AWS Step Functions — Concepts, Architecture & Interview Q&A; Guide

A Light■Theme Illustrated PDF covering core concepts, architecture, JSON examples, best practices, and 25+ interview questions for cloud engineers.

■ Core Concepts

• AWS Step Functions is a serverless orchestration service to build workflows (state machines) that coordinate multiple AWS services or tasks.

• Workflows are defined using Amazon States Language (ASL), a JSON-based declarative language.

• It supports two workflow types:

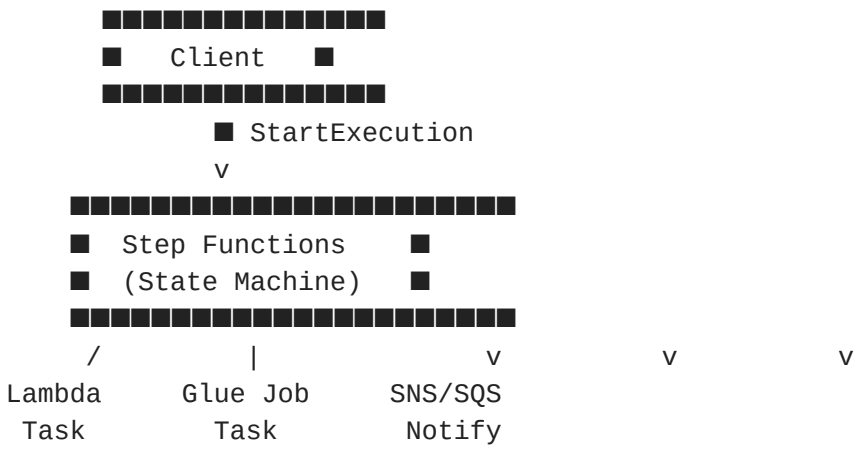
– Standard Workflows: Durable, exactly-once, up to 1-year execution, ideal for long-running, auditable workflows.

– Express Workflows: Short-duration (<5 minutes), high-throughput, at-least-once execution, optimized for streaming/event processing.

• Common use-cases: ETL pipelines, ML orchestration, microservice coordination, human approval flows, data pipeline automation.

• Step Functions handle retries, error handling, parallel execution, wait/delay logic, and visual monitoring out of the box.

■■ Simplified Orchestration Flow



■ Example: Amazon States Language (ASL) Workflow

This example defines a state machine that invokes a Lambda function, handles errors, and proceeds to a success state.

```
{
  "Comment": "Example AWS Step Function Workflow",
  "StartAt": "InvokeLambda",
  "States": {
    "InvokeLambda": {
      "Type": "Task",
      "Resource": "arn:aws:lambda:us-east-1:123456789012:function:processData",
      "Retry": [{
        "ErrorEquals": ["Lambda.ServiceException", "Lambda.AWSLambdaException"],
        "IntervalSeconds": 3,
        "MaxAttempts": 2,
        "BackoffRate": 2.0
      }],
      "Catch": [{
        "ErrorEquals": ["States.ALL"],
        "Next": "FailureState"
      }],
      "Next": "SuccessState"
    },
    "SuccessState": {
      "Type": "Succeed"
    },
    "FailureState": {
      "Type": "Fail",
      "Cause": "Lambda invocation failed"
    }
  }
}
```

■ Human Approval Example using WaitForTaskToken

```
{
  "StartAt": "WaitForApproval",
  "States": {
    "WaitForApproval": {
      "Type": "Task",
      "Resource": "arn:aws:states:::lambda:invoke.waitForTaskToken",
      "Parameters": {
        "FunctionName": "approvalHandler",
        "Payload": {
          "token.$": "$$.Task.Token",
          "input.$": "$"
        }
      },
      "Next": "Approved"
    },
    "Approved": {
      "Type": "Succeed"
    }
```

```
    }
}
```

■ AWS Step Functions Interview Q&A;

Q: What is AWS Step Functions?

A: A serverless orchestration service that coordinates multiple AWS services using state machines.

Q: What are workflow types?

A: Standard (durable, exactly-once) and Express (fast, at-least-once).

Q: What is a State Machine?

A: A JSON-defined workflow using Amazon States Language that defines states, transitions, and error handling.

Q: What are common State types?

A: Task, Choice, Wait, Parallel, Map, Pass, Succeed, Fail.

Q: How does Step Functions integrate with AWS services?

A: Via service integrations using SDK integrations or Resource ARNs (e.g., Lambda, ECS, Glue, SNS).

Q: What is Amazon States Language (ASL)?

A: A JSON-based declarative language defining the workflow structure and state transitions.

Q: How are errors handled?

A: With Retry and Catch fields per state for retries and alternate paths.

Q: What is WaitForTaskToken pattern?

A: Used to pause execution until an external callback is received, ideal for human approvals.

Q: Difference between Parallel and Map?

A: Parallel runs multiple branches concurrently; Map iterates over array items executing sub-workflows.

Q: How can you monitor workflows?

A: Using AWS Console visualization, CloudWatch metrics/logs, and AWS X-Ray for tracing.

Q: What are Express workflows suited for?

A: High-volume, short-lived, event-driven workflows such as streaming or IoT pipelines.

Q: What are Standard workflows suited for?

A: Long-running, auditable, durable workflows such as data pipelines or approvals.

Q: How does data pass between states?

A: Via JSON objects using InputPath, OutputPath, ResultPath, and Parameters fields.

Q: What are service integration patterns?

A: Request-Response, Run a Job (.sync), Wait for Callback (.waitForTaskToken).

Q: Can Step Functions call external APIs?

A: Yes, through AWS SDK integrations or custom Lambda functions.

Q: How to include retries?

A: Define a Retry field with IntervalSeconds, MaxAttempts, and BackoffRate for exponential retries.

Q: How does error catching work?

A: Use Catch field with ErrorEquals list and Next transition to handle exceptions gracefully.

Q: How do you secure Step Functions?

A: Control IAM roles, limit service access, encrypt input/output data, use VPC integration for secure tasks.

Q: When to use Step Functions vs Lambda directly?

A: Use Step Functions for complex orchestration, multiple dependencies, retries, and state management.

Q: Cost model difference between Standard vs Express?

A: Standard is billed per state transition; Express charges per request + execution duration.

Q: How to deploy Step Functions?

A: Use AWS Console, SAM templates, or CloudFormation; version state machines for CI/CD.

Q: How to handle human approval?

A: Use WaitForTaskToken with callback or external API trigger to resume workflow.

Q: What is the input/output JSON size limit?

A: 256KB per state execution input/output.

Q: Can Step Functions orchestrate Glue and Batch?

A: Yes, using service integration ARNs for Glue jobs or Batch job definitions.

Q: What’s a good real-world use-case?

A: Data pipeline orchestration: start Glue job, process Lambda, then send SNS notification.

■ Best Practices

• Use Standard workflows for long-running, auditable orchestrations.

• Use Express for high-volume, fast, low-latency processes.

• Implement retries and exponential backoff for transient failures.

• Keep input/output payloads small (under 256KB).

• Use WaitForTaskToken for external/human-in-the-loop processes.

• Log execution details and metrics to CloudWatch.

• Use version control (SAM/CloudFormation) for state machine deployment.

• Apply least privilege IAM roles for Step Functions and invoked services.

■ Standard vs Express Workflow Comparison

Feature	Standard Workflow	Express Workflow
Execution Type	Durable, exactly-once	Fast, at-least-once
Duration	Up to 1 year	Up to 5 minutes
Throughput	Limited to thousands/sec	High (100K/sec)
Cost Model	Per state transition	Per request + duration
Use Cases	Data pipelines, approvals	Streaming, IoT, event-driven