

Popular streaming systems

STREAMING CONCEPTS



Mike Metzger
Data engineer

Streaming tools

- Various tools available depending on needs
- Allows designers to specify the **best tool for the job**
- Common systems include:
 - Celery
 - Kafka
 - Spark Streaming



Celery

- **Distributed task queue / FIFO**
- Used primarily as a **job** or **task queue**
- Often used for **asynchronous tasks**
 - Sending password reset emails
 - Fulfilling digital orders
 - Resizing images
- Allows for **real-time processing** of significant quantity of messages
- Provides functionality for **management and scaling** (vertical & horizontal)



Apache Kafka

- **Distributed** event streaming system
- Designed to send events between **producers** and **consumers**
 - **Producers** create events on a **topic**
 - **Topics** are basically **messages** of a specified form
 - **Consumers** receive new events
- Different consumers can **handle events as needed** (logging, transformations, relaying, etc)
- Handles **storing of events** as specified
- Extremely **powerful**, but can be **tricky** to **set up**



Kafka applications

- Best used for **passing data between multiple systems**
 - **Single** source of truth
 - Change data **capture**
 - Data **backups**
 - Data system **migrations**

Spark streaming

- Part of **Apache Spark**
- Designed to process **streaming** data
- Builds upon the capabilities of Spark to process data in **Scala, Python, SQL, and so forth**
- Useful for processing **large amounts** of data and in **machine learning** scenarios
- Able to **transition from batch to stream** processing fairly easily
- Not designed to store or log events, but primarily to **process or modify** the data



Let's practice!
STREAMING CONCEPTS

Real-world use case: streaming music service

STREAMING CONCEPTS



Mike Metzger
Data Engineer

Streaming music

- Consider the scenario
- **Not focusing on the actual music** being streamed
- More interested on the **user(s)**
 - Interactions
 - Music preferences
 - Other details

Interactions

Primary questions

- What?
 - When?
 - Where?
- Like / Don't play
 - Next / Previous / Skip
 - Select Channel / Playlist
 - Add / remove song from playlist

How to store data

- Data is archived as a **log**
- Sent as **interactions** occur
- Number of interactions will **vary considerably** between users
- Logged data can be **analyzed later** on

Analytics

- What about **preferences**?
 - Can be obtained from logged data
 - Favorite genres, bands, etc
 - Most popular times of day
- Other **details**?
 - Most popular app platform / version
 - Location data from stream

Let's practice!
STREAMING CONCEPTS

Real-world use case: sensor data

STREAMING CONCEPTS



Mike Metzger
Data Engineer

What is sensor data?

- **Automated devices** that monitor some aspect of interest
 - Temperature monitors
 - Electricity usage monitors
 - Vehicle presence detection
 - Many others
- Tend to communicate with **centralized services** for management and data reporting
- Can range from a **few sensors to millions or even billions**

Connected doorbell

- Monitors primarily for **doorbell presses**
- Contains extra **video / audio capabilities**
- Allows **live, remote interaction**
- Can use camera / environmental sensors for additional **detection capabilities**



¹ C05731, CC BY-SA 4.0 [removed], via Wikimedia Commons

What are we monitoring?

- Button presses
- **Movement** detection
- **Sound** detection
- Requires more **intensive interaction** than just logging of events

Data handling

- How to store data
 - General event data (button press)
 - Sensor-based events (light sensor or audio pickup)
 - Raw data for later analysis
- Different services can have **different SLAs**, even in same product

Let's practice!

STREAMING CONCEPTS

Real-world use case: vaccination clinic

STREAMING CONCEPTS



Mike Metzger
Data Engineer

Data processing review

- **Batch**
 - Great for *large sets of data*
 - Potentially *poor latency*
- **Queue**
 - Awesome to *maintain order*
 - Can be *tricky to manage*
- **Stream**
 - *Fantastic for latency* / unknown data characteristics
 - *Scaling considerations*

Complex systems

- Not all processes fit within a single processing type
- Many real-world scenarios may require **multiple components** to build the best processing model
- Concepts can be applied to **various components** as required

Vaccination clinic

- **Multiple, simultaneous** moving pieces
- Vary based on **locale** and **requirements**
- Consider a **large self-contained** clinic
- Concepts apply to smaller pharmacies / doctor's offices as well

Vaccination clinic areas

- **Arrival / entrance**
 - Entry and temperature check, with a single line
- **Registration**
 - Check-in & validation on info, multiple registrars
- **Vaccine administration**
 - Actual application of vaccine, multiple stations
- **Monitoring**
 - Patients checked for any post-application reactions, many seats
- **Departure**
 - Exit from clinic

Let's practice!
STREAMING CONCEPTS

Congratulations!

STREAMING CONCEPTS



Mike Metzger
Data Engineer

Next steps

- Learn more about **specific streaming platforms**
 - Apache Kafka ([Apache Kafka](#)) or ([Confluent](#))
 - Apache Spark ([Apache Spark](#))
- **Apply** current data implementations to stream processes
- Work with data consumers to better **determine best processing options for a given situation**

Thank you!
STREAMING CONCEPTS