Seventh International Workshop on Serverless Computing (WoSC7) 2021
https://www.serverlesscomputing.org/wosc7/papers/p5

SFL: A Compiler for Generating Stateful AWS Lambda Serverless Applications

Lukas Brand
s-lbran1@haw-landshut.de
University of Applied Sciences Landshut
Landshut, Germany

Markus Mock
mock@haw-landshut.de
University of Applied Sciences Landshut
Landshut, Germany

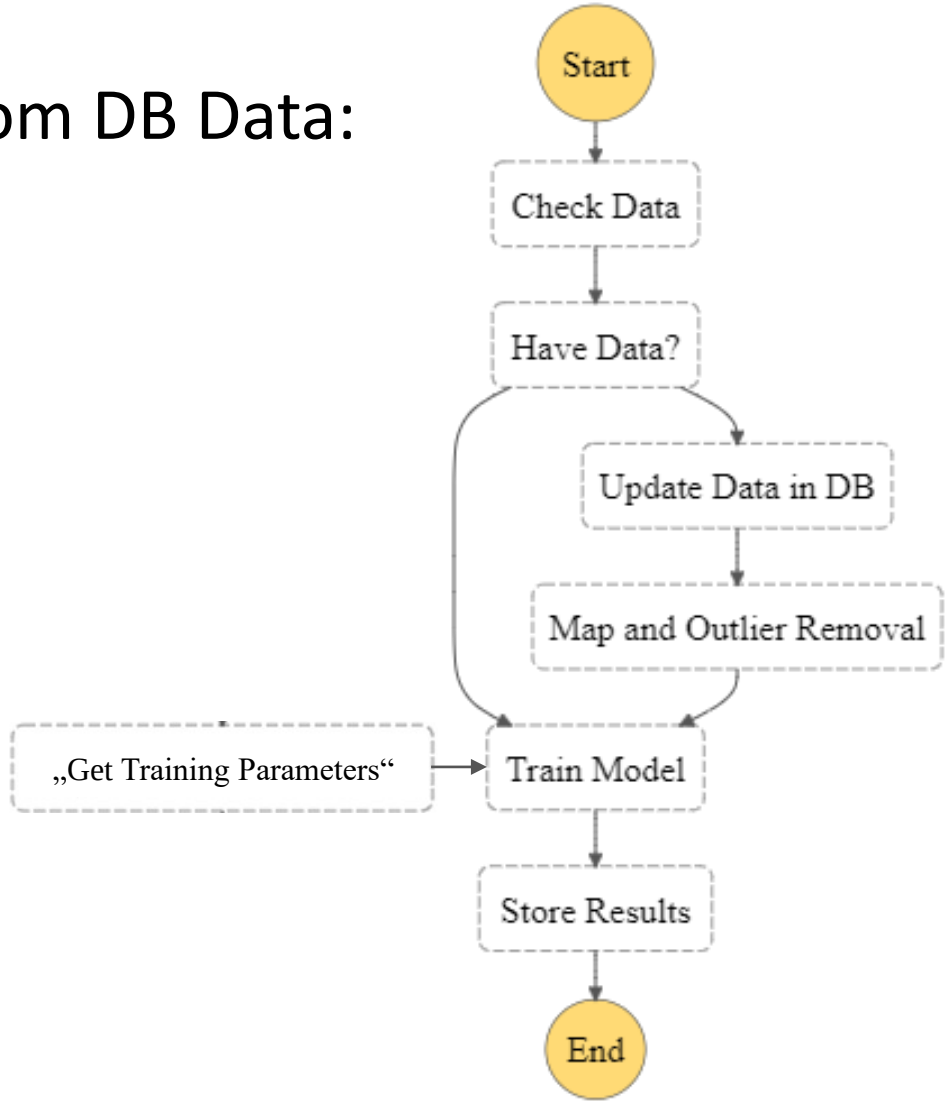
How can we allow for

- stateful -
- easy to develop -
- in an existing cloud (AWS)-

serverless applications?

