博客：AWS StepFunction 与外部应用程序集成的深入剖析

Step Functions (以下简称SF)是AWS提供的一项无服务器编排服务，您可以将AWS Lambda 函数和其他 AWS 服务组合在一起，以构建关键型业务应用。通过 SF的图形控制台，您可以使用基于Json的ASL(Amazon Sate Language)快速构建可视化的应用程序工作流，并以图形界面方式监控每个工作流的运行状态。

在实际场景中，我们通常会遇到工作流中有对外部应用做异步调用的需求，这些应用可能是运行在EC2/EKS甚或移动设备的一段程序，也可能是等待人工审批的一段流程，这些外部应用往往需要数分钟、数小时，甚至数天才能完成。Stepfunction提供的基于activity的回调模式可以完美支持以上场景，不但可以方便地定义异步任务和等待流程，并且在等待任务完成的过程中不会产生任何额外费用。

本博客将模拟一个典型场景，通过需求场景描述、stepfunction原理讲解以及代码实现几个步骤，为您深入剖析Stepfunction对回调模式的实现。

1. 场景描述

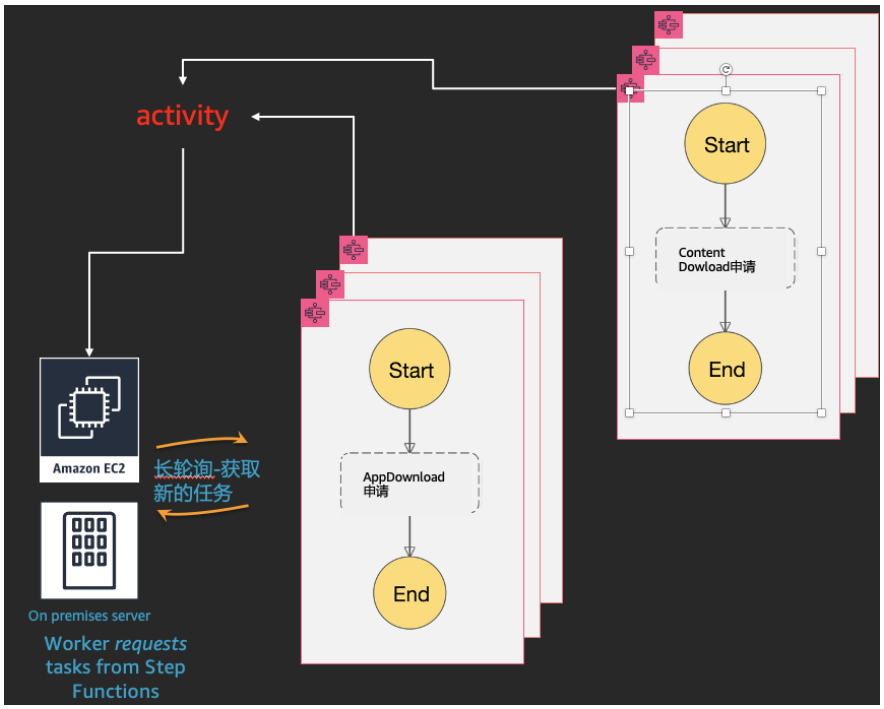
本博客模拟的场景是基于运行在EC2上的一个公共服务模块构建多个stepfunction流程。

比如，我们在EC2上构建了一个公共校验服务并持续运行，该服务会根据请求者的年龄来判断请求者是否可以通过校验。而上层则有多种类型的流程需要使用该校验服务，如app安装流程、内容下载流程等等。每个工作流只有通过年龄校验，才可以继续下一步动作。

2. Stepfunciton工作原理

对上述场景，您可以使用SF提供的activity功能在状态机中执行校验任务，并将该校验任务托管在EC2/EKS甚至本地服务器等几乎任何位置中。本示例中任务将托管在EC2中，通过python脚本实现。当SF状态机执行到activity所在的任务状态时，SF将安排活动并等待校验任务的工作线程给予响应。

