# Event-driven Architecture: Orchestrating Cloud Native Workflows

Md. Mostafa Al Mahmud

AWS Community Builder (Serverless)

Software Engineer, Brain Station 23 PLC

# Md. Mostafa Al Mahmud

https://mdmostafa.com/

https://www.linkedin.com/in/md-mostafa/

https://twitter.com/md__mostafa


Software Engineer @ Brain Station 23 PLC

# Md. Mostafa Al Mahmud

https://mdmostafa.com/

https://www.linkedin.com/in/md-mostafa/

https://twitter.com/md__mostafa


Worked in the projects of some valuable brands

SUZUKI

ROGERS

Build With Innovation

Synspective

incepta

rehive

BRAIN STATION 23

HandyMama

# Today's Agenda

- Event-driven architecture overview and key concepts
- Core benefits of EDA and why to use it
- Common usage patterns in EDA
- Orchestration vs Choreography
- Considerations while building an EDA
- Best practices
- Demo - Error Handling
- Helpful resources

# EDA - Powering Agility and Scalability

- Events are inclusive in modern applications
- EDA drives innovation across industries
- A central point to modern applications
- Communication between services and development teams
- Many comprehensive toolkits availability

# Core Values of Event-Driven Architectures

- Independent feature build and development
- Enhanced Scalability and Resilience
- Effortless feature integration in the current app
- Loose coupling and modularity
- Asynchronous processing

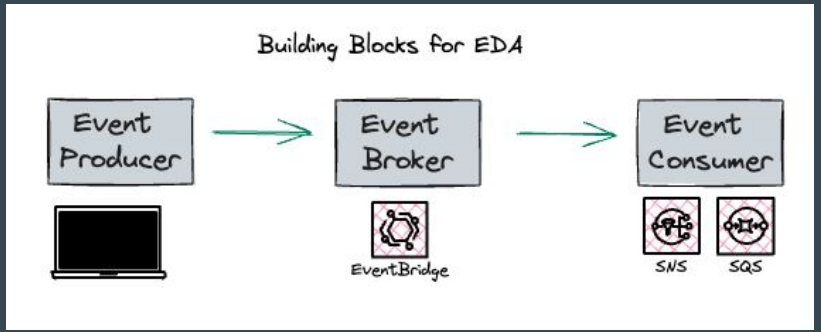
# Why to Use EDA

- Real-time updates
- Effortless scaling
- Targeted communication
- Simplified integration
- Security and privacy
- Minimize resource utilization

# Key Concepts of Event-Driven Architectures

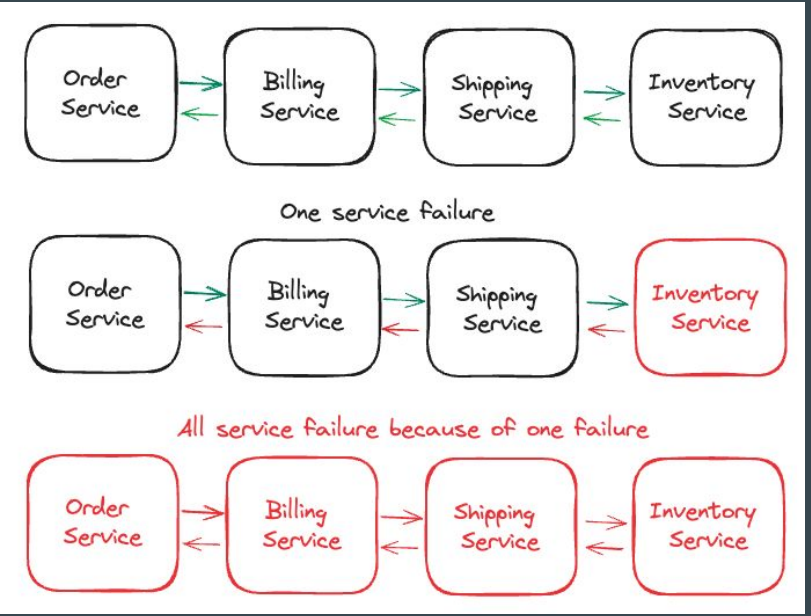
**There are several building blocks**

- Events: Signals of state changes
- Immutable and observable
- Producers: Publish events
- Consumers: Downstream components that react to events
- Event Brokers: Mediate communication
    - Event Routers: Push events to targets
    - Event Stores: Consumers pull events



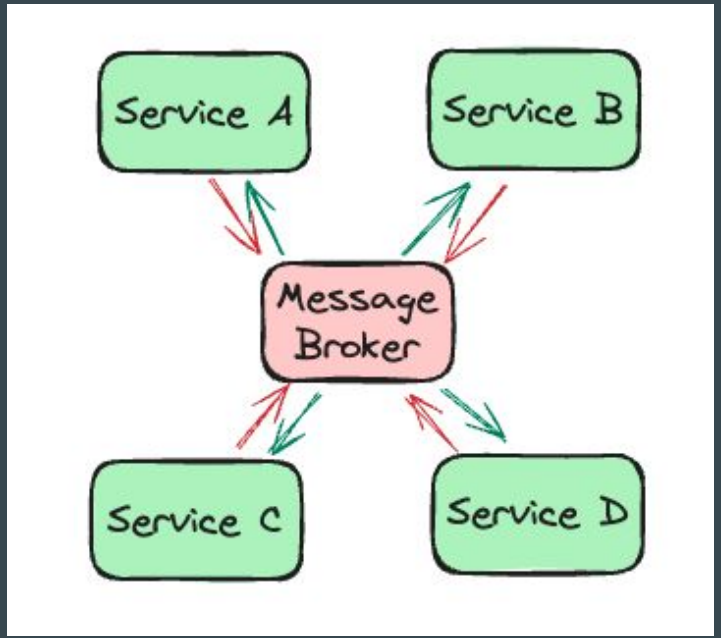
# Tight Coupling vs. Loose Coupling

- Drawbacks of tight coupling
  - Development Challenges
  - Scalability issues
  - And so on
- E-commerce Example: Tightly coupled services (orders, billing, shipping, inventory) create a fragile flow



# Tight Coupling vs. Loose Coupling

- Loose coupling: the power of **Events**



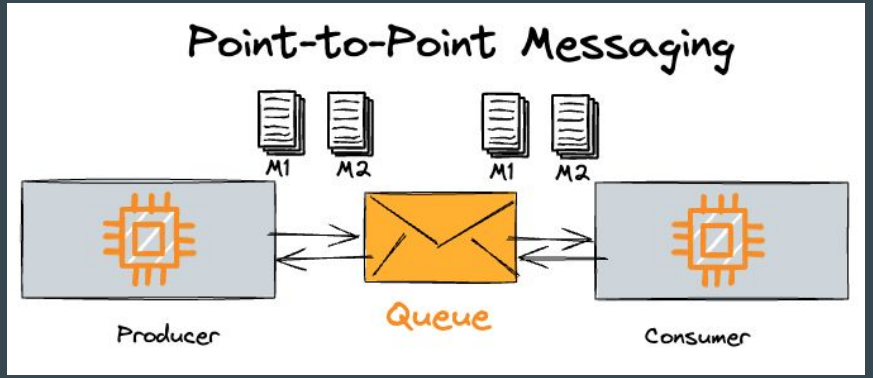
# Idempotency

- Have no unintended consequences upon retries
- Crucial concept for EDA due to inherent retry mechanisms
- Techniques for achieving idempotency:
  - Building idempotent services.
  - Utilizing idempotency keys in events.

# Common Patterns of EDA

## Point-to-Point Messaging

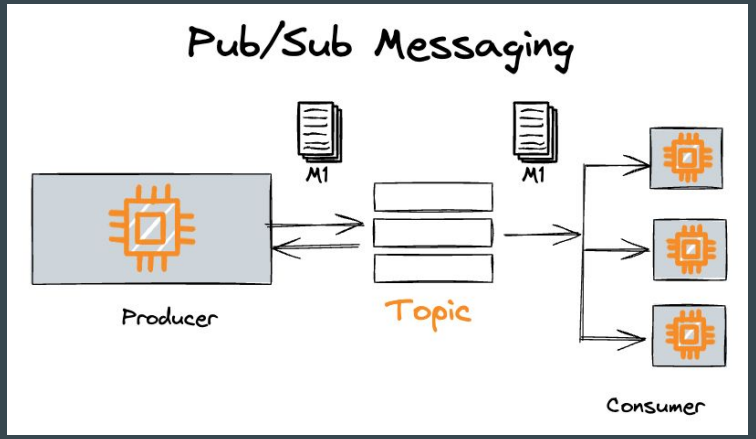
- Producers send messages to a single consumer.
- Messaging queues serve as event brokers.
- Messages persist until consumed, ensuring reliability.
- "Dumb pipes" in microservices communication

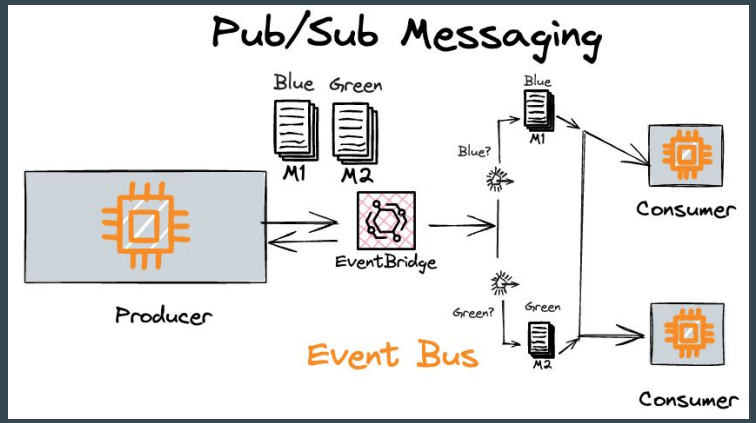


# Common Patterns of EDA

## Publish-Subscribe Messaging

- Producers send the same message to one or many consumers.
- Utilizes event routers instead of queues.
- Generally lacks event persistence.
- Event bus (another event router type)

