Intro to event-based computing

STREAMING CONCEPTS

Mike Metzger
Data Engineer

Old days

- **Shared** access
- **Batched** jobs
- Programs **run by operators** and results returned to users
- Delays, missing results, etc

Personal computers

- Often **single user**
- But still behaved in a **batch** manner
- Computer would basically run tasks **in order** as provided
- GUI gave rise to **event-based interactivity**

Event-based processing

- ***Doesn't*** **run at a specific time**
- Tasks run when an **event occurs**
    - User clicks a button
    - A new file is uploaded to a directory
- Can still **start a batch process**
- Event-based systems **wait for something to occur**

Example event-based task

Web click-stream monitoring

- User activity occurs when **clicking on links / components** of a webpage
- The **client application determines** what resources are needed and requests these from a server
- The **server returns** the appropriate info and often logs the request
- These clicks (user events) are often **stored** or sent to a central location for storage and **later analysis.**

Let's practice!

STREAMING CONCEPTS

Queuing

STREAMING CONCEPTS

Mike Metzger
Data Engineer

What is queuing?

- Basically, a **line**
- Useful for processing in **order**
- First-in, first-out (**FIFO**)
- Sometimes referred to as a **buffer**
- Details **vary** a lot **by implementation**

[1] Photo by Joshua Tsu on Unsplash

Why queues?

- Queues allow **tracking** of processing order
- Can be processed by a **single person / program or multiple**
- Can be **disconnected** from the remainder of the processing pipeline
- Reasonably **easy to scale** vertically or horizontally
    - Vertical scaling by adding faster hardware
    - Horizontal scaling by adding more executors

Queue issues

- **Bad data** or processing errors
  - *Customer pays with invalid credit card*
- Data size **variances**
  - *Supermarket fast lane with 100 items*
- Sometimes difficult to **know the length** of the queue
  - *First preview showing of a movie*
- Scaling **limits**
  - *Not enough space for more registers*

Let's practice!

STREAMING CONCEPTS

Single system data streaming

STREAMING CONCEPTS

Mike Metzger
Data Engineer

Intro to streaming

What is streaming?

- Data **doesn't stop** until processed
  - Once initially processed, may have other data processing components
- Is **open-ended** (no specific end event)
- Is **defined by the flow** of data, **not the content**

Logs

- **Store** event information
- Could be a simple **text** or **binary** file
- Or a system to export information to multiple clients (ie, Apache Kafka)
- Will store information **until resources are exhausted / pruned**
- Purpose of the log **depends on** the **application**

```
210507-162356 - SUCCESS: Open vvlj45.txt
210507-162254 - ERROR: Open hjry57.txt failed
210507-161523 - SUCCESS: Open kbhn78.txt
210507-161235 - ERROR: Open ldge12.txt failed
210507-160127 - WARNING: keop98.txt exists
210507-155958 - SUCCESS: Open hqaz64.txt
210507-155439 - SUCCESS: Open neuf36.txt
210507-152335 - SUCCESS: Open mqpa91.txt
210507-144756 - ERROR: Open pqzi32.txt failed
210507-143541 - SUCCESS: Open urmn15.txt
210507-143152 - SUCCESS: Open fgty82.txt
210507-141732 - SUCCESS: Open mlwe96.txt
```

System event log

- Present on **Windows, Mac, Linux**
- Processes and stores various **system event information**
- Windows **EventLog**, Mac / Linux **syslog**

Components:

- **Listener**: *Accepts* messages
- **Parser:** Understands how to *read* messages
- **Logic**: *Decides* what to do
- **Writer**: *Stores* the messages for later

Let's practice!

STREAMING CONCEPTS

Batching vs streaming

STREAMING CONCEPTS

Mike Metzger
Data Engineer

Quick review

- Batch processes handle data in **groups**, or batches
- The most important details about batch processing is the batch **size**, and the batch **frequency**
- Queues store / process data **in order** of insertion
- **Queues are batches**, with a batch size of one!
- Streams handle data **without pausing** along the way
- Streams **don't have a defined end**
- Streams **maintain order**!

Fire!

- **Bucket brigade**
  - Batch size (**how large** is the bucket)
  - Batch frequency (**how fast** to pass bucket)

- **Fire hose**
  - **Continuous** amount of data
  - Not sure **how much** water

[1] Albert B. Kinne, Public domain, via Wikimedia Commons [2] Commander, U.S. Naval Forces Europe-Africa/U.S. 6th Fleet, Public domain, via Wikimedia Commons

How to determine the best approach?

- Depends on **requirements**
- If we can process in **groups**, **batching** often best due to simplicity
- If we need **order**, but it's okay to pause, use a **queue**
- If we need **continuous** data, or we don't know how much data, try **streaming**
- If we **can't stop** until the data is processed, use **streaming**

Let's practice!

STREAMING CONCEPTS