Stateful Serverless Application Example

Training a Model from DB Data:

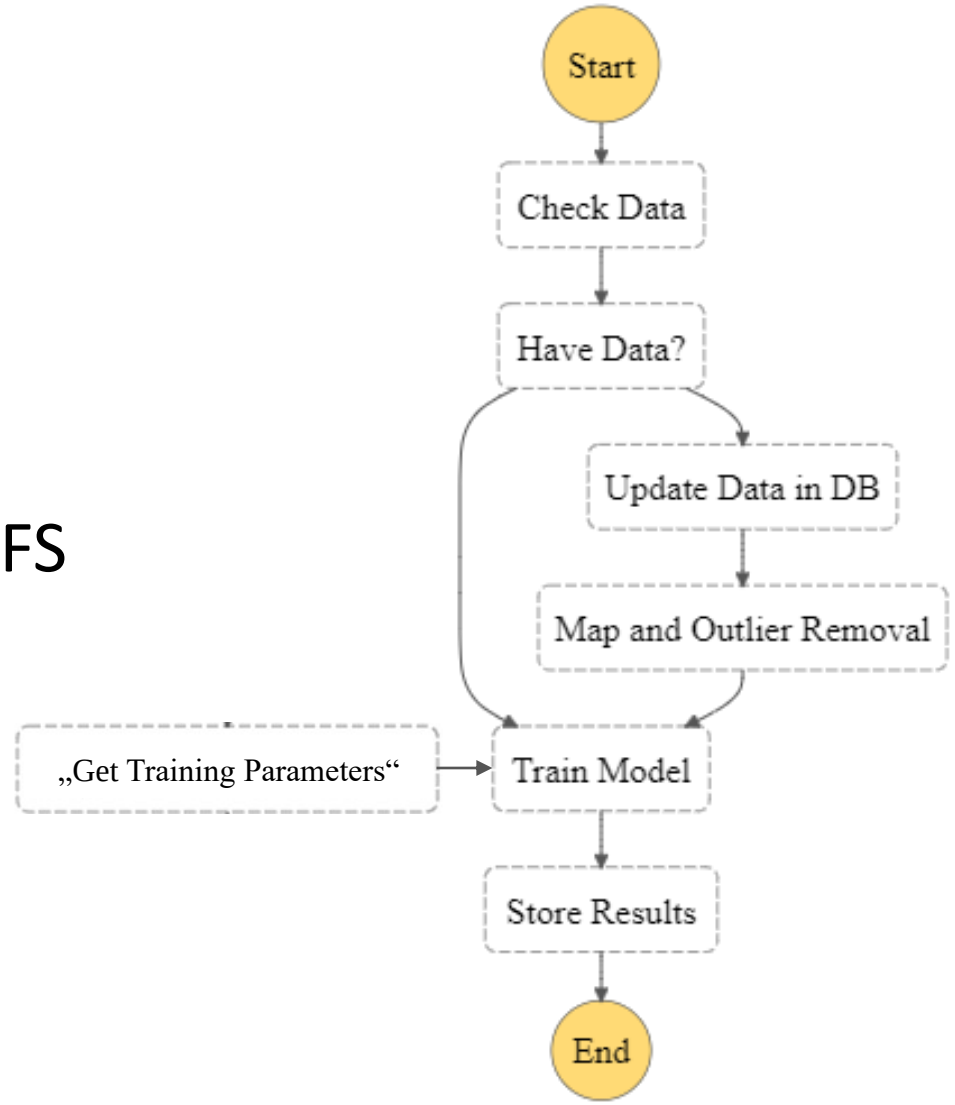


Stateful Serverless Application Example

Training a Model from DB Data:

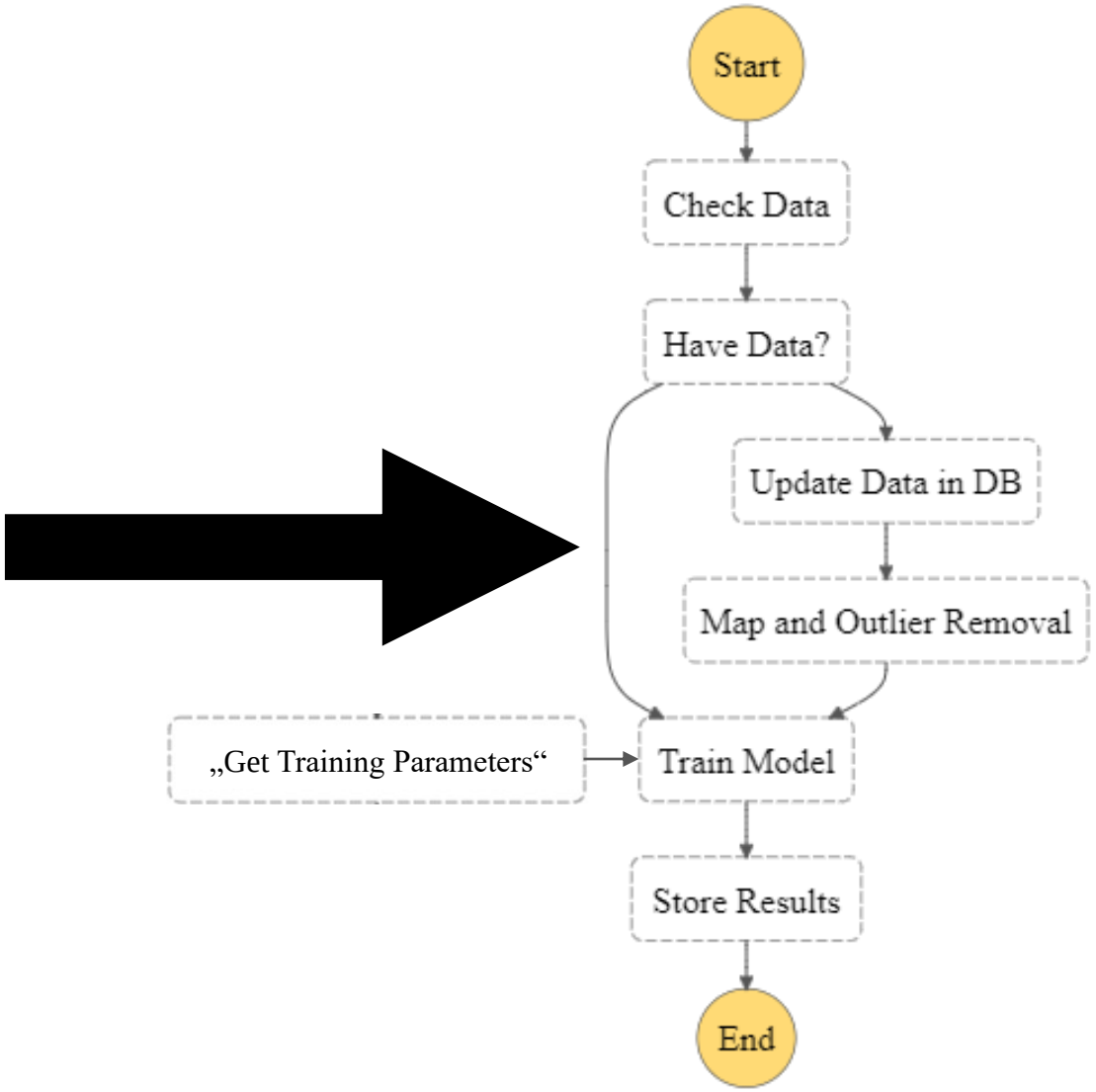
- five Lambda Functions
- Orchestration (Step Functions)
- Persistence in Orchestrator/DynamoDB/EFS
- Cloud Formation Template

=> Created by our Compiler



Stateful Serverless Application Example

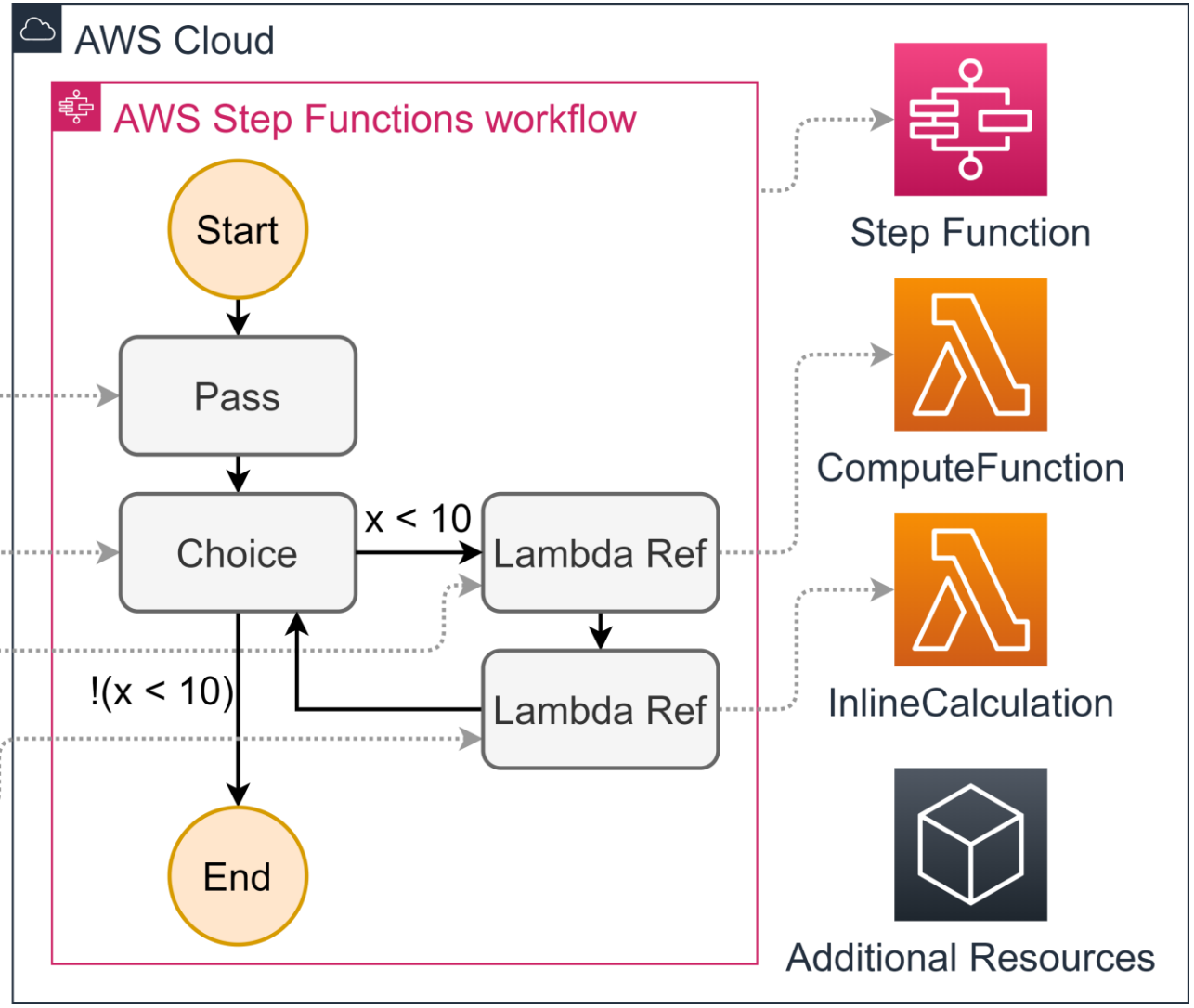
```
Statics {
    Boolean dataExists
    String databaseName
    global String trainingParameters
    String trainingResults
}
Functions {
    CheckData<python3_8>(dataExists) ~!{# ... }!~
    UpdateDataInDB<python3_8>(databaseName) ~!{# ... }!~
    MapAndOutlierRemoval<python3_8>(databaseName) ~!{# ... }!~
    TrainModel<python3_8>(trainingParameters, databaseName, trainingResults) ~!{# ... }!~
    StoreResults<python3_8>(trainingResults) ~!{# ... }!~
}
StateCode {
    Start() {
        dataExists = false
        databaseName = "ConfiguredDatabase"
        trainingResults = ""

        CheckData()
        if (dataExists == false) {
            UpdateDataInDB()
            MapAndOutlierRemoval()
        }
        TrainModel()
        StoreResults()
    }
}
```



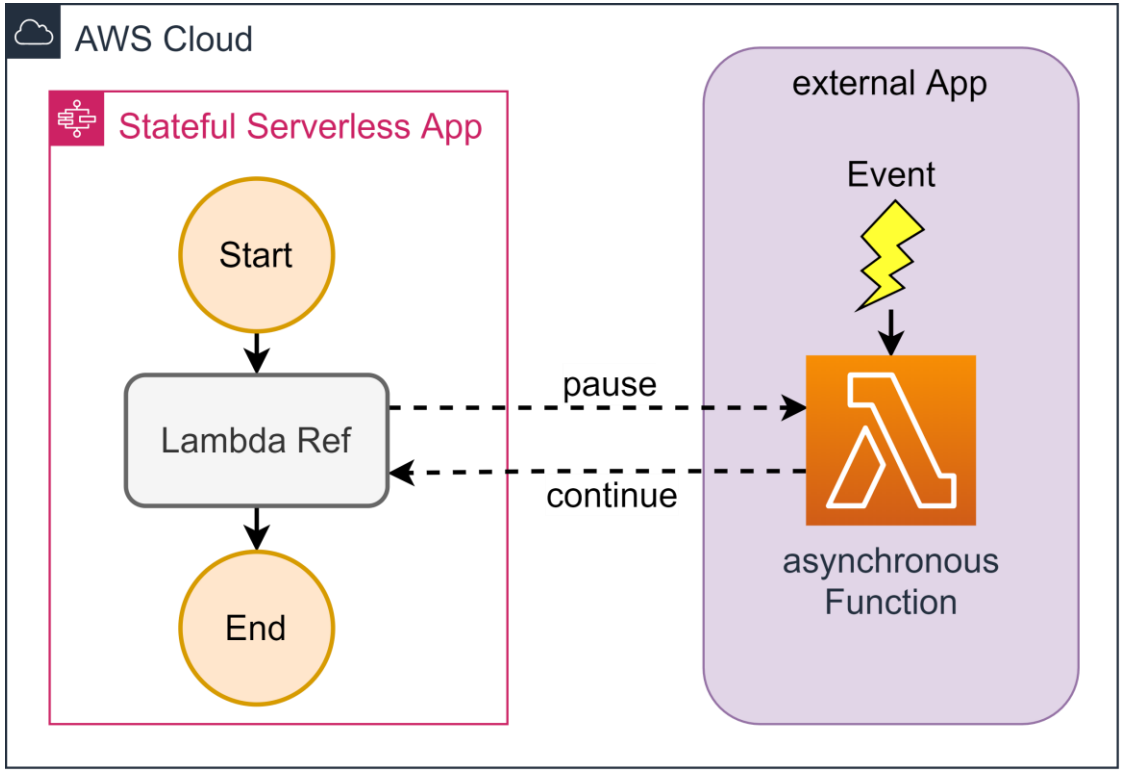
SFL Language

SFL Code

```
Statics {
    Integer x
}
Functions {
    ComputeFunction<python3_8>(x)
        ~!{
            #This does heavy
                computation
    }!~
}
StateCode {
    Start() {
        x = 0
        while(x < 10) {
            ComputeFunction()
            x = x + 1
        }
    }
}
```

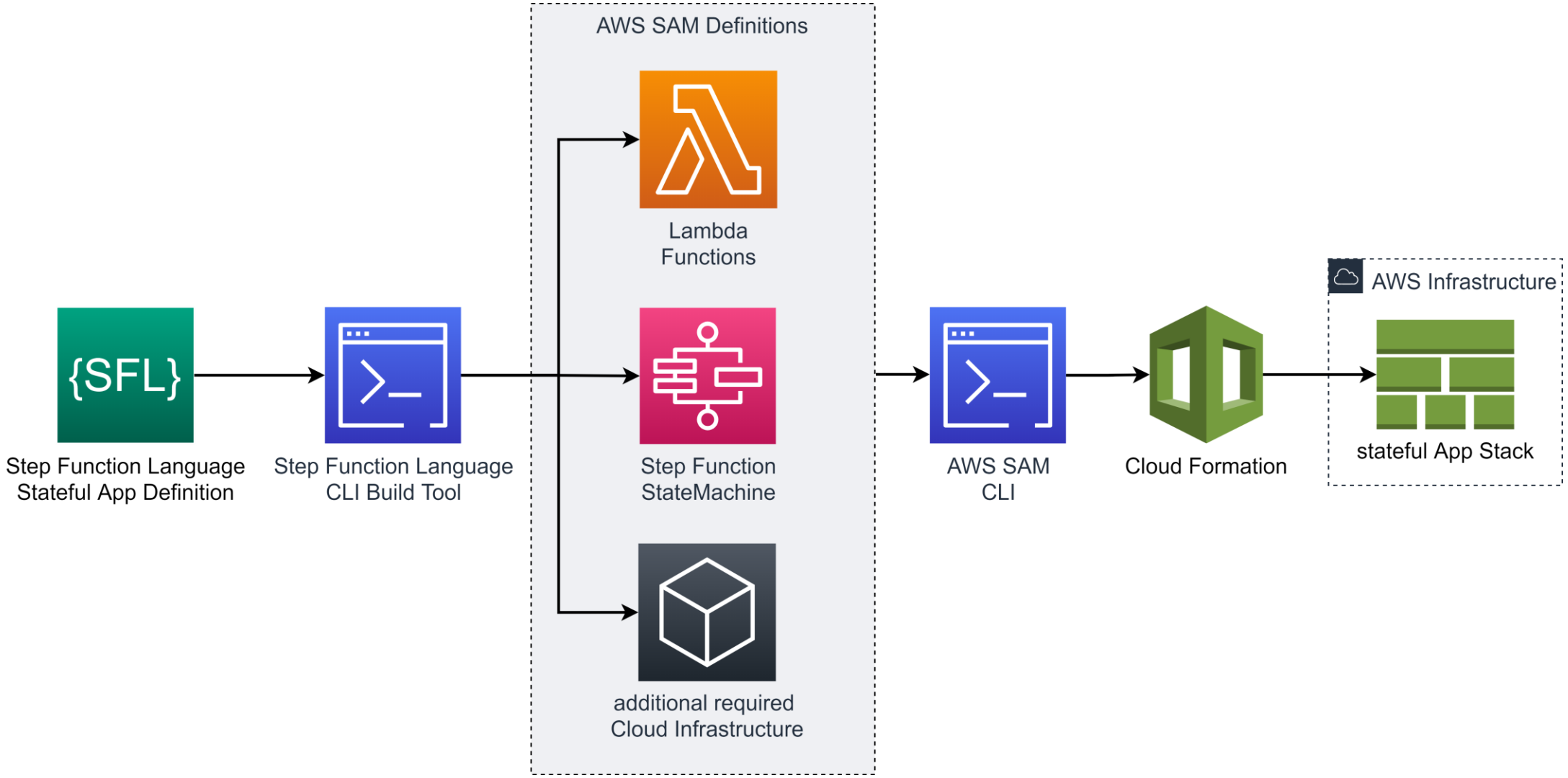


Asynchronous Lambda Functions




```
"Task-e5ac8996-2072-45f6-991a-a8ce301a3187" : {
  "Parameters" : {
    "FunctionName" : "${FunctionAWrapperRef}",
    "Payload" : {
      "taskToken.$" : "$$.Task.Token",
      "Input.$" : "$"
    }
  },
  "Next" : "Choice-732060a2-2348-4568-b13b-118d962c34f5",
  "Resource" : "arn:aws:states:::lambda:invoke.waitForTaskToken",
  "Type" : "Task"
},
```

SFL Build Pipeline



Evaluation

Manual implementation of the same applications on Azure

```
StateCode {
    Start() {
        x = 0 //Integer
        d = x + 1 //Double
        s = "zero" //String
        while(x < 10) { FunctionA() }
        while(d < 6) { FunctionB() }
        while(s ≠ "ten") { FunctionC() }
    }
}
```

```
import azure.durable_functions as df

def orchestrator_function(context: df.DurableOrchestrationContext):
    x: int = 0
    d: float = x + 1
    s: str = "zero"

    while x < 10:
        x = yield context.call_activity('FunctionA', x)

    while d < 6:
        d = yield context.call_activity('FunctionB', d)

    while s ≠ "ten":
        s = yield context.call_activity('FunctionC', s)

    return {"x": x, "d": d, "s": s}


main = df.Orchestrator.create(orchestrator_function)
```

Evaluation Applications

Our compiler was used to create 2 microbenchmarks on AWS

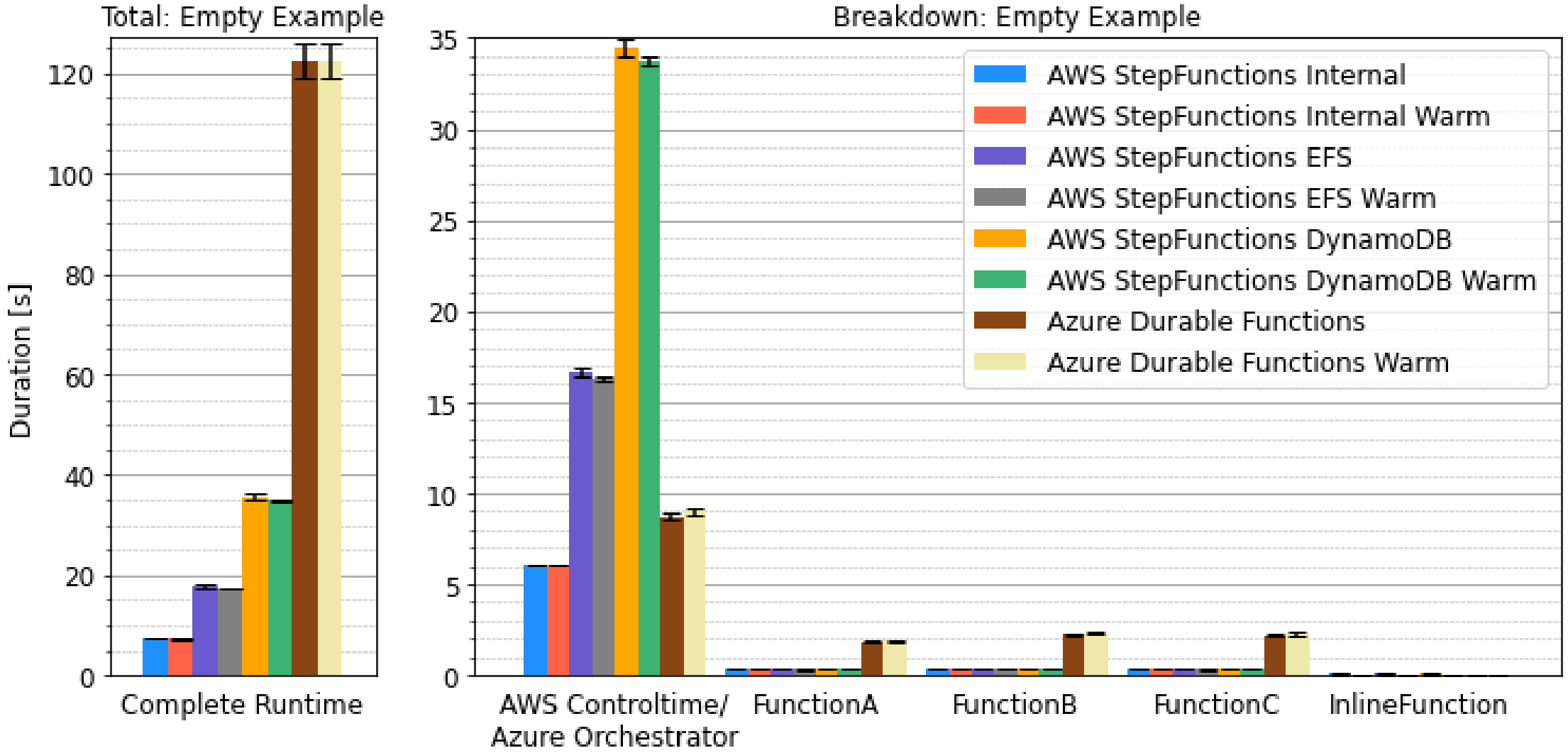
Empty Example

```
Functions {
    FunctionA<python3_8>(x) ~!{
        x = x + 1
    }!~
    // ...
}
```

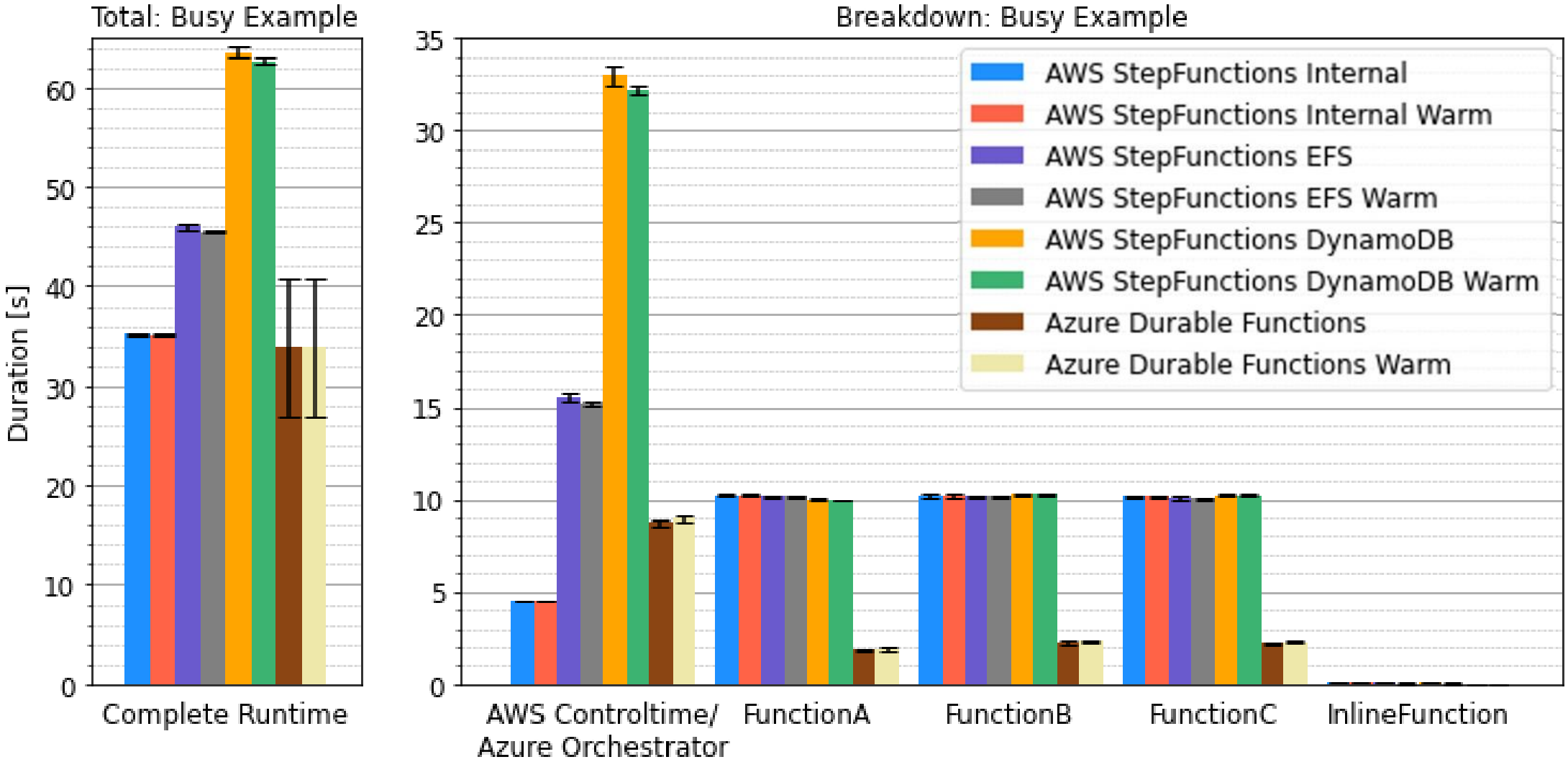
Busy Example

```
Functions {
    FunctionA<python3_8>(x) ~!{
        i = 0
        while i < 1_000_000 :
            i += 1

        x = x + 1
    }!~
    // ...
}
```

Empty Example



Busy Example



Thank you for your attention!