AWS IoT Analytics

Pipelines

AWS IoT Analytics Mini-User Guide

Introduction

As we connect more and more IoT devices, our data grows exponentially. We need to process that data to gain insights and intelligence quickly and efficiently. But, gaining insights from the IoT data can be quite difficult.

With Pipelines—part of the fully-managed IoT analytics service—you can process noisy messages from IoT devices into useful data.

Handle heterogeneous messages generated by IoT devices, even if they greatly vary in terms of schema structure and unit of data. Manage messages with erroneous or blank values for the fields by assuring these messages will not contribute towards your goal. Process messages to create a normalized repository for efficient querying. We know that IoT data by itself isn't sufficient enough. Pipelines offer a simple mechanism for cleaning, transforming and enriching IoT messages.
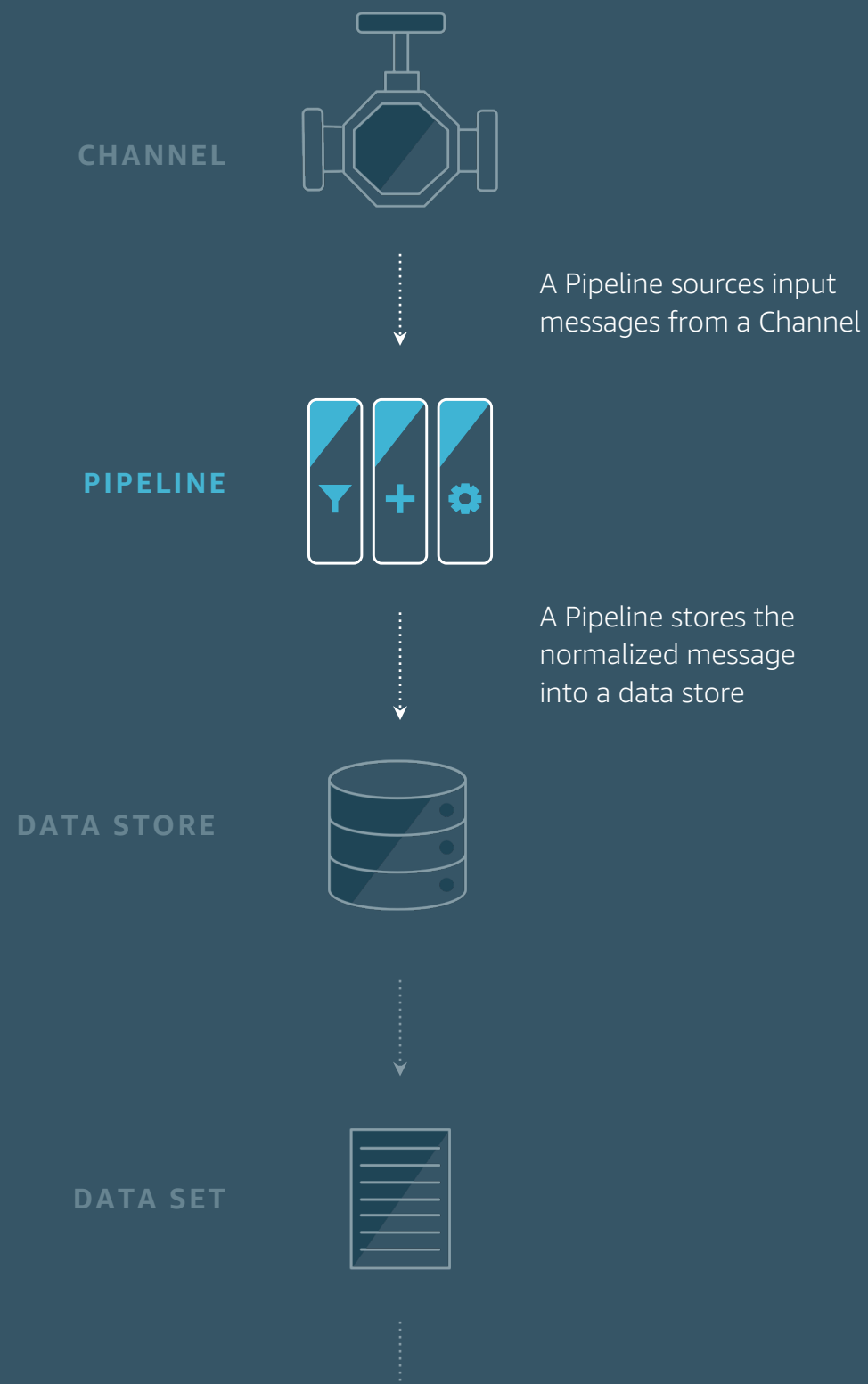
Adding value to your business

A Pipeline is essential for processing your IoT data because the noisiness of real-world IoT data, combined with the need to enrich the incoming data with time, location and other relevant information creates a challenge for application developers. We wanted to make this easier.

Pipelines can perform data transformations, execute conditional statements and enrich messages with data from external sources. For example, you could combine weather data and mapping information to create better information about a device's environment.

Cleansing, enriching and transforming data typically requires custom code that takes time to build, test, maintain and adds processing cost to their IoT applications. AWS IoT Analytics does that coding for you, adding greater context to your IoT data so that you get deeper, more informed insights.

What is a Pipeline?

CHANNEL

A Pipeline sources input messages from a Channel

PIPELINE

A Pipeline stores the normalized message into a data store

DATA STORE

DATA SET

A Pipeline is a processing resource that allows you to perform a series of activities for cleaning, transforming and enriching IoT messages.

Pipelines consume unprocessed IoT messages from Channels and, through a series of ordered steps, transforms them into usable data. At each step, they perform some activity on the message and pass on that modified message to the next step.

Pipelines enable customers to easily transform raw messages into useful, queryable data. The linear progression of activities makes it simple to layer on additional data enrichments, correct for inaccuracies, or remove unnecessary messages.

Step by step

Pipelines are where you can access data preparation techniques that make it easy to prepare and process your data for analysis. Pipelines cleanse false readings, fills gaps in the data, and performs mathematical transformations of message data. As the data is ingested, IoT Analytics can process it using conditional statements, filter data to collect just the data you want to analyze and enrich it with information from the AWS IoT registry. You can also use AWS Lambda functions to enrich your device data from external sources like the Weather Service, HERE Maps, Salesforce, or Amazon DynamoDB. For example, you could combine weather data and mapping information to create better information about a device's environment.

The mechanics inside a Pipeline consist of these four steps:

Cleanse and filter: AWS IoT Analytics let you define AWS Lambda functions that can be triggers on when IoT Analytics detects missing data, so you can run code to estimate and fill gaps. You can also define max/min filters and percentile thresholds to remove outliers in your data.

Transform: AWS IoT Analytics can transform messages using mathematical or conditional logic you define, so you can perform common calculations like Celsius into Fahrenheit conversion.

Enrich: AWS IoT Analytics can enrich data with external data sources such as a weather forecast information, and then route the data to the IoT Analytics data store.

Re-Process: AWS IoT Analytics can re-process your data from the Channel connected to the Pipeline. Insights are often needed from the raw data rather than processed data. Simply connect the Pipeline to the appropriate Channel to re-process.

The diagram below depicts a high-level dataflow from ingest through analytics and visualizations. Notice the Pipeline section that includes Lambda. When your data is processed in a Pipeline, Lambda functions are used to cleanse and enrich your data. Once the data us through the Pipeline, it's stored in an an IoT optimized data store ready to be analyzed.
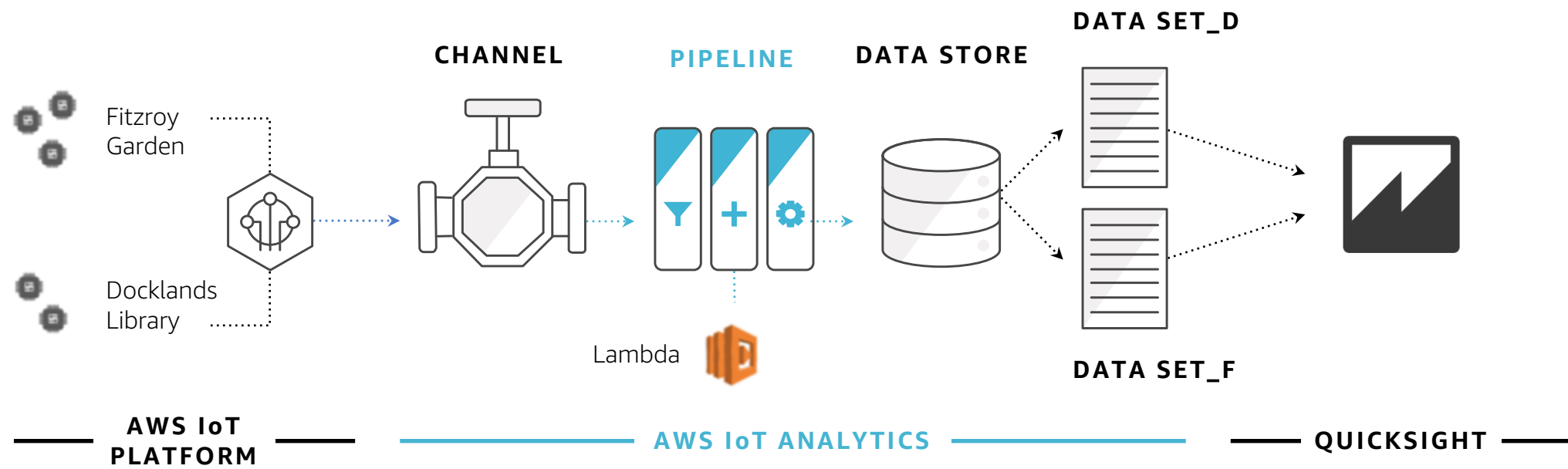
Essentially with this dataflow, the aim is to perform the following functions:

1. Publish Raw JSON messages to AWS IoT

2. Enrich each JSON message with the following, given the Australian ISO8601 timestamp
   (e.g. 2014-12-16 21:05:00.000)
   a. Epoch time stamp in UTC (1418677200)
   b. USA Pacific ISO8601 timestamp (12/15/2014 13:00:00.000)

3. Store the JSON messages in a single data store

4. Create data sets, querying the data store and classifying the data based on two sensor locations
   (Fitzroy Gardens and Docklands Library)

If you recall from the Channels user guide, this dataflow is not the only plausible solution to implement the above functions. There can be various other data flows that would still give the same result. The main goal here is that we want two data sets: one for each location, having appropriate timestamp.

How to get started

Four basic rules:

1. A Pipeline can source messages from only ONE Channel
2. A Pipeline can store its output to only ONE data store
3. Multiple Pipelines can source messages from ONE Channel
4. Multiple Pipelines can store the messages in ONE data store
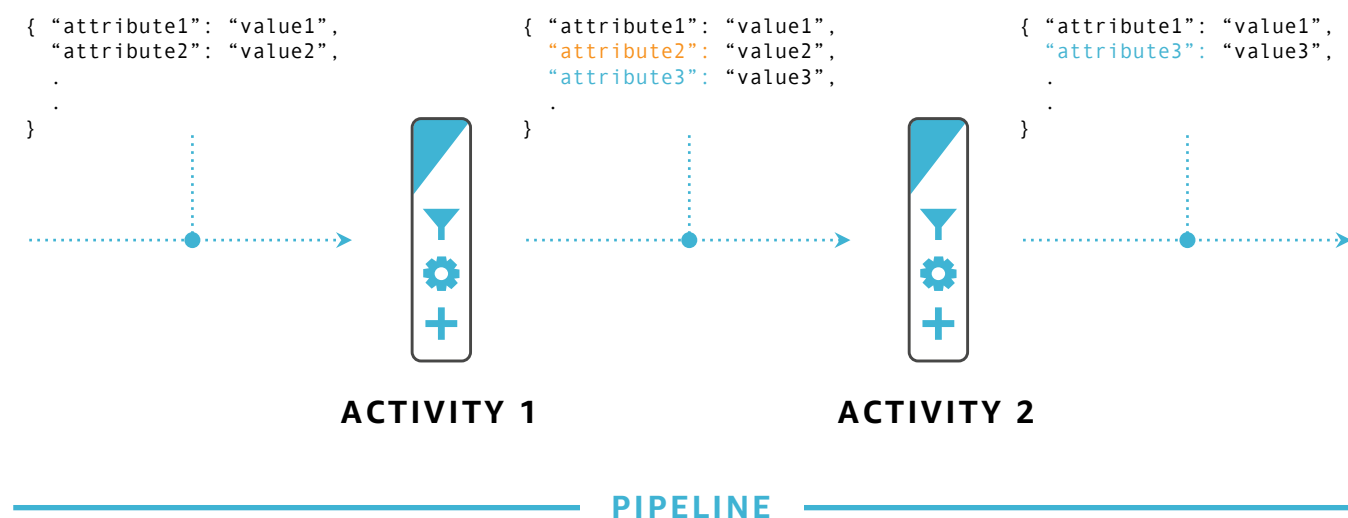
Prerequisites:

Before creating a Pipeline, you must have the following components created in the AWS IoT Analytics platform.

Channel to specify the source of the messages
Data store to specify the destination for the processed messages

Activities in Pipeline:

A Pipeline contains a series of steps for processing the messages. Each step is called an activity. Each activity receives an input message from a previous activity and passes on the processed message to the next activity.


```
{ "attribute1": "value1",
  "attribute2": "value2",
  .
  .
}
```

```
{ "attribute1": "value1",
  "attribute2": "value2",
  "attribute3": "value3",
  .
  .
}
```

```
{ "attribute1": "value1",
  "attribute3": "value3",
  .
  .
}
```

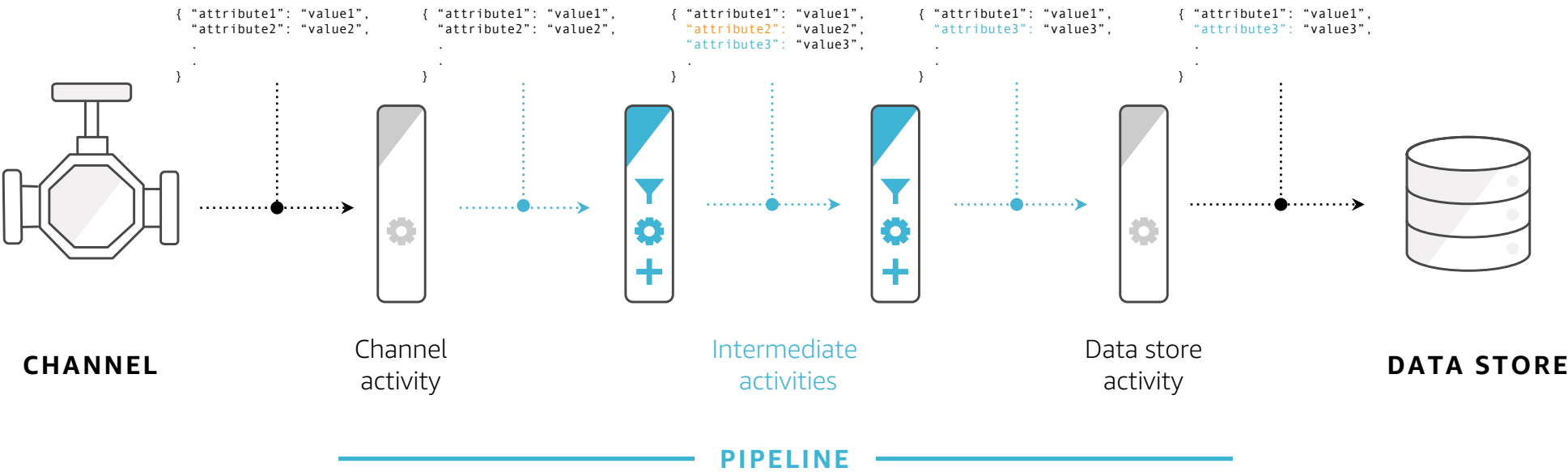**ACTIVITY 1**

ACTIVITY 2

PIPELINE

Each activity is designed for a specific task, such as filtering out messages, removing attributes or calculating new attributes. Current architecture can accommodate a maximum of 25 activities in a Pipeline.

The AWS IoT Analytics platform offers ten types of activities:

	Activity	Required	Description
1	Channel	Yes, at the beginning	This activity determines the source of the messages to be processed.
2	Data store	Yes, at the end	This activity specifies where to store the processed data.
3	Lambda	No	This activity is for invoking the AWS lambda functions. Quite useful for performing complex processing on messages.
4	AddAttributes	No	Adds attributes based on existing attributes in the message.
5	RemoveAttributes	No	Removes the attributes from messages.
6	SelectAttributes	No	Creates a new message by using the specified attributes from the incoming message, every other attribute is dropped.
7	Filter	No	Used for filtering out the messages based on its attributes.
8	DeviceRegistryEnrich	No	Allows you to add data from the AWS IoT Device Registry to your message payload.
9	DeviceShadowEnrich	No	Adds information from AWS IoT Thing Shadows Service to a message.
10	Math	No	A math activity computes an arithmetic expression using the message's attributes.

When expanded completely, the architecture might look like this:

CHANNEL Channel activity Intermediate activities Data store activity DATA STORE

PIPELINE

How to build a simple Pipeline

You can use the AWS IoT Analytics console, AWS CLI or AWS SDKs for creating Pipelines. Let's create a simple Pipeline through AWS CLI by following three simple steps.

Step 1: Define the order of activities as JSON payload. We will store this JSON payload as "my_Pipeline.json". Here, we will build the simplest functional Pipeline.

```
{
    "PipelineName": "myPipeline",
    "PipelineActivities": [
        {
            "Channel": {
                "name": "Channel_activity_1",
                "ChannelName": "myChannel",
                "next": "store_activity_1"
            }
        },
        {
            "Datastore": {
                "name": "store_activity_1",
                "DatastoreName": "myDatastore"
            }
        }
    ]
}
```

Step 2: Use the AWS CLI to create a Pipeline by using the JSON payload

```
aws iotanalytics create-Pipeline --cli-input-json file://my_Pipeline.json
```

Step 3: Record the ARN of the Pipeline as returned by the command

```
{
    "PipelineArn": "arn:aws:iotanalytics:us-west-2:xxxxyyyyzzzz:Pipeline/myPipeline",
    "PipelineName": "myPipeline"
}
```

Easily analyze data for deeper insights to make better, more accurate decisions for IoT applications and machine learning use cases. With AWS IoT Analytics, you can collect, pre-process, enrich, store and analyze your IoT data.

Start using Pipelines with AWS IoT Analytics in minutes:

aws.amazon.com/
iot-analytics