校验任务的工作线程通过使用GetActivityTask并发送相关activity的 ARN 来轮询SF。当activity中有新的等待任务时，工作线程的GetActivityTask获得响应，响应内容包括任务的输入参数和 任务的唯一标识符（taskToken）。工作线程根据输入参数执行本次的校验任务，完成工作后，可以使用SendTaskSuccess或SendTaskFailure向SF的activity提供成功或失败报告。



实现方法

3. 实现方法

作为示例，我们设计了两个工作流，分别用于模拟AppDownload申请流程和ContentDownload申请流程，执行这两个流程时都会包含年龄作为输入参数。创建过程包含如下步骤：

i. 创建activity
ii. 分别为两个流程创建独立的状态机
iii. EC2上构建持续运行的校验服务
iv. 流程测试

3.1 创建activity

在Step Functions控制台中，创建一个名为ageCheck的活动，并记录此活动的arn：
arn:aws:states:**RegionId**:**AccountId**:activity:ageCheck

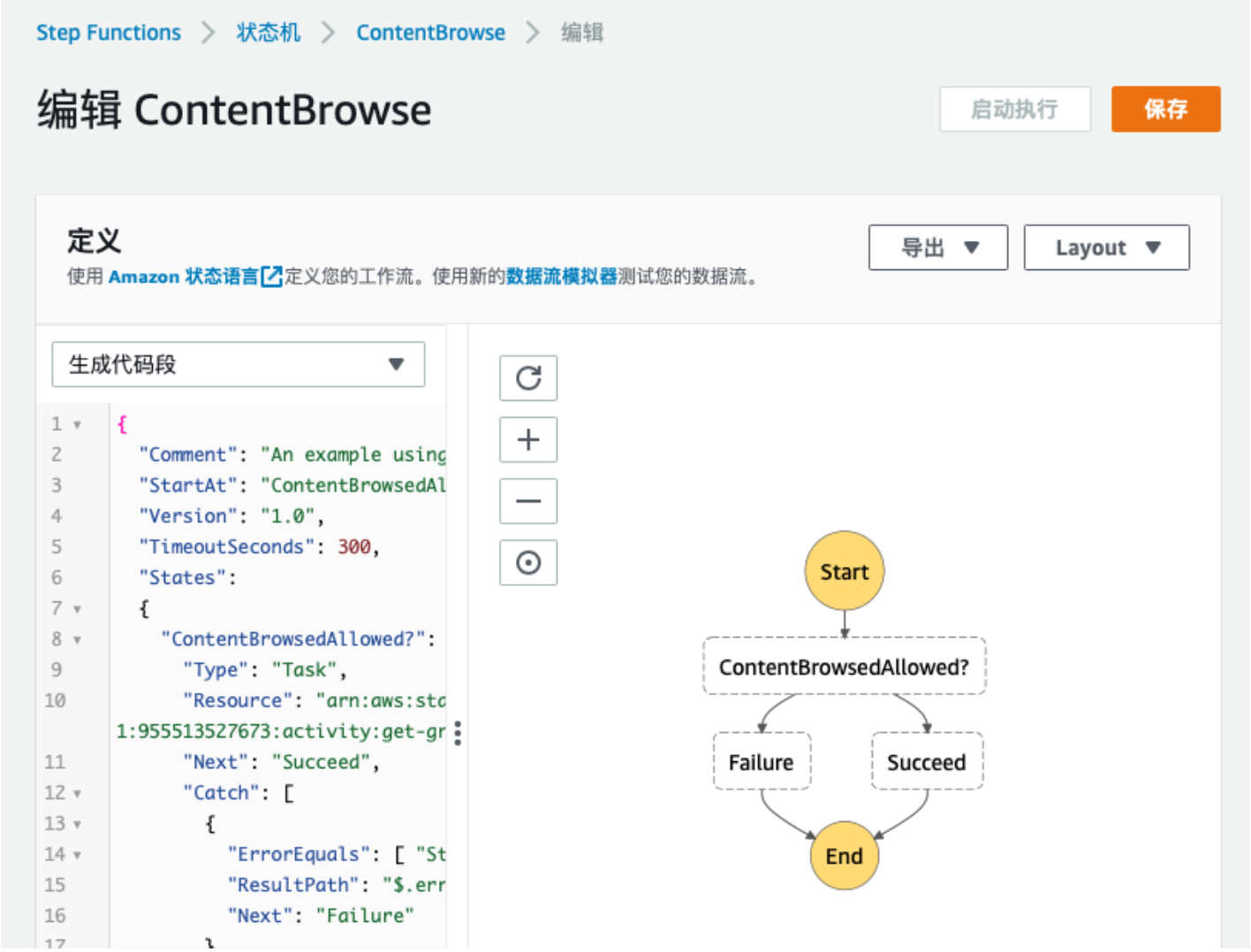
3.2 分别为两个流程创建独立的状态机

在Stepfunction控制台上为ContentBrowse工作流创建状态机。使用控制台创建的默认角色。将状态机命名为ContentBrowse，并使用以下ASL代码。

代码中Resource即为上步骤中创建的activity arn，请用实际内容替换。代码中的TimeoutSeconds为状态机的等待超时时间，该时间最长可指定为一年。

JSON

```
{
  "Comment": "An example using a Task state.",
  "StartAt": "ContentBrowsedAllowed?",
  "Version": "1.0",
  "TimeoutSeconds": 300,
  "States":
  {
    "ContentBrowsedAllowed?": {
      "Type": "Task",
      "Resource": "arn:aws:states:eu-west-1:955513527673:activity:ageCheck",
      "Next": "Succeed",
      "Catch": [
        {
          "ErrorEquals": [ "States.ALL" ],
          "ResultPath": "$.error-info",
          "Next": "Failure"
        }
      ]
    },
    "Failure": {
      "Type": "Fail",
      "Cause": "Invalid response.",
      "Error": "CheckFailed"
    },
    "Succeed": {
      "Type": "Succeed"
    }
