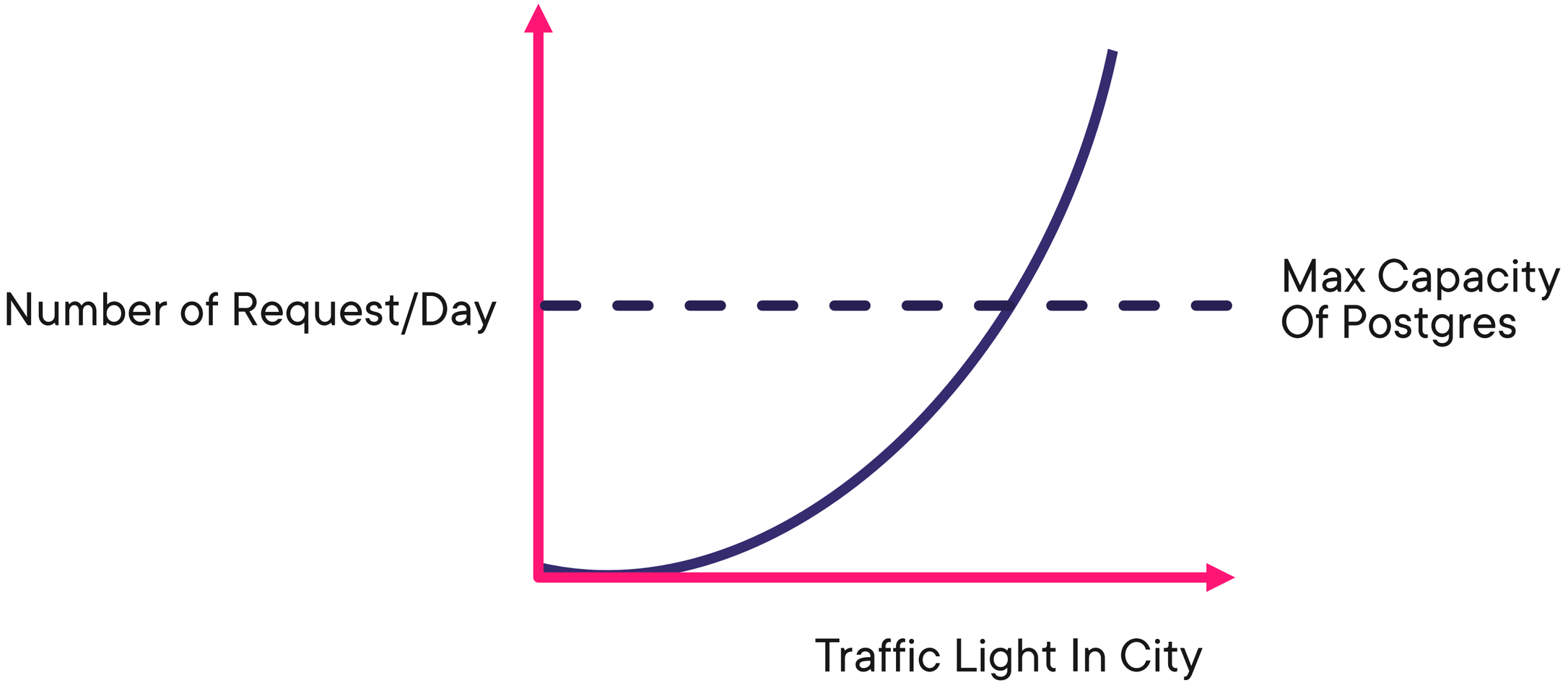
Amazon SNS

Azure HDInsight, Azure Streaming Service, and Google Pub/Sub

Why Kafka?

A Case Study: Traffic Lights

A Case Study: Traffic Lights

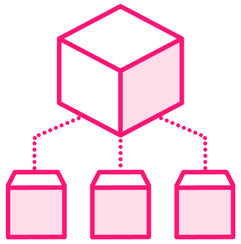


A Case Study: Traffic Lights

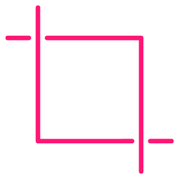
A Case Study: 911 Systems

Kafka comes to solve this!

What Is Kafka?



