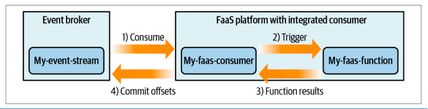
Adam Bellemare - Building Event-Driven Microservices

9. Microservices Using Function-as-a-Service (“serverless”)

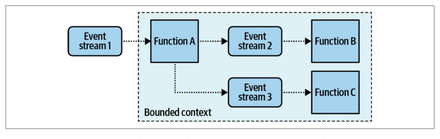
- Designing Function-Based Solutions as Microservices
  - Commit Offsets Only After Processing Has Completed
    - If you should not lose data
  - Less (number of functions) Is More
    - Many functions can lead to highly fragmented solution that is difficult to support
- Choosing a FaaS Provider
  - Open source
    - OpenWhisk
    - OpenFaaS
    - Kubeless
    - Apache Pulsar
  - 3rd party
    - Amazon Web Services (AWS)
    - Google Cloud Platform (GCP)
    - Microsoft Azure
- Building Microservices Out of Functions
  - 4 main components to consider

Function	Event stream(s)	Trigger	Policies and metadata
`myFunction`	`myInputStream`	`onNewEvent`	`<...>`

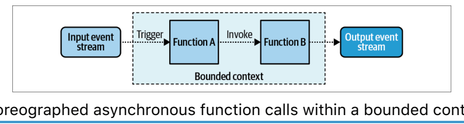
    - The function
    - Input event stream
    - Triggering logic
    - Error and scaling policies, with metadata
- Cold Start and Warm Starts
  - Warm function are put into hibernation and then quickly reused (warm start)
- Starting Functions with Triggers



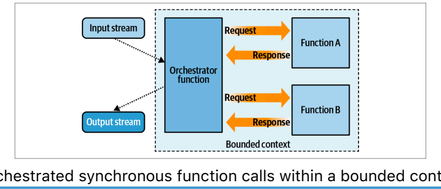
  - Triggering Based on New Events: The Event-Stream Listener
    - Batch size
      - the maximum number of events to dispatch for processing
    - Batch window
      - the maximum amount of time to wait for additional events, instead of triggering the function immediately
  - Triggering Based on Consumer Group Lag
    - the function does not consume the events until after it is started
      - Function is establishing a client connection with the event broker, consuming the events, and committing back any offset updates
  - Triggering on a Schedule
    - scheduled to start up periodically and at specific datetimes
  - Triggering Using Webhooks
  - Triggering on Resource Events
    - For instance, creating, updating, or deleting a file in a filesystem can trigger functions, as can the same modifications made to a row in a data store
- Performing Business Work with Functions
  - FaaS are highly scalable
  - FaaS should be used with great care when concurrency and determinism are important!
- Maintaining State
  - This greatly reduces the flexibility of FaaS providers
    - Function can have connection to external state store
- Functions Calling Other Functions
  - Event-Driven Communication Pattern



  - Direct-Call Pattern
    - Choreographed asynchronous function calls within a bounded context



    - Orchestrated synchronous function calls within a bounded context



- Termination and Shutdown
  - once it has completed its work
  - or it reaches the end of its allocated lifespan, generally in the range of 5–10 minutes
- Tuning Your Functions
  - Allocating Sufficient Resources
  - Batch Event-Processing Parameters
- Scaling Your FaaS Solutions
  - Be careful about thrashing triggers and scaling policy. Frequent rebalancing of partition assignments can be expensive for event brokers