  }
}
```



用类似的方法创建AppDownload状态机。该状态机的工作流程可以与ContentBrowse不同，而是仅在年龄校验部分使用相同的activity资源。

3.3 EC2上构建持续运行的校验服务

下面的示例使用python脚本构建了一个简单的基于activity的工作线程。该脚本使用 get_activity_task 持续轮询上面创建的activity。当有新的任务加入时，该线程将获得响应，响应中包含任务的TaskToken和输入参数。

任务从响应中取得输入参数进行处理。当校验通过时，即携带TaskToken向activity发送 send_task_success；否则发送send_task_failure，并包含错误代码，该错误代码可以用于后续流程的错误处理。

Python

```python
import boto3
import os
import time
from botocore.exceptions import ClientError
import urllib
import json

print('Loading function')
#sfnArn = os.environ['sfnarn']
sfnArn = "arn:aws:states:RegionId:AccountId:activity:ageCheck"
print(sfnArn)

sfClient = boto3.client('stepfunctions')

outputstring = "{\"type\":\"adult\",\"message\":\"allowed\"}"

while 0<1:
  response = sfClient.get_activity_task(activityArn=sfnArn, workerName='abc')
  task_token, input_ = response['taskToken'], response['input']
  params = json.loads(input_)
  if task_token is None:
    time.sleep(5)
  else:
    age = params.get('age')
    if age<18:
      print("under 18")
      sfClient.send_task_failure(
        taskToken=task_token,
        error='age check failed',
        cause='child under 18 is not allowed!')
    else:
      print("more than 18")
      sfClient.send_task_success(
        taskToken=task_token,
        output=outputstring)
```

Stepfunction的activity在收到响应后可以进行相应分支选择。本示例中，缺省会进入succeed分支，如果捕获到error信息则进入Failure分支。

4. 流程测试

为了让您更直观地理解activity的运行原理，我们分别通过命令行和程序运行两种方式进行测试

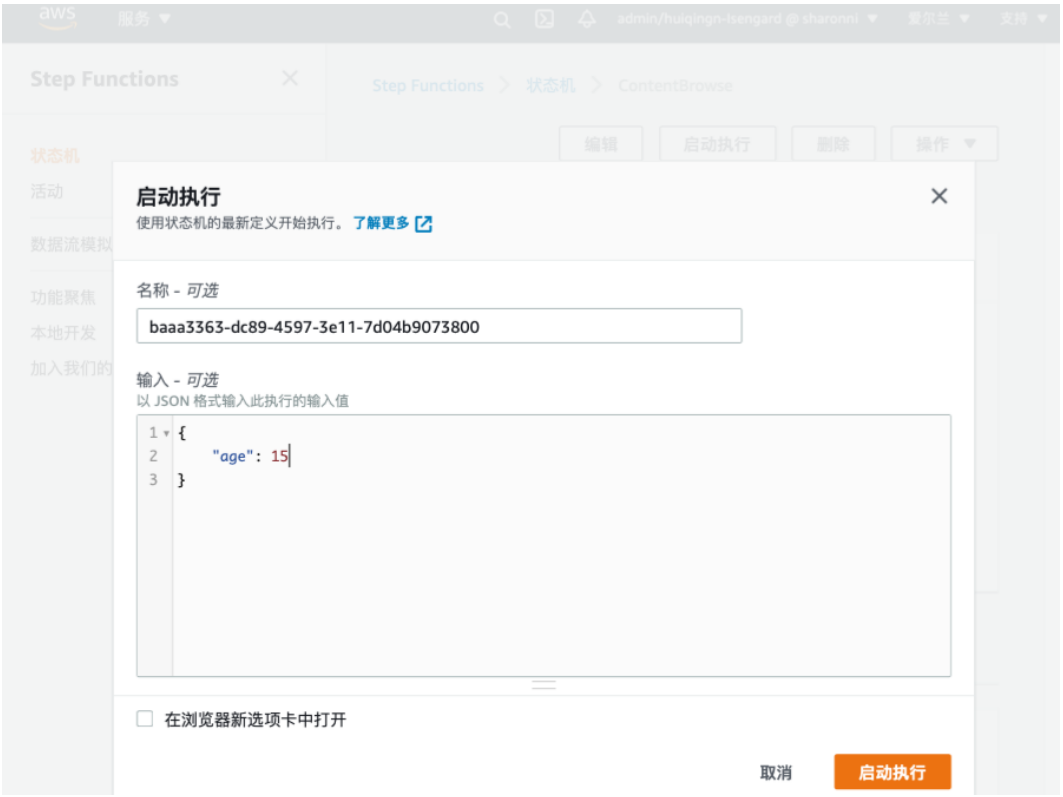
v. 使用aws stepfunctions命令行模拟外部应用进行测试：

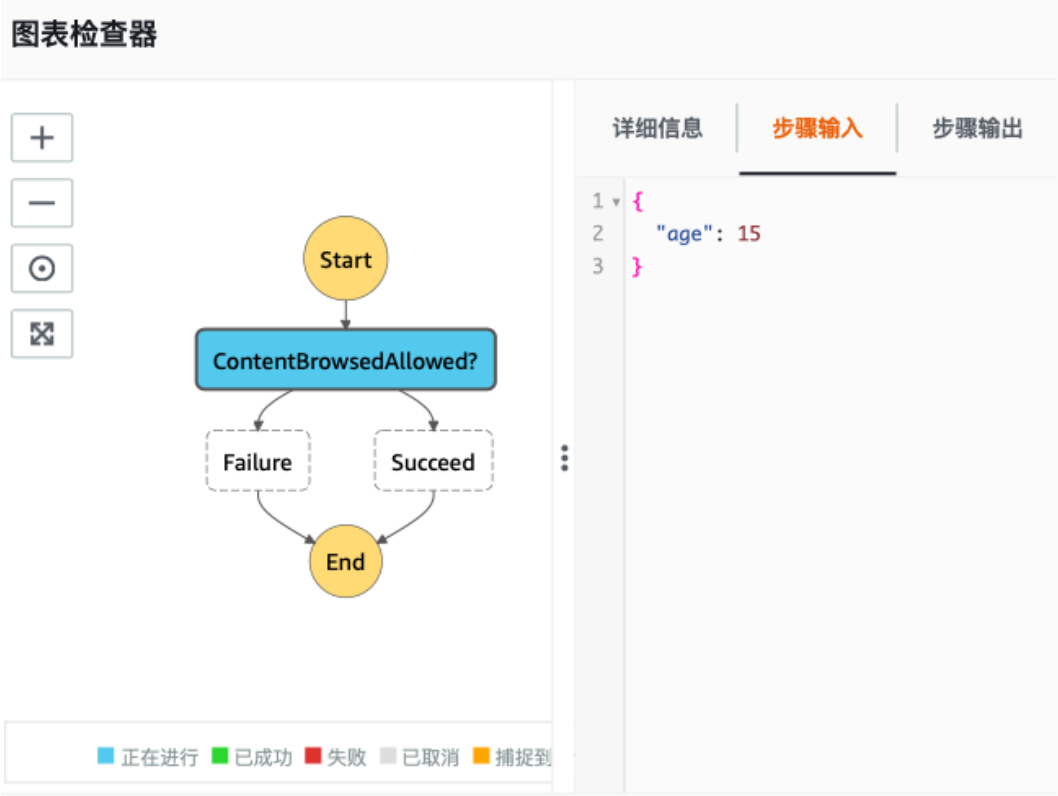
   启动一个工作流并使状态机等待在activity所在任务，通过命令行方式获取TaskToken和输入参数

vi. 运行EC2上的校验服务，测试状态机的运行

4.1 使用aws stepfunctions命令行模拟外部应用进行测试

首先基于如下input启动ContentBrowse状态机





然后用如下命令行获取当前activity上的任务信息

Shell

```
aws stepfunctions get-activity-task --activity-arn arn:aws:states:eu-west-1:955513527673:activity:ageCheck
```

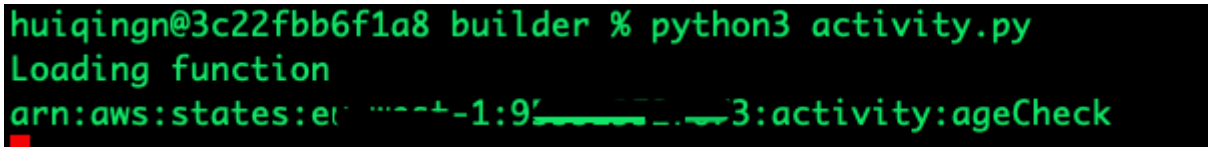
可以看到，get-activity-task将获得包含tasktoken和input两部分内容的返回结果。应用程序只需进一步获取这部分内容，即可进行逻辑处理。

JSON

```
{
    "taskToken": "AAAAKgAAAAIAAAAAAAAAAeLyqk91Rq3vEI6YFkSC+5KiFECN3YQvqfzuFHTxLOh9AqC9pF+XBiplHRd68ttWAPspQ3dYqhBdr6fiNE0ecPi87q3FnCoNTKN2ZvSgCmNheN4S1kCT8O/WXNpFwXwS5GtvyYpQAbGU1DcA2SayArm00dxGh25LHUXL+noY9n4UX2pHuCgTULSIdrEBjQiPsJCcBwJKIuSAIW1O6vG8MluEIiB1jjtvAQEwXGkaYkPXoZr3H+gQeXNZ5s5st3UWEFX+c4fhEka2IZxMCqNjSBLN6IYOd5xg8z9/b5Rcp0N/NYNTyg3DZUXeL2pyv57WoQ8R/RVgvRRnPMR6dpKOGbUWvwExvDK12pDKMQT6/UbkxLuP+R1vXgrzVjXhdXlvE/2z3KssSdfxOPBEI4g06GEtEtOTR7Lecbpcyq10i08BYiIAZhap+panflcdIPvbH5LuAemIeKYtUy4y1Mu0i8hrxf/cSecfM50Kom7kR4vBH+aD32ayscp8LFqBi058eN5pgdWHDYgJE/jst0taphY=",
    "input": "{\n    \"age\": 15\n}"
}
```

4.2 运行EC2上的校验服务，测试状态机的运行

首先启动校验服务，并持续运行：



然后分别启动ContentBrowse 和AppDownload两个状态机进行测试，具体测试内容请参见如下录屏

5. 总结

基于stepfunction的activity功能，可以方便实现状态机与外部应用的集成。如果您有运行在本地、EC2或基于容器运行的应用，希望不做应用迁移的情况下在AWS上快速构建新的工作流，使用stepfunction的activity功能是个不错的选择。同时，您还可以基于本博客的思路，基于同一个应用构建多个不同流程，减少代码开发，实现资源共享。