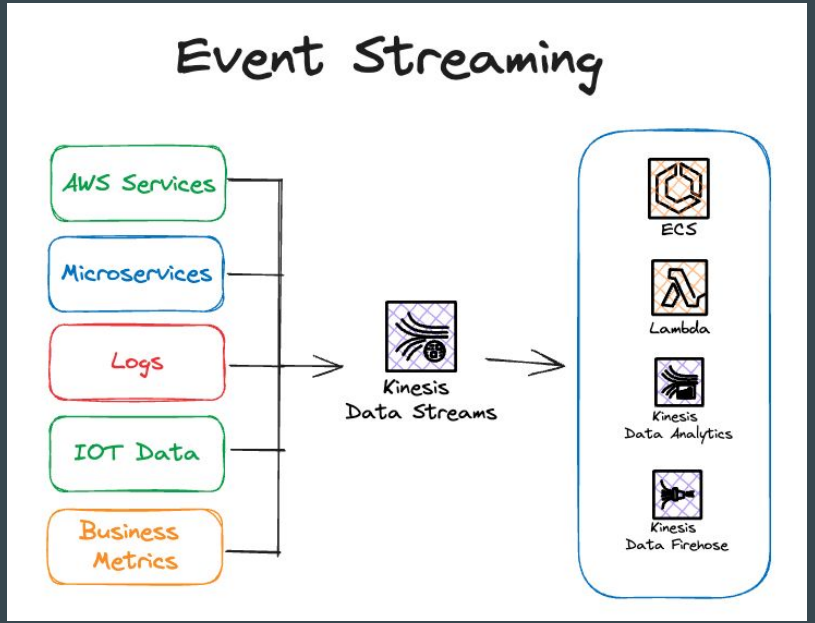


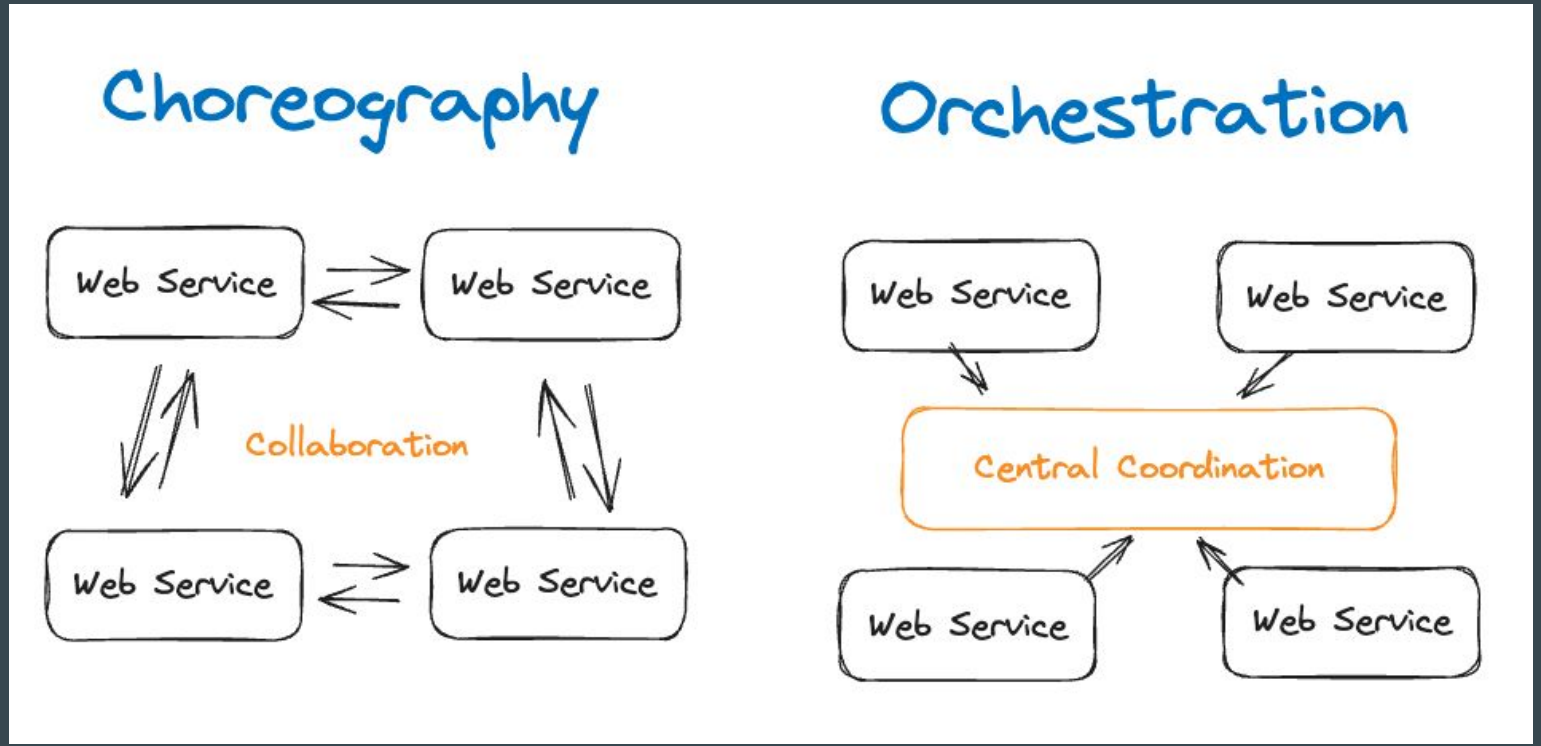
# Common Patterns of EDA

## Event streaming

- Continuous flows of events or data
- Consumers typically poll for new events
- Event streams can be processed individually or collectively over time.
- Data streams interpret data over time, often used for real-time analytics or data persistence.



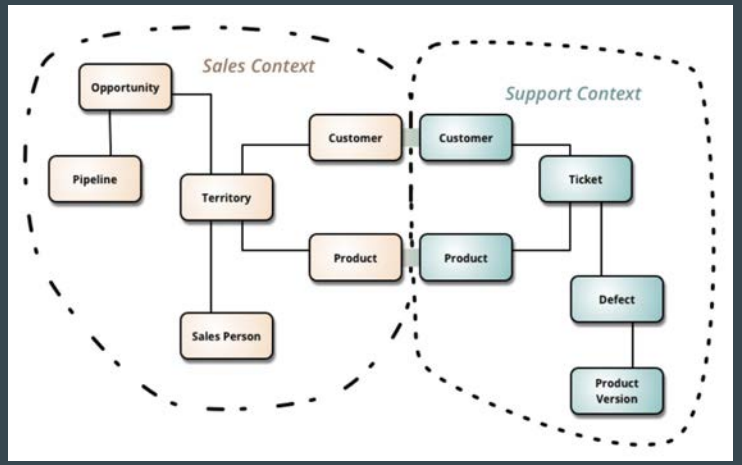
# Common Patterns of EDA



# Common Patterns of EDA

## Choreography

- Ideal for communication between Bounded Contexts.
- Producers focus on event delivery (fire-and-forget).
- Event schema ensures message clarity.
- Reduces dependencies between contexts (loose coupling).
- Use EventBridge as event bus

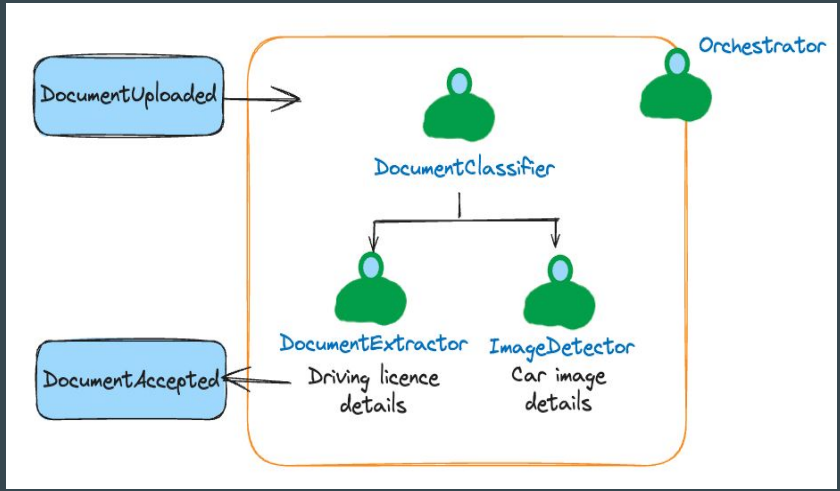



https://martinfowler.com/bliki/BoundedContext.html

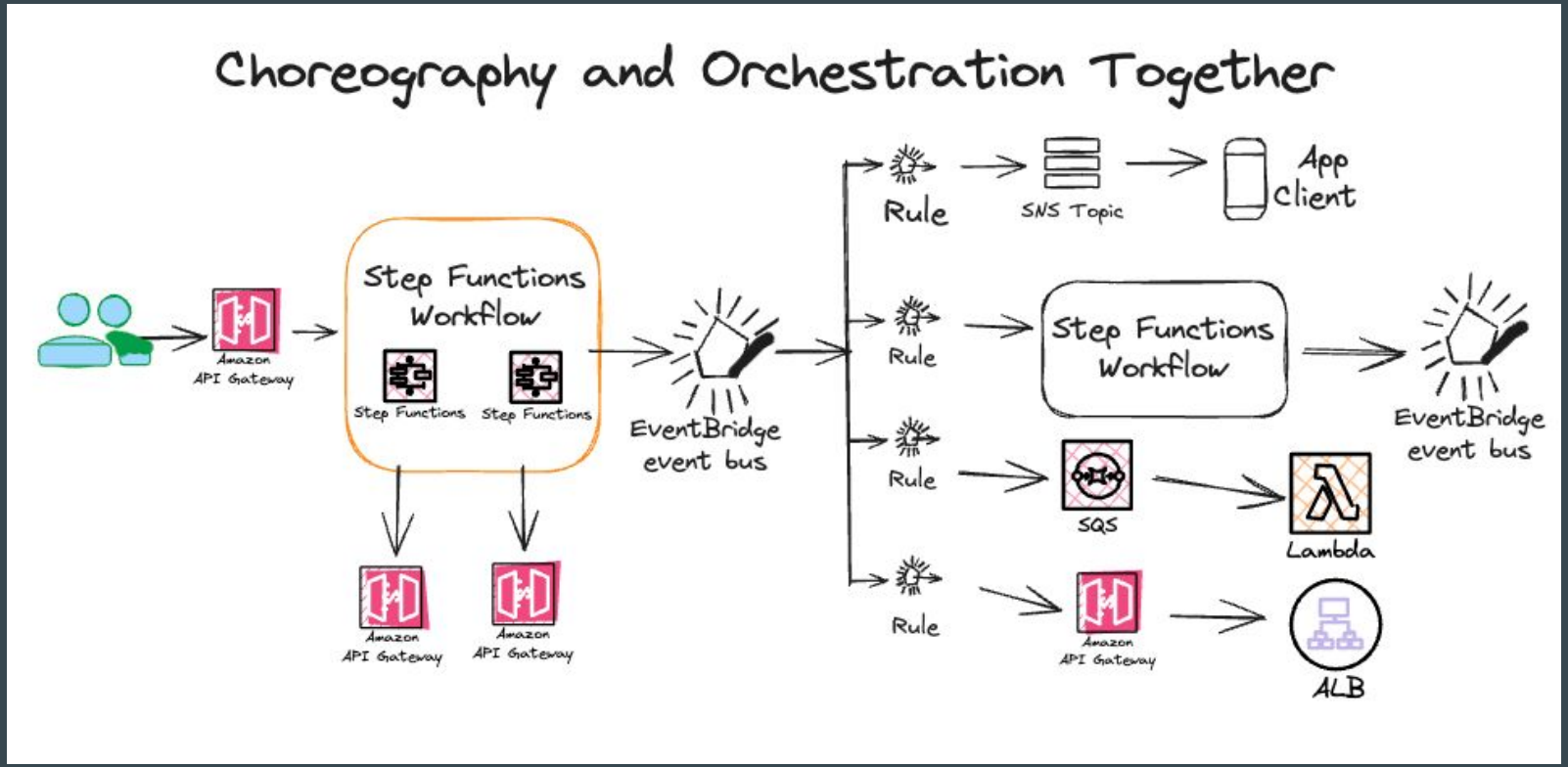
# Common Patterns of EDA

## Orchestration

- Manages service integration sequence
- Maintains application state for complex workflows
- Handles errors and retries
- Use AWS Step Functions, Amazon MWAA

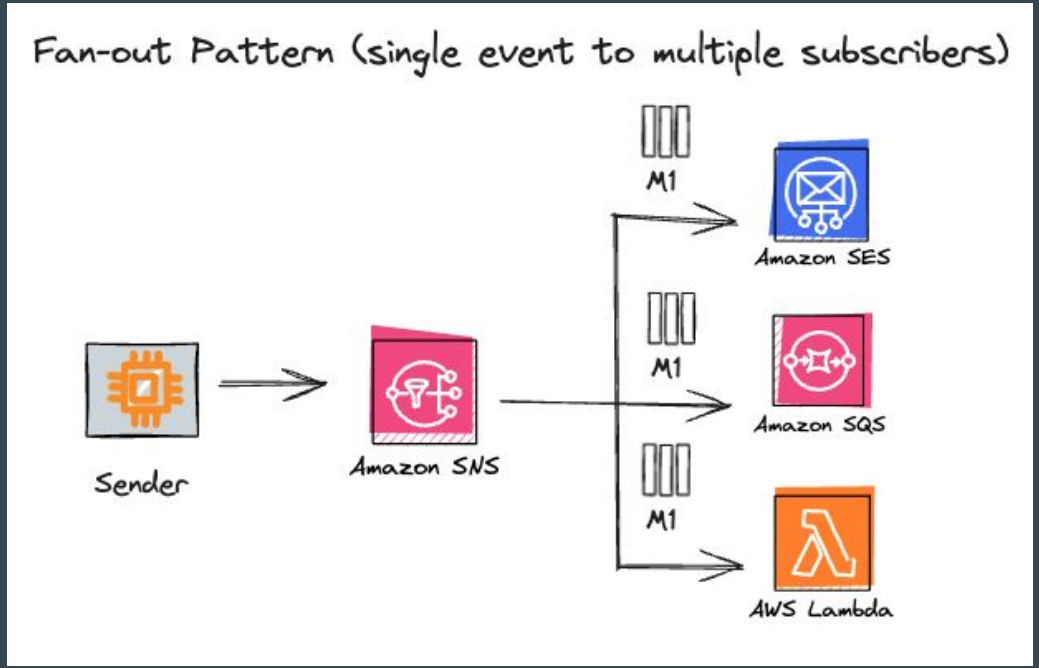


# When to Use Choreography and Orchestration Together



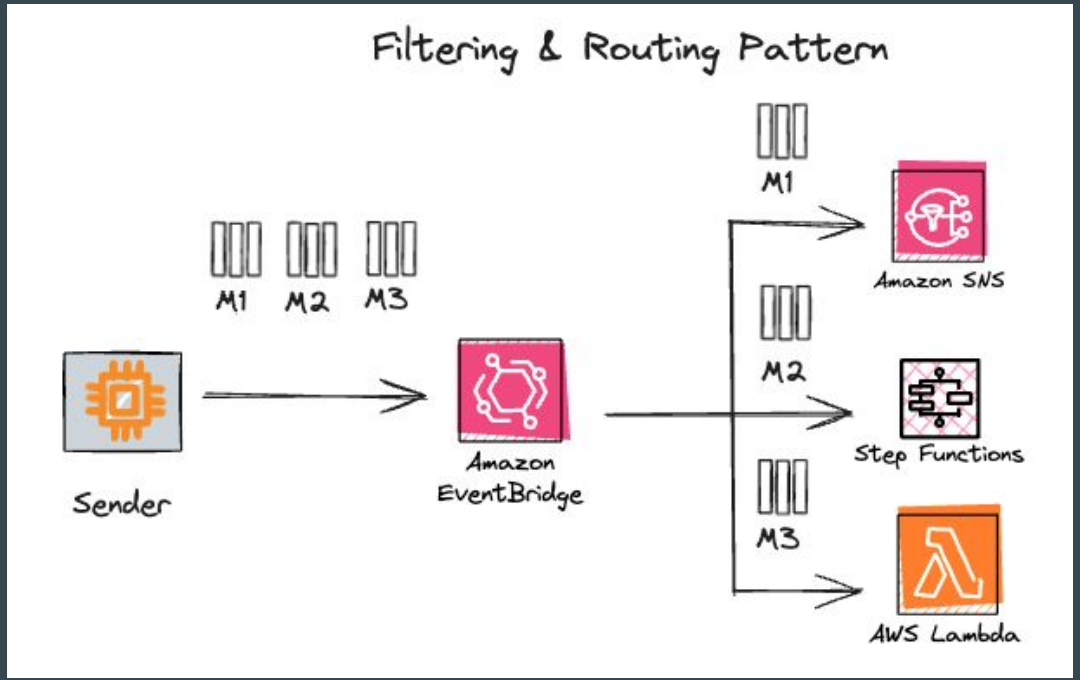
# When to Use Combining Patterns

**Fan-out: Distributing a single event to multiple subscribers**



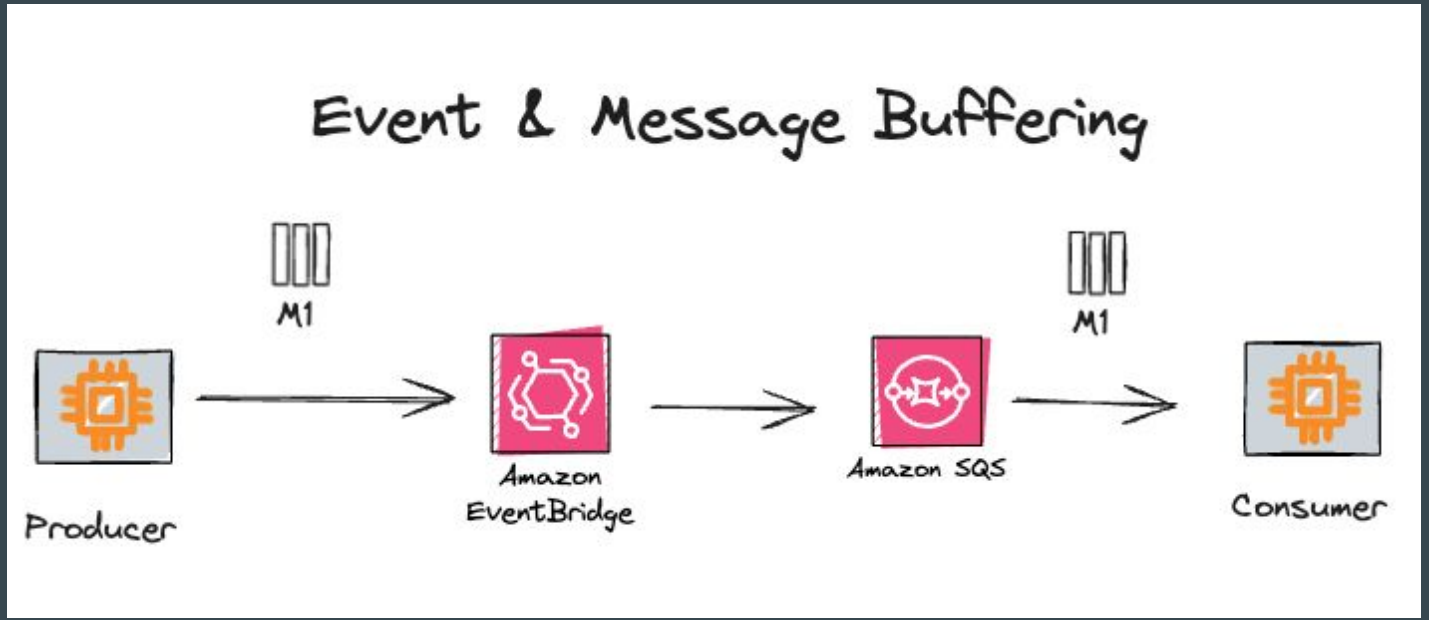
# When to Use Combining Patterns

**Event Filtering & Routing: Directing events to specific targets**



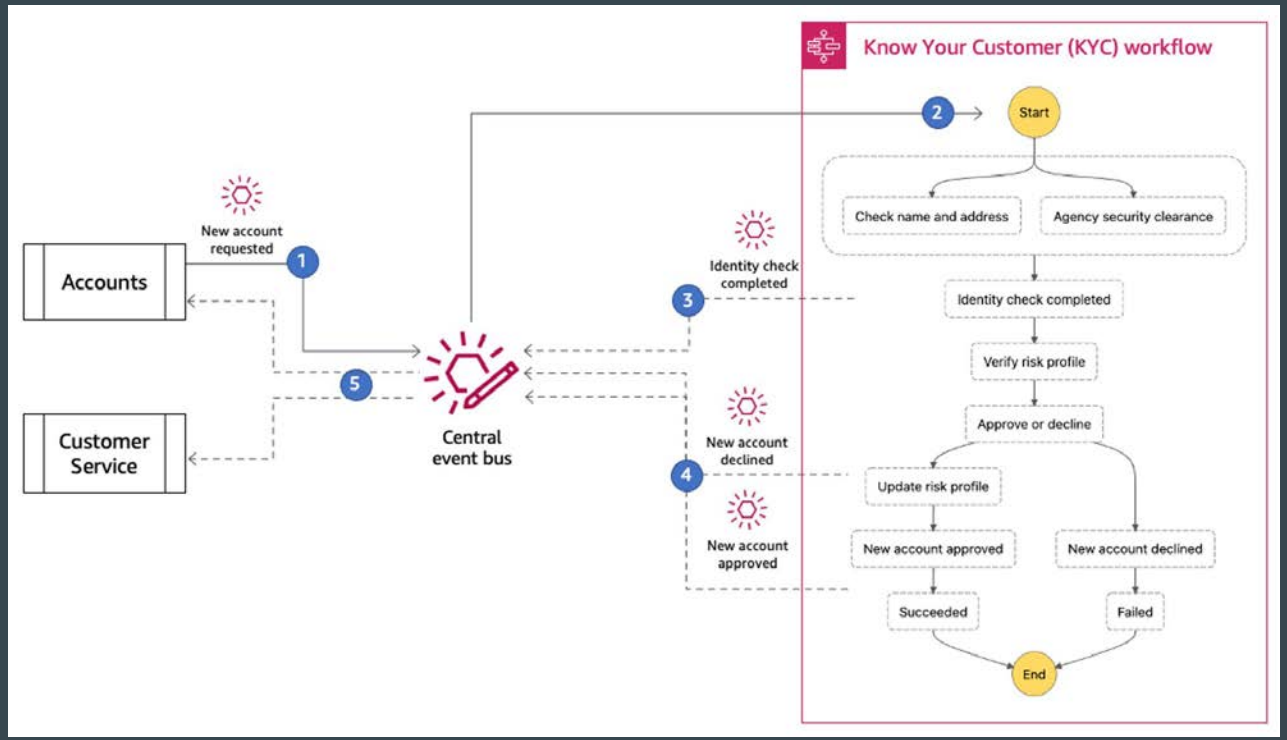
# When to Use Combining Patterns

## Event and Message Buffering



# When to Use Combining Patterns

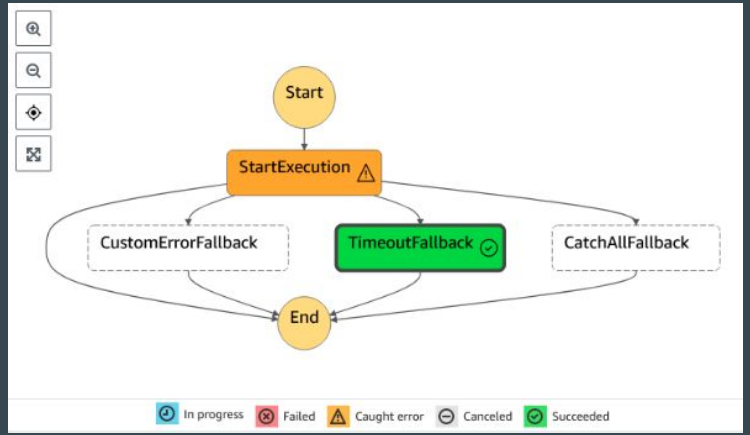
## Workflow Orchestration




https://aws.amazon.com/blogs/compute/introducing-the-amazon-eventbridge-service-integration-for-aws-step-functions/

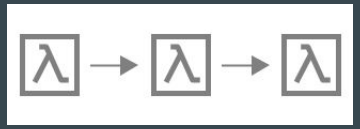
# Step Functions

- Serverless orchestration service for EDA workflows
- Visually define workflows
- Workflow components:
    - State machines: entire workflows
    - States: individual steps
    - Task states: utilize other AWS services (e.g., Lambda functions) to perform tasks
- Standard vs. Express workflows

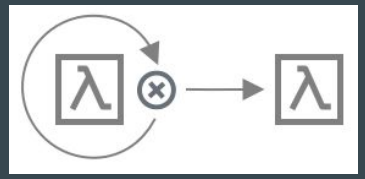



https://catalog.us-east-1.prod.workshops.aws/workshops/9e0368c0-8c49-4bec-a210-8480b51a34ac/en-US/development/error-handling/step-4

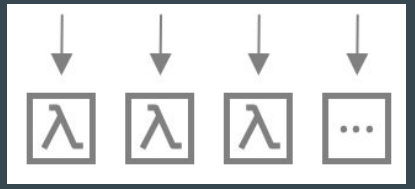
# Step Functions - Use Cases



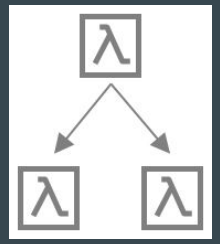
Function orchestration



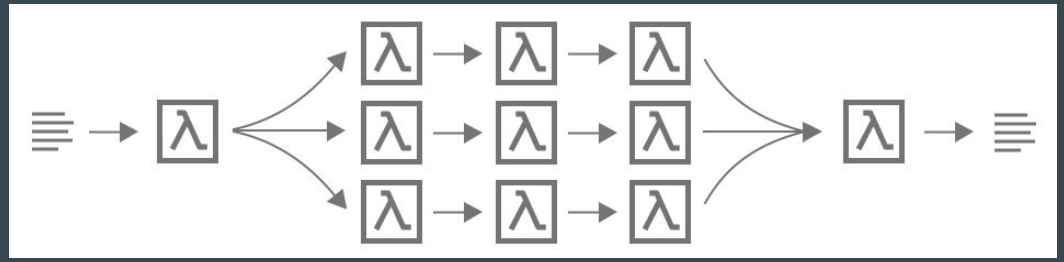
Error handling



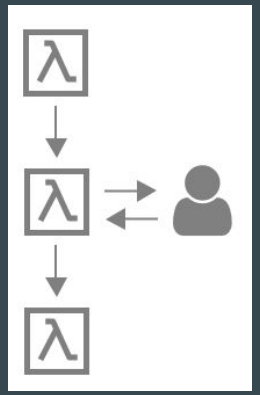
Parallel processing



Branching



Dynamic Parallelism



Human in the loop

https://docs.aws.amazon.com/step-functions/latest/dg/welcome.html

# Demo - Error Handling

# Best Practices of EDA

- Event identifications with event storming
- Naming convention
- ECST events
- Notification events
- Conformist pattern
- ACL
- OHS
- Event-first thinking
- Idempotency
- Ordering (order/unorder events)

# Resources

- https://aws.amazon.com/blogs/compute/introducing-the-amazon-eventbridge-service-integration-for-aws-step-functions/
- https://docs.aws.amazon.com/eventbridge/latest/userguide/eb-what-is-how-it-works-concepts.html
- https://d1.awsstatic.com/SMB/aws-modernization-intro-to-eda-guide-2022-smb-build-websites-and-apps-resource.pdf
- https://aws.amazon.com/event-driven-architecture/
- https://theburningmonk.com/2020/08/choreography-vs-orchestration-in-the-land-of-serverless/
- https://docs.aws.amazon.com/step-functions/latest/dg/welcome.html
- https://catalog.us-east-1.prod.workshops.aws/workshops/9e0368c0-8c49-4bec-a210-8480b51a34ac/en-US/development/error-handling
- https://d1.awsstatic.com/psc-digital/2023/gc-300/build-eda-on-aws/Build-EDA-AWS-eBook.pdf

# Be connected with me


# Thank you

AWS community builders