Event Streaming Plataform



Scalable

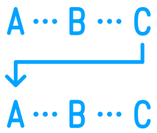
High Throughput

Difference between Kafka and the Rest of Solutions

RabbitMQ: As Kafka uses topics in an unidirectional log without handshake, the **difference in performance** is brutal, easily 1000x

Mulesoft: Kafka **makes it simple** to create a channel of communication simply by producing a message the topic will be autocreated

Redis: Redis **does not handle streaming data** nor has a history past messages

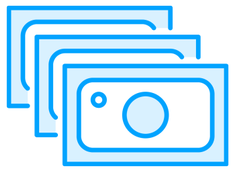


Amazon SQS: It based on queues, way better tan RabbitMQ but still there is an **order of magnitude in throughput** and speed between Kafka and SQS. Plus SQS is not exactly cheap

Difference between Kafka and the Rest of Solutions

Amazon SNS: Very, very similar; but **way pricer**

Azure Streaming Services: This service in particular is **not exactly meant for pub-sub in general**



Google Pub/Sub: The same as Amazon SNS, very similar but **pricer**

Demo: Introducing Globomantics Case Study

- Amazon bill went up 300x
- Apache Kafka can replace such services
- It can reduce the bill significantly

Getting the Best out of This Course

Prerequisites

Prerequisites

- Basic Java

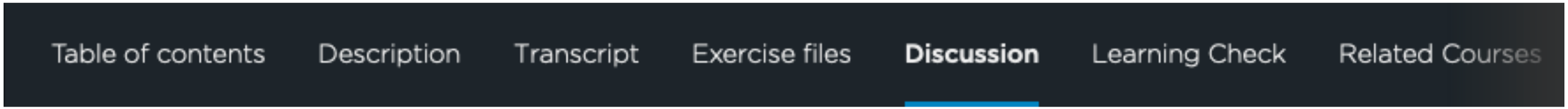
- Basic Docker

Course Suggestions

- Java SE 17 Fundamentals

- Docker: The Big Picture

More Resources



SKILLS

Home

Browse

Search...

AUTHOR TOOLS

Home

Analytics

Author's nest

Author kit

Axel Sirota

Pluralsight Author

Following

133 Followers

Ask me on Twitter!

@AxelSirota

GitHub Repository for the Course

axel-sirota add all	`429f0d2` 11 hours ago	**4** commits
module3	add all	11 hours ago
module4	add all	11 hours ago
.gitignore	module 3 demos	3 days ago
LICENSE	Initial commit	4 days ago
README.md	Initial commit	4 days ago
requirements.txt	add structure	2 days ago
workspace.yml	module 3 demos	3 days ago

https://github.com/axel-sirota/getting-started-kafka

Version Check

Version Check

Apache Kafka – Confluent Version 7.4 (equivalent to Kafka open source 3.4)

Apache Zookeeper 3.8.1

Kafka Rest Proxy – Confluent Version 7.4

Kafka Connect – Confluent Version 7.4

ksqldb 0.29

Schema Registry – Confluent Version 7.4

Kafka Streams – Confluent Version 7.4

Version Check

This course is 100% applicable to:

- Apache Kafka – Confluent Version ≥ 6.0 (equivalent to Kafka open source version ≥2.6)
- Apache Zookeeper 3.4.10 through 3.5.9 (3.5.9 recommended)
- Kafka Rest Proxy – Confluent Version ≥ 6.0
- Kafka Connect – Confluent Version ≥ 6.0
- ksqldb 0.2
- Schema Registry – Confluent Version ≥ 6.0
- Kafka Streams – Confluent Version ≥ 6.0

https://docs.confluent.io/platform/current/installation/versions-interoperability.html

Version Check

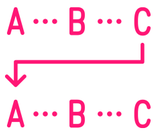
This course is NOT aplicable to:

- Apache Kafka - Confluent Version < 6.0
- Apache Zookeeper < 3.4.10

What Will You Be Able to Do When Finishing This Course?

After This Course, Not Only:

You will be able to create applications that produces and consumes messages from a topic in Kafka



Actually you will be fully equipped to create an event driven application with full governance over how messages flow

You will be able to expose this messages as an API to perform queries over it, or even create tables to understand how Streams of data behave

Finally, you will also have the knowledge to secure your cluster and scale it up to have high availability

Outline Of The Course

Kafka Architecture

Kafka Streams

Perform administrative tasks

Creating and Configuring Producers and Consumers

Kafka Connect

Final Thoughts

Key Takeaways