Proposta de modelo agnóstico para composição e orquestração serverless de workflows científicos

André Vieira, Daniel Cordeiro *

[1]Programa de Pós-graduação em Sistemas de Informação (PPgSI)
Universidade de São Paulo (USP)

{vieira.andre,daniel.cordeiro}@usp.br

Abstract. *Serverless computing offers potential benefits for the execution of scientific workflows, but presents challenges related to the stateless nature of functions and the complexity of orchestration. This work proposes a model for composing and orchestrating scientific workflows in serverless environments, utilizing a platform-agnostic specification language to define the workflow structure. The approach employs an event-driven architecture with persistent state management to coordinate task execution. The goal is to enable portability and scalable execution of complex scientific processes in the cloud, overcoming the inherent limitations of the serverless paradigm.*

Resumo. *A computação serverless oferece potenciais benefícios para a execução de workflows científicos, mas apresenta desafios relacionados à natureza stateless das funções e à complexidade da orquestração. Este trabalho propõe um modelo para compor e orquestrar workflows científicos em ambientes serverless, utilizando uma linguagem de especificação agnóstica de plataforma para definir a estrutura do fluxo. A abordagem emprega uma arquitetura dirigida a eventos com gerenciamento de estado persistente para coordenar a execução das tarefas. O objetivo é permitir a portabilidade e a execução escalável de processos científicos complexos na nuvem, superando as limitações inerentes ao paradigma serverless.*

1. Introdução

A computação serverless, com sua promessa de redução da complexidade operacional e eficiência de custos, tem ganhado tração em diversos domínios [Baldini et al. 2017]. Por sua vez, workflows científicos tornaram-se indispensáveis na orquestração de tarefas computacionais complexas para pesquisa e análise de dados [Atkinson et al. 2017].

A interseção entre computação serverless e workflows científicos apresenta oportunidades e desafios. Enquanto arquiteturas serverless podem, potencialmente, aprimorar a escalabilidade e a utilização de recursos dos workflows científicos, elas também introduzem complexidades relativas à gestão de estado, transferência de dados e coordenação de execução. Além disso, a natureza *stateless* das funções serverless pode conflitar com

*Este trabalho foi realizado com apoio da Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), procs. #2019/26702-8, #2021/06867-2 e #2023/00811-0, do CCD Cidades Carbono Neutro (FAPESP #2024/01115-0), do CNPq (proc. #444766/2024-3) e do grupo de pesquisa THUS (Techno-Human Systems of the Future) do Centro Internacional de Pesquisa (IRC) 'Transitions' (CNRS, USP e FAPESP CIP #2025/01171-0).

os requisitos frequentemente *stateful* dos workflows científicos, demandando abordagens inovadoras para harmonizar essa divergência.

O objetivo deste trabalho é propor um modelo conceitual para compor e orquestrar workflows científicos de maneira serverless, agnóstico à plataforma de nuvem empregada, com planos para implementação e validação futura. Espera-se alcançar a portabilidade da especificação do fluxo, além de mecanismos de coordenação passíveis de implementação em diferentes provedores.

2. Fundamentação

Funções serverless, que são as unidades de implantação do modelo Function-as-a-Service (FaaS), permitem a execução de código em contêineres *stateless*, gerenciados dinamicamente pelo provedor de nuvem [Castro et al. 2019]. A complexidade operacional é reduzida, visto que a alocação e gerenciamento de servidores ficam a cargo do provedor.

Workflows científicos são processos estruturados que combinam dados e operações em etapas configuráveis para resolver problemas científicos [Deelman et al. 2015]. Eles envolvem múltiplas fases de processamento, análise e computação, exigindo recursos significativos e uma orquestração complexa.

A *Abstract Function Choreography Language* (AFCL) é uma linguagem de domínio (DSL) projetada para definir o fluxo de funções serverless de forma agnóstica à plataforma [Ristov et al. 2021]. Ela abstrai a estrutura e o comportamento do workflow, suportando cenários complexos de fluxo de dados.

3. Proposta

O trabalho, em seu estágio preliminar atual, visa uma arquitetura projetada para executar workflows científicos especificados na AFCL (versão 1.1) na Amazon Web Services (AWS), com planos para implementação e validação futura. O objetivo é analisar uma especificação e coordenar dinamicamente a invocação das funções (tarefas do workflow) utilizando serviços nativos da plataforma de nuvem para o gerenciamento de estado e o disparo de etapas da execução.

3.1. Arquitetura

A arquitetura é composta pelos seguintes componentes:

- **Função Lambda coordenadora.** Responsável por interpretar a especificação do workflow AFCL, invocar funções Lambda específicas para cada tarefa e gerenciar a progressão do workflow com base em mudanças de estado. Suporta diversos construtos da AFCL, incluindo sequências, condicionais (*if*, *switch*), execução paralela e laços de repetição (*parallelFor*, *for*, *while*).
- **Funções Lambda de tarefas.** Funções independentes e *stateless*, executadas de forma assíncrona, que representam tarefas individuais do workflow tratadas como caixas pretas. Os resultados das tarefas são capturados por meio da funcionalidade Lambda Destinations, garantindo baixo acoplamento e orquestração reativa.
- **Tabela de estado no DynamoDB.** Um mecanismo de armazenamento persistente para manter o estado do workflow, resultados das tarefas, contadores para execução paralela e dados de iteração dos laços de repetição. A tabela de estado

permite que a coordenadora recupere, acompanhe e progrida o workflow de forma contínua entre invocações do Lambda.
- **DynamoDB Streams.** Um gatilho que invoca a Lambda coordenadora mediante eventos de conclusão de tarefas registrados no DynamoDB. Esse design garante uma progressão da orquestração com base nas tarefas concluídas, permitindo execução eficiente e responsiva do workflow.

Esses componentes operam de forma integrada para criar uma máquina de estados leve, baseada em eventos e sem execução contínua. A orquestração ocorre em invocações curtas do Lambda, que reagem às mudanças de estado. Isso garante escalabilidade: o Lambda pode instanciar múltiplas funções coordenadoras simultaneamente em resposta a eventos concorrentes. O dimensionamento sob demanda do DynamoDB lida com altas taxas de leitura/gravação do estado, tornando o sistema totalmente serverless e escalável, com o workflow sendo conduzido pelas atualizações nos dados.

3.2. Desafios

Os principais desafios abordados nessa arquitetura são:

- ***Parsing* e modelagem.** Interpretar a AFCL 1.1 e representar sua estrutura de workflow em uma linguagem de programação.
- **Invocação de funções e passagem de dados.** Cada "função" do AFCL corresponde a uma função Lambda a ser invocada (tratada como caixa preta). As saídas de dados de uma função tornam-se entradas para outras; os resultados devem ser corretamente direcionados.
- **Ramificações e laços**. A lógica condicional no AFCL (e.g., um *switch* escolhendo entre *f2* e *f3*) deve resultar na invocação apenas das funções relevantes. Laços de repetição exigem invocações repetidas até que uma condição ou contagem seja alcançada.
- **Paralelismo e sincronização.** Blocos paralelos ou laços do tipo *parallelFor* na AFCL requerem invocações simultâneas de múltiplas funções, com agregação dos resultados antes de prosseguir (barreira de sincronização).
- **Escalabilidade dentro dos limites do Lambda**. Para garantir escalabilidade dentro das restrições do serviço, como o tempo de execução de 15 minutos e limites de memória, a orquestração deve ser dirigida a eventos e composta por invocações *stateless*. Isso permite a coordenação passo a passo do workflow, com cada etapa executada em resposta a eventos específicos.

3.3. Análise comparativa

Plataformas como Apache Airflow e Argo Workflows contam com mecanismos maduros de gestão de estado e suporte a workflows complexos, porém, requerem infraestrutura em constante operação, o que contrasta com a natureza serverless proposta neste trabalho. Já o Step Functions, embora serverless e integrado ao ecossistema AWS, apresenta risco de aprisionamento tecnológico (*vendor lock-in*) e sua linguagem de definição (Amazon States Language) difere da proposta agnóstica da AFCL.

3.4. Limitações práticas

A dependência de um serviço de banco de dados para gerenciamento de estado pode se tornar um gargalo em cenários de altíssima concorrência ou com payloads de estado muito

grandes, impactando a latência e os custos. Além disso, a natureza distribuída e reativa da solução aumenta a complexidade da depuração e monitoramento dos workflows. Outro fator a considerar é o impacto de *cold starts* nas funções, que podem introduzir atrasos em fluxos sensíveis ao tempo.

4. Considerações

Este trabalho propôs um modelo para execução de workflows científicos em ambientes serverless, utilizando a AFCL como especificação agnóstica. Apresentou-se uma arquitetura para implementação na AWS, combinando Lambda e DynamoDB para orquestração escalável e dirigida a eventos. Demonstrou-se como a persistência de estado e gatilhos reativos podem coordenar tarefas, incluindo paralelismo, condicionais e laços, superando limitações como o caráter *stateless* das funções e restrições de tempo de execução.

Embora o modelo tenha sido ilustrado conceitualmente para um provedor específico, a AFCL estabelece as bases para implementações em outras plataformas. Essa estratégia oferece uma solução escalável para orquestrar processos científicos em infraestruturas serverless, promovendo portabilidade e potencial redução de custos e tempo de processamento.

Investigações futuras incluem a avaliação do modelo com diferentes tipos de workflows, incluindo benchmarks padronizados como Montage[1] e CyberShake[2]. Estes experimentos deverão medir métricas essenciais como tempo de execução total, latência de invocação das funções, custo operacional e capacidade de escalabilidade sob diferentes cargas de trabalho e complexidades estruturais do workflow.

Também será explorada a implementação adaptada do orquestrador para outras plataformas de nuvem, visando validar completamente o objetivo de agnosticismo relativo ao provedor de nuvem escolhido.

Referências

Atkinson, M., Gesing, S., Montagnat, J., and Taylor, I. (2017). Scientific workflows: Past, present and future. *Future Generation Computer Systems*, 75:216–227.

Baldini, I., Castro, P., Chang, K., Cheng, P., Fink, S., Ishakian, V., Mitchell, N., Muthusamy, V., Rabbah, R., Slominski, A., and Suter, P. (2017). *Serverless Computing: Current Trends and Open Problems*, pages 1–20. Springer Singapore, Singapore.

Castro, P., Ishakian, V., Muthusamy, V., and Slominski, A. (2019). The rise of serverless computing. *Commun. ACM*, 62(12):44–54.

Deelman, E., Vahi, K., Juve, G., Rynge, M., Callaghan, S., Maechling, P. J., Mayani, R., Chen, W., Ferreira da Silva, R., Livny, M., and Wenger, K. (2015). Pegasus, a workflow management system for science automation. *Future Generation Computer Systems*, 46:17–35.

Ristov, S., Pedratscher, S., and Fahringer, T. (2021). Afcl: An abstract function choreography language for serverless workflow specification. *Future Generation Computer Systems*, 114:368–382.

[1] Workflow de processamento de imagens astronômicas.

[2] Workflow sismológico para análise probabilística de perigo sísmico.