

Aprimoramento do Framework Orama⁺⁺ para Suporte à IBM Cloud Code Engine e Avaliação de Benchmarks

Gustavo Rinaldi Braga de Albuquerque¹, Leonardo Rebouças de Carvalho¹, Aleteia Araujo¹

¹Departamento de Ciência da Computação – Universidade de Brasília (UnB)
Caixa Postal 70910-000 – Brasília – DF – Brasil

gustavorbalbuquerque@gmail.com, leouesb@gmail.com, aleteia@unb.br

Abstract. *The Function-as-a-Service (FaaS) paradigm offers scalability, but faces vendor lock-in challenges. Consequently, platforms based on open standards, such as IBM Cloud Code Engine, are gaining prominence in the market. This work presents the expansion of the Orama⁺⁺ framework with the integration of IBM Cloud Code Engine, automating the resource lifecycle. Experiments across three use cases confirm the effectiveness of the solution, consolidating Orama⁺⁺ for evaluations across five major cloud providers.*

Resumo. *O paradigma Function-as-a-Service (FaaS) oferece escalabilidade, mas enfrenta os desafios de vendor lock-in. Sendo assim, plataformas baseadas em padrões abertos, como o IBM Cloud Code Engine, ganham destaque no mercado. Este trabalho apresenta a expansão do framework Orama⁺⁺ com a integração do IBM Cloud Code Engine, automatizando o ciclo de vida dos recursos. Experimentos em três casos de uso confirmam a eficácia da solução, consolidando o Orama⁺⁺ para avaliações em cinco grandes provedores.*

1. Introdução

A computação em nuvem consolidou-se como base para sistemas em larga escala, com gastos globais previstos para ultrapassar 1,4 trilhão de dólares até 2029 [Gartner, Inc. 2025]. Dentre os serviços, destaca-se o paradigma *Function-as-a-Service* (FaaS), projetado para atingir 15,02 bilhões de dólares com crescimento anual de 27,8%. O FaaS viabiliza o modelo *Serverless* [Nupponen and Taibi 2020], no qual o provedor gerencia a infraestrutura e as funções são disparadas em resposta a eventos.

Entretanto, a heterogeneidade entre plataformas gera forte dependência de fornecedor (*vendor lock-in*) que, além de dificultar comparações justas de desempenho e custo [Hassan et al. 2021], essa dependência restringe a arquitetura da aplicação a APIs proprietárias, tornando futuras migrações entre provedores um processo de alto custo e risco. Para mitigar esse problema em ambientes multinuvem, o *framework* Orama⁺⁺ [Rebouças de Carvalho et al. 2026] orquestra *benchmarks* em provedores como AWS, GCP, Azure e Alibaba Cloud, empregando Aprendizado de Máquina para predição de desempenho. O Orama⁺⁺¹ automatiza o ciclo de vida de experimentos FaaS [de Carvalho et al. 2024], cobrindo a implantação e a coleta de métricas em casos de uso padronizados [Carvalho et al. 2023]. Assim, este trabalho expande a capacidade do *framework* Orama⁺⁺ integrando o IBM Cloud Code Engine como novo provedor suportado.

¹<https://github.com/unb-faas/orama>

2. Proposta

Diferentemente de plataformas FaaS tradicionais, a solução da IBM executa funções sobre infraestrutura em contêineres e padrões abertos Knative e Kubernetes da CNCF[IBM b], permitindo assim novas comparações entre arquiteturas.

Para viabilizar essa integração, foram desenvolvidos *scripts* em *Terraform* [IBM a], permitindo ao Orama⁺⁺ gerenciar o ciclo de vida dos recursos de forma transparente. Os arquivos `.tf` abrangem a configuração do provedor e variáveis, como região e limite de memória (`provider.tf` e `variables.tf`), e a criação do grupo de recursos e projeto no Code Engine (`faas-basic.tf`). Nos cenários com persistência, como o de banco de dados, também são provisionados serviços e credenciais externas, como no Cloudant (`cloudant.tf`) e no IBM Cloud Object Storage (`cos.tf`). Durante a declaração das funções GET, POST e DELETE, as credenciais e URLs necessárias são inseridas dinamicamente no código durante o provisionamento (`faas-get.tf`, `faas-post.tf` e `faas-delete.tf`). Ao final, o arquivo `outputs.tf` exporta os *endpoints* gerados para utilização pelo *framework*.

A extensão abrange três casos de uso (Figura 1): **(i) Calculadora (`icecalc`)**: processa parâmetros numéricos e executa operações matemáticas; **(ii) API para Banco de Dados (`ice2cloudant`)**: três funções de gerenciamento acopladas ao IBM Cloudant; e **(iii) API para Armazenamento (`ice2cos`)**: três funções vinculadas ao IBM Cloud Object Storage para manipulação de JSONs. A expansão consolida o Orama⁺⁺ como ferramenta unificada para orquestração e predição em cinco plataformas públicas.

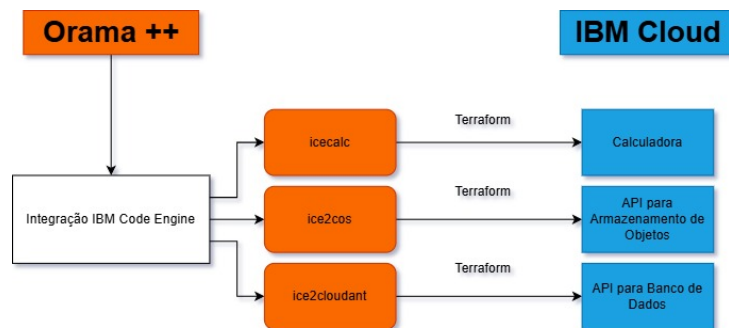


Figura 1. Diagrama de expansão do *framework* Orama⁺⁺.

3. Resultados e Conclusão

Os experimentos avaliaram o ciclo de vida automatizado no IBM Cloud Code Engine utilizando quatro níveis de concorrência (1, 2, 4 e 8), e trinta repetições (Tabela 1).

Tabela 1. Latência média dos casos de uso no IBM Cloud Code Engine.

Caso de uso	Operação	Latência média (ms)
Calculadora	Cálculo	1200,00
Cloudant	POST	2200,00
Cloudant	GET	2175,00
Object Storage	POST	2400,00
Object Storage	GET	2200,00

A calculadora obteve a menor latência (variando de aproximadamente 800 ms a 1700 ms) por ser um processamento isolado. As operações no Cloudant e Object Storage registraram médias superiores, variando na faixa de 1500 ms a 3500 ms, refletindo os custos de autenticação, rede e persistência. Os resultados confirmam que o Orama⁺⁺ executou com êxito o provisionamento e a coleta de métricas na infraestrutura IBM. Em trabalhos futuros, pretende-se adicionar suporte a cargas de trabalho mais complexas e um módulo de estimativa de custos específico para a IBM.

Referências

- Carvalho, L., Kamienski, B., and Araujo, A. (2023). How FaaS with DBaaS performs in different regions: an evaluation by the orama framework. In *Anais do XXIV Simpósio em Sistemas Computacionais de Alto Desempenho*, pages 241–252, Porto Alegre, RS, Brasil. SBC.
- de Carvalho, L. R., Kamienski, B., and Araujo, A. (2024). Main FaaS providers behavior under high concurrency: An evaluation with Orama framework distributed architecture. *SN Comput. Sci.*, 5(5).
- Gartner, Inc. (2025). Forecast: Public cloud services, worldwide, 2023-2029, 3q25 update. Technical report, Gartner, Inc., Stamford, CT, USA. Acessado em 16/07/2026.
- Hassan, H. B., Barakat, S. A., and Sarhan, Q. I. (2021). Survey on serverless computing. *Journal of Cloud Computing*, 10(1):39.
- IBM. *Getting started with Terraform on IBM Cloud*. IBM Cloud Docs. Acessado em: 4 ago. 2026.
- IBM. *Learn about Code Engine*. IBM Cloud Docs. Acessado em: 4 ago. 2026.
- Nupponen, J. and Taibi, D. (2020). Serverless: What it is, what to do and what not to do. In *2020 IEEE ICSA-C*, pages 49–50.
- Rebouças de Carvalho, L., Rocha Filho, G. P., and Araujo, A. (2026). Orama⁺⁺: Extending serverless benchmarking with tiobe and halstead metrics for improved performance prediction. In *Proceedings of the 16th International Conference on Cloud Computing and Services Science - CLOSER*, pages 13–24. INSTICC, SciTePress.