AWS re:Invent

API321

Event-processing workflows at scale with AWS Step Functions

Sam Dengler
Principal Serverless SA
Amazon Web Services

Shawn Myron
Principal Product Manager
Amazon Web Services

AWS re:Invent

aws

Agenda

Related breakouts

What is AWS Step Functions

Introducing AWS Step Functions Express Workflows

AWS Step Functions Express Workflows demo

Related breakouts

API305 Building serverless machine-learning workflows

API316 Building serverless workflows using AWS Step Functions

API318 Deep dive on event-driven development with Amazon EventBridge

---

API306 Building event-driven architectures

API311 Managing business processes using AWS Step Functions

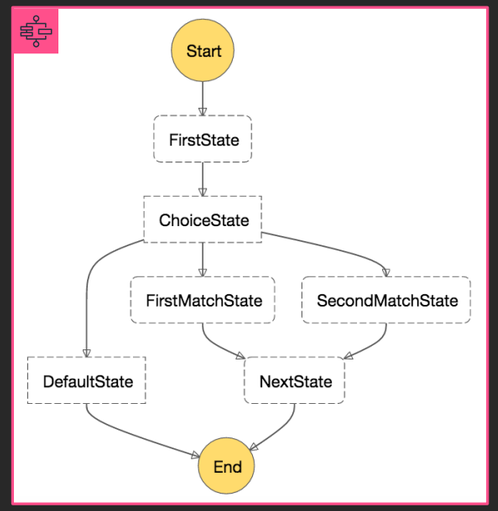
AWS Step Functions 101

AWS Step Functions: Visual workflows

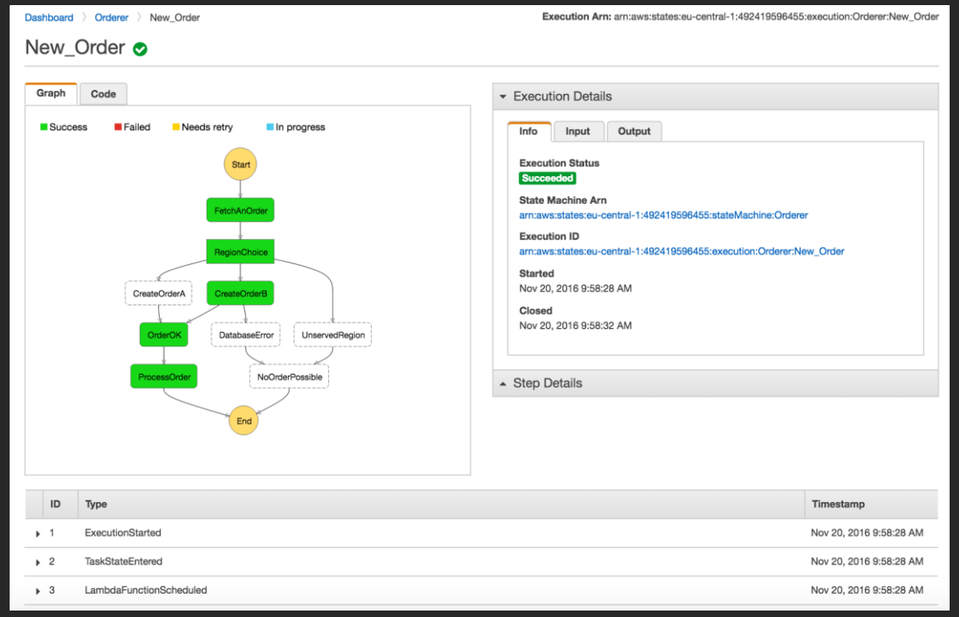
Define in JSON

```
{
  "StartAt": "Generate dataset",
  "States": {
    "Generate dataset": {
      "Resource": "arn:aws:lambda:us-east-
1:ACCOUNT:function:SAMPLE",
      "Type": "Task",
      "Next": "Train model (XGBoost)"
    },
    "Train model (XGBoost)": {
      "Resource":
"arn:aws:states:::sagemaker:createTrainingJob.sync",
      "Parameters": {
        "AlgorithmSpecification": {
```

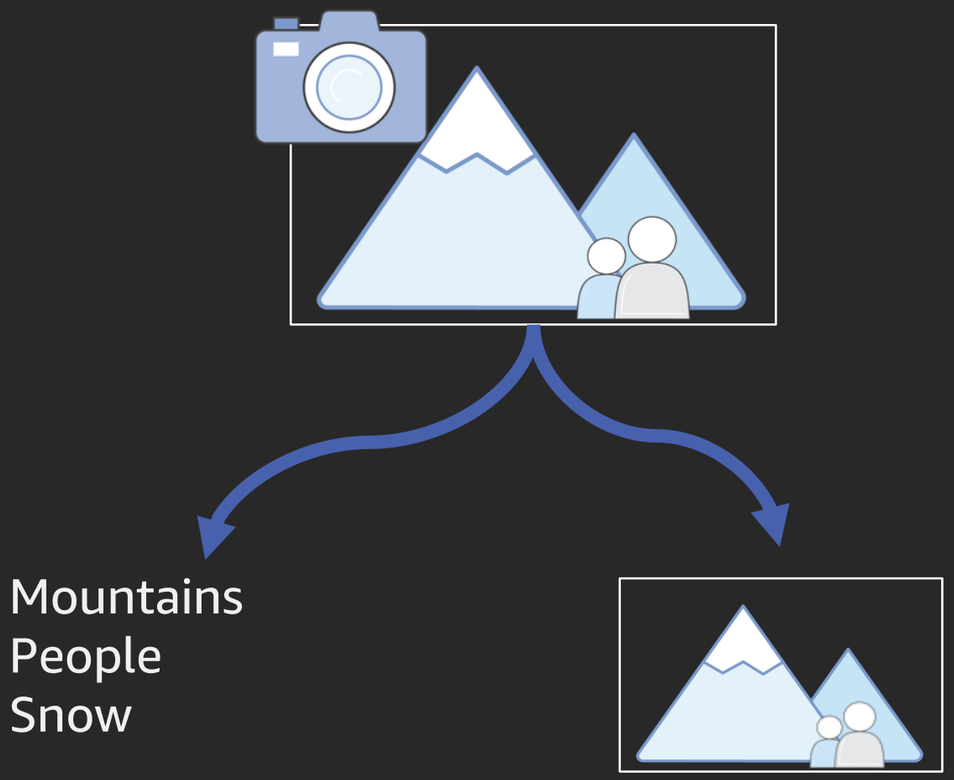
Visualize in the console

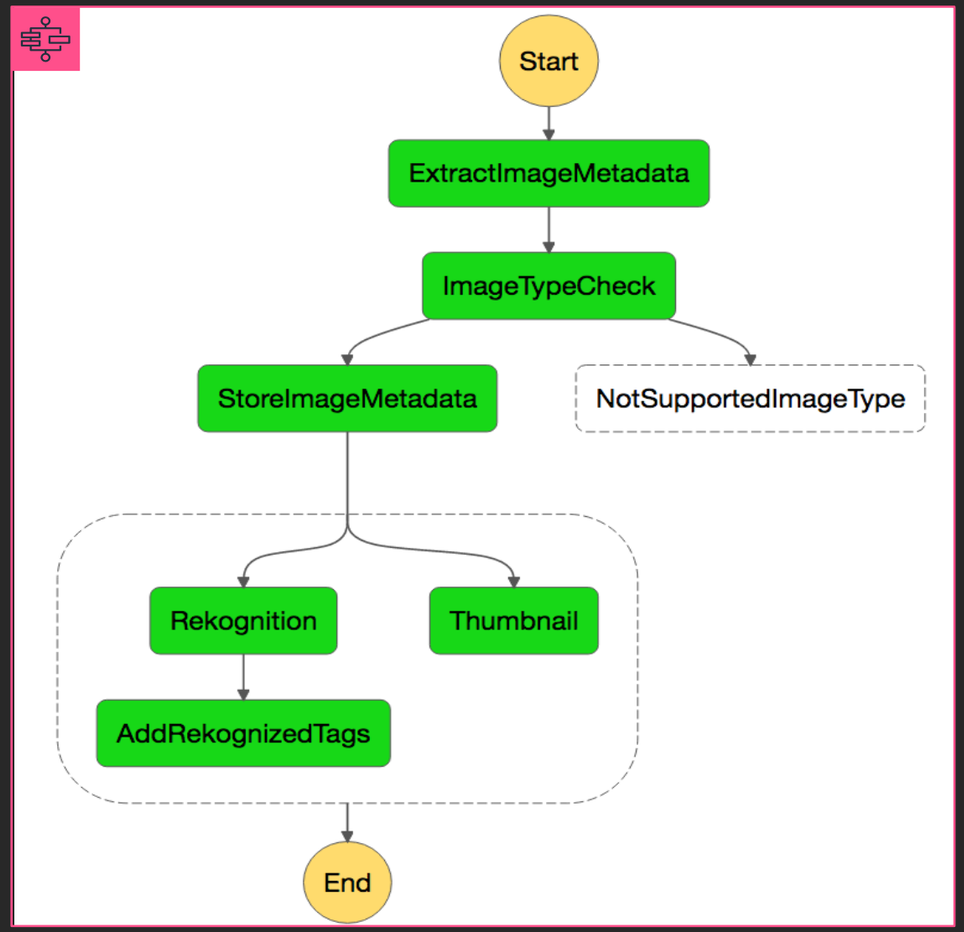


Monitor executions



Build workflows using service integrations





Serverless workflows on Step Functions

Build and update apps quickly

Step Functions lets you build visual workflows that enable fast translation of business requirements into technical requirements. You can build applications in a matter of minutes, and when needs change, you can swap or reorganize components without customizing any code.

Write less code

Step Functions manages the logic of your application for you and implements basic primitives such as branching, parallel execution, and timeouts. This removes extra code that may be repeated in your microservices and functions.

Improve resiliency

Step Functions manages state, checkpoints, and restarts for you to make sure that your application executes in order and as expected. Built-in try/catch, retry, and rollback capabilities deal with errors and exceptions automatically.

Granular

Challenge

- Improve scaling and management of legacy application that performs machine data translation through ingestion and processing from large-scale agricultural machinery

Results

- Easy integration between serverless and legacy application components
- Scalable approach to running code through multiple technology stacks
- Self-documenting executions with Step Functions execution history

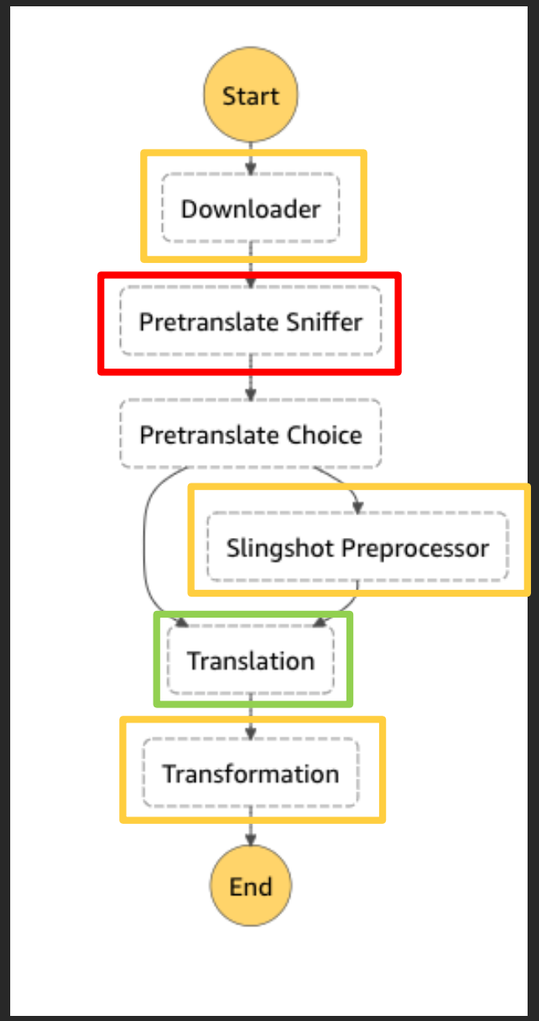
Patterns

- Step Functions to bridge from legacy to modern application development

Granular

How they did it

- Considered writing their own workflow engine but discounted it, as they wanted to focus on their business logic
- Used Step Functions because the activity API could easily combine Amazon EC2, Kubernetes, and AWS Lambda to create a modern application that is inclusive of useful legacy components



Python on containers on Amazon EC2

Python Lambda function

C# and on Windows Server (Amazon EC2)

Introducing:
AWS Step Functions Express Workflows

AWS re:Invent

aws

Introducing:

AWS Step Functions Express Workflows

Orchestrate AWS compute, database, and messaging services at rates up to 100,000 events per second, suitable for high-volume event processing workloads such as IoT data ingestion, microservices orchestration, and streaming data processing and transformation

Faster: greater than 100K state transitions per second

Designed for short-duration workflows: < 5 mins.

Cost effective at scale

Standard vs. Express Workflows

	Standard	Express
Console	Step Functions	Step Functions
State machine definition	Amazon States Language	Amazon States Language
Documentation	Step Functions	Step Functions

Standard vs. Express Workflows

	Standard	Express
Maximum duration	365 days	5 minutes
Start execution refill rate	300 per second	6,000 per second
State transition refill rate	1,300 per second	None
Execution semantics	Exactly-once workflow step execution	At-least-once workflow step execution

Standard vs. Express Workflows

	Standard	Express
Executions	Executions are persisted and have ARNs	Executions are not persisted except as log data
Execution history	Stored in Step Functions, with tooling for visual debugging in the console	Sent to Amazon CloudWatch Logs
Service integrations	Supports all service integrations and activities	Supports all service integrations. Does not support activities.
Patterns	Supports all patterns	Does not support Job-run (.sync) or Callback (.wait For Callback)

Demo

Thank you!

Sam Dengler
fsd@amazon.com

Shawn Myron
myrshawn@amazon.com

AWS re:Invent

aws

Please complete the session survey in the mobile app.