RECA
REVISTA ELETRÔNICA DE COMPUTAÇÃO APLICADA
Vol. 5 Nro. 1 2024

COMPUTAÇÃO EM NUVEM E ARQUITETURA *SERVERLESS*:
Revisão bibliográfica sobre o uso de *serverless* em *back-end* de aplicações

Rodrigo Marques Ribeiro
Graduado em Engenharia de Software - Uni-FACEF
rodrigomarqribeiro@gmail.com

Rodolfo Marques Ribeiro
Graduado em Engenharia de Software - Uni-FACEF
rodolfomarqribeiro@gmail.com

Geraldo Henrique Neto
Mestre em Ciências com Ênfase em Informática Médica e Docente - Uni-FACEF
geraldo@facef.br

Resumo

A introdução da computação em nuvem e das arquiteturas *serverless* marca um avanço significativo na evolução da tecnologia da informação, proporcionando às organizações uma infraestrutura flexível e escalável. Este trabalho visa realizar uma análise detalhada dessas tecnologias, destacando suas características, benefícios e desafios, além de investigar suas aplicações práticas no cenário contemporâneo. A escolha do tema é justificada pela crescente importância dessas tecnologias na transformação digital das organizações, que buscam inovação e eficiência operacional. A análise inclui uma revisão bibliográfica que ilustra a aplicação prática dessas tecnologias, permitindo uma compreensão mais profunda de suas implicações e corroborado, em sua conclusão, o uso da arquitetura *serverless* para rápidas cargas de trabalho como operações de *backend* e BFFs (ao menos em áreas que necessitam de maior escala).

Palavras-chave: Computação em Nuvem. Arquiteturas Serverless. Transformação Digital. Eficiência Operacional.

Abstract

The introduction of cloud computing and serverless architectures marks a significant advancement in the evolution of information technology, providing organizations with a flexible and scalable infrastructure. This work aims to conduct a detailed analysis of these technologies, highlighting their characteristics, benefits, and challenges, as well as investigating their practical applications in the contemporary landscape. The choice of this topic is justified by the growing importance of these technologies in the digital transformation of organizations seeking innovation and operational efficiency. The analysis includes a bibliographic review that illustrates the practical application of these Technologies, allowing for a deeper understanding of their implications and concluding that the use of serverless architecture is suitable for rapid workloads such as backend operations and BFFs (at least in areas requiring greater scale).

RECA
REVISTA ELETRÔNICA DE COMPUTAÇÃO APLICADA
Vol. 5 Nro. 1 2024

Keywords: Cloud Computing. Serverless Architectures. Digital Transformation. Operational Efficiency.

1 Introdução

A adoção da computação em nuvem e de arquiteturas serverless representa um avanço importante na evolução da tecnologia da informação. Com a ascensão da computação em nuvem, as organizações agora têm acesso a uma infraestrutura flexível e escalável, enquanto a adoção de arquiteturas *serverless* está transformando a maneira como as aplicações são desenvolvidas e implantadas (Zhang; Cheng; Boutaba, 2010). Nesta seção, discutiremos a importância e a relevância dessas tecnologias na era digital atual.

Este trabalho tem como objetivo realizar uma análise abrangente da computação em nuvem e das arquiteturas *serverless*, explorando seus conceitos, características, vantagens e desafios. Além disso, investigaremos suas aplicações práticas, que continuarão a moldar futuras tendências tecnológicas segundo Castro *et al.* (2019).

A escolha deste tema é impulsionada pela crescente relevância da computação em nuvem e das arquiteturas *serverless* no cenário tecnológico contemporâneo. Com a rápida transformação digital das organizações em busca de eficiência operacional e inovação, entender essas tecnologias tornou-se fundamental para profissionais da Tecnologia da Informação (TI), pesquisadores e acadêmicos. Assim, procuramos responder à seguinte questão: “Por que optar por *serverless* em vez de serviços tradicionais de *backend*?” A análise contempla uma revisão bibliográfica apresentada ao longo do artigo.

Este trabalho está estruturado da seguinte forma: na seção 2, serão discutidos os fundamentos da computação em nuvem, incluindo definições, modelos de serviço e de implantação. A seção 3 abordará os aspectos e desafios da computação em nuvem, como modelos de serviço, vantagens, desvantagens, segurança e migração. Em seguida, na seção 4, exploraremos as arquiteturas *serverless*, incluindo conceitos, implementação, ferramentas e exemplos práticos. A seção 5 apresentará uma comparação entre arquiteturas tradicionais e *serverless*, focando em desempenho, escalabilidade e custos. Na seção 6, discutiremos os desafios e as perspectivas futuras da computação em nuvem e das arquiteturas *serverless*, seguidos por uma revisão bibliográfica na seção 7. Por fim, na seção 8, serão apresentadas considerações finais e recomendações para trabalhos futuros.

2 Fundamentos da computação em nuvem

A computação em nuvem é um modelo de computação que permite o acesso *on-demand* (sob demanda) a um conjunto compartilhado de recursos de computação configuráveis, como redes, servidores, armazenamento, aplicativos e serviços. Esses recursos podem ser provisionados e liberados rapidamente com esforço mínimo de gerenciamento ou interação do provedor de serviços (Mell; Grance, 2011).

A computação em nuvem é caracterizada por cinco atributos essenciais: serviço sob demanda, acesso à rede de banda larga, agrupamento de recursos, rápida elasticidade e serviço medido. Esses atributos permitem que as organizações

aproveitem a computação em nuvem para melhorar a eficiência operacional, reduzir custos e aumentar a agilidade dos negócios (Mell; Grance, 2011).

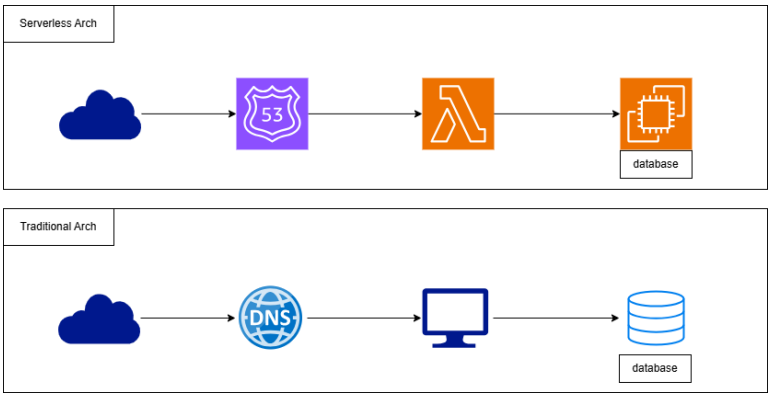
Existem três modelos de serviço de computação em nuvem: Infraestrutura como Serviço (IaaS), Plataforma como Serviço (PaaS) e Software como Serviço (SaaS). Cada modelo oferece diferentes níveis de controle, flexibilidade e gerenciamento, permitindo que as organizações escolham o modelo que melhor atenda às suas necessidades (Mell; Grance, 2011).

Além disso, a computação em nuvem pode ser implantada em quatro modelos: nuvem privada, nuvem comunitária, nuvem pública e nuvem híbrida (Mell; Grance, 2011). Cada modelo de implantação oferece diferentes níveis de segurança, privacidade e gerenciamento, permitindo que as organizações selecionem o modelo que melhor se adapte aos seus requisitos de negócios e segurança.

2.1 Arquiteturas tradicionais versus arquiteturas em nuvem

As arquiteturas tradicionais de TI geralmente envolvem a alocação de recursos de *hardware* e *software* em uma infraestrutura física local. Isso pode levar a altos custos de capital e operacionais, bem como a desafios em termos de escalabilidade e flexibilidade. Além disso, as arquiteturas tradicionais podem exigir um gerenciamento de TI significativo, o que pode desviar recursos valiosos de atividades de negócios mais estratégicas (Zhang; Cheng; Boutaba, 2010).

Figura 1: Ilustração do backend de aplicações tradicionais e *serverless*.



Fonte: Os autores.

Em contraste, as arquiteturas em nuvem permitem que as organizações acessem recursos de computação virtualizados por meio da Internet. Isso oferece benefícios significativos em termos de escalabilidade, flexibilidade e custo. Com a computação em nuvem, as organizações podem escalar rapidamente seus recursos de TI para atender às demandas flutuantes, sem a necessidade de investimentos significativos em infraestrutura de TI. Além disso, a computação em nuvem permite que as organizações paguem apenas pelos recursos de TI que usam, o que pode resultar em economias de custo significativas (Zhang; Cheng; Boutaba, 2010).

Na figura 1 pode ser visto um exemplo de ambas as arquiteturas. Embora o esboço das duas seja visualmente similar, há uma complexidade significativa a nível de

code-design da aplicação, especialmente para migrações do sistema para o ambiente de *cloud* e, por consequência, refatoração de alguns fluxos. Isso inclui questões relacionadas à segurança, privacidade, conformidade e interoperabilidade. Além disso, a migração de dados existentes para a nuvem pode ser um processo complexo e demorado (Zhang; Cheng; Boutaba, 2010). Portanto, é importante que as organizações considerem cuidadosamente esses desafios ao planejar essa transição de maneira a mitigar os riscos citados.

2.2 Visão geral da computação *serverless*

A computação *serverless*, também conhecida como *Function as a Service* (FaaS), é um modelo de computação em nuvem que abstrai a infraestrutura de servidores, permitindo que os desenvolvedores se concentrem apenas na escrita e implantação do código (Roberts, 2016). Neste modelo, a alocação de recursos é totalmente gerenciada pelo provedor de serviços em nuvem, o que permite uma escalabilidade automática e eficiente.

A computação *serverless* oferece vários benefícios em relação aos modelos tradicionais de computação, mesmo aqueles já hospedados em nuvem. Isso inclui a capacidade de desenvolver e implantar aplicativos mais rapidamente, reduzir a complexidade operacional e melhorar a eficiência de custos (Roberts, 2016). Logo, a computação *serverless* permite que as organizações paguem apenas pelo tempo de execução do código, em vez de pagar por recursos de servidor ociosos e/ou instâncias não usadas em sua totalidade.

No entanto, a computação *serverless* também apresenta desafios, como monitoramento e depuração de funções, gerenciamento de estado e latência de inicialização a frio[1]. Além disso, nem todas as cargas de trabalho são adequadas para esse modelo, especialmente aquelas que exigem longos períodos de execução ou uso intensivo da Unidade Central de Processamento (CPU) (Roberts, 2016), como operações de cronjobs não granulares e softwares legados. Portanto, é essencial avaliar cuidadosamente as características de cada aplicação antes de optar por uma arquitetura serverless, garantindo que ela seja a solução mais eficiente para as necessidades específicas.

3 Computação em nuvem: aspectos e desafios (IaaS, PaaS, SaaS)

A computação em nuvem oferece diferentes modelos de serviço, cada um com características e níveis de controle distintos para os usuários. Mais abaixo, os principais modelos estão expressos em detalhes:

- ***IaaS (Infrastructure as a Service)***: permite que os usuários controlem os sistemas operacionais e as aplicações, mas não a infraestrutura de rede, tornando possível a locação de servidores e armazenamento, no modelo *pay-as-you-go* (Buyya et al., 2009).
- ***PaaS (Platform as a Service)***: nesse modelo, os usuários gerenciam apenas as aplicações, enquanto a infraestrutura subjacente é administrada pelo provedor.

[1] A **inicialização a frio** ocorre quando uma função é chamada pela primeira vez ou após um período de inatividade, exigindo a criação e o carregamento de um novo ambiente de execução. Isso pode resultar em um atraso adicional, antes que a função comece a processar a solicitação (Roberts, 2016).

Esse modelo facilita o desenvolvimento e a implantação de aplicações (Buyya et al., 2009).

- ***SaaS (Software as a Service)***: nesse modelo, os usuários acessam aplicações via Internet, sem controle sobre o sistema operacional ou a infraestrutura. Dessa forma, elimina-se a necessidade de instalação local do software (Zhang; Cheng; Boutaba, 2010).

3.1 Vantagens e desvantagens da computação em nuvem

A computação em nuvem oferece várias vantagens, incluindo escalabilidade, flexibilidade, eficiência de custos e acesso a tecnologias avançadas (Mell; Grance, 2011). A escalabilidade permite que as organizações ajustem rapidamente seus recursos de Tecnologia da Informação (TI) para atender às demandas flutuantes. A flexibilidade permite que as organizações experimentem novas ideias e inovem rapidamente. A eficiência de custos permite que as organizações reduzam seus custos de recursos alocados, pagando apenas pelos recursos que usam. E o acesso a tecnologias avançadas permite que as organizações aproveitem as últimas inovações no ramo (Buyya *et al.*, 2009).

As questões de segurança e privacidade na migração para a nuvem são cruciais, pois os usuários precisam confiar nos provedores para proteger seus dados sensíveis. Essa confiança é frequentemente colocada à prova, dado que os provedores podem ter diferentes abordagens e níveis de segurança, deixando os dados vulneráveis a vazamentos e ataques. A pesquisa de Zhang, Cheng e Boutaba (2010) destaca que a segurança é uma preocupação central que não deve ser subestimada, uma vez que falhas nesse aspecto podem resultar em consequências graves para as empresas.

A dependência de provedores de serviços em nuvem pode resultar em bloqueio de fornecedores, conforme apontado por Andrikopoulos et al. (2013). Comprometer-se com um único provedor limita a flexibilidade e dificulta a adaptação a mudanças de mercado ou inovações tecnológicas.

3.2 Segurança e privacidade de dados na nuvem

Com base na segurança e privacidade de dados, aspectos essenciais na computação em nuvem, os usuários precisam confiar que os provedores de serviço implementem medidas eficazes de proteção, como criptografia, controle de acesso e auditorias frequentes. Contudo, incidentes de segurança e vazamentos de dados podem ocorrer devido a falhas de sistema, ataques cibernéticos ou erros operacionais, acarretando prejuízos financeiros e danos à reputação da organização (Catteddu; Hogben, 2009).

Além disso, a computação em nuvem levanta questões complexas sobre conformidade com leis e regulamentações de privacidade de dados, como o GDPR na União Europeia e a LGPD no Brasil. Esses regulamentos determinam requisitos específicos sobre coleta, uso, armazenamento e transferência de dados pessoais. As

organizações devem assegurar que suas práticas de computação em nuvem estejam em conformidade com essas leis para evitar sanções legais e proteger a confiança dos clientes. O não cumprimento pode resultar em penalidades, processos judiciais e sérios impactos na imagem institucional (Catteddu; Hogben, 2009).

3.3 Desafios na migração para a nuvem

A migração para a nuvem, por ser um processo complexo, as organizações devem avaliar cuidadosamente suas necessidades e requisitos, selecionar o modelo de serviço em nuvem e o provedor de serviços em nuvem adequados, planejar e executar a migração e gerenciar a infraestrutura em nuvem após a migração (Andrikopoulos *et al.*, 2013).

Além disso, as organizações devem considerar os custos de migração, o tempo de inatividade durante a migração, a compatibilidade das aplicações e a formação dos funcionários. A migração para a nuvem pode exigir mudanças significativas na arquitetura das aplicações, nos processos de negócios e nas habilidades dos funcionários. Portanto, é importante que as organizações planejem cuidadosamente a migração para a nuvem e considerem a assistência de consultores ou fornecedores especializados em migração para a nuvem (Andrikopoulos *et al.*, 2013).

4 Arquiteturas *Serverless*: conceitos e implementação

Os principais componentes de uma arquitetura *serverless* incluem funções, eventos, serviços de gerenciamento de estado e serviços de integração (Eivy, 2017). As funções são pedaços de código que são executados em resposta a eventos. Os eventos são sinais que indicam que algo aconteceu, e, pelo conceito de *observables*/*triggers*, chamam determinadas funções correlatas. Esses eventos variam desde solicitações Hypertext Transfer Protocol (HTTP) a alterações em banco de dados e serviços de mensageria (Eivy, 2017; Baldini *et al.*, 2017). Os serviços de gerenciamento de estado permitem que as funções mantenham e acessem o estado entre as invocações, já os de integração, permitem que as funções se comuniquem com outros serviços e aplicações.

4.1 Frameworks e ferramentas para desenvolvimento *serverless*

Existem vários *frameworks* e ferramentas disponíveis para o desenvolvimento *serverless*, como o *Serverless Framework*, que fornece uma maneira fácil de desenvolver, implantar e gerenciar aplicações em vários provedores de serviços em nuvem (Baldini *et al.*, 2017). Outras ferramentas incluem plataformas de integração contínua e entrega contínua (CI/CD), como o Jenkins, Travis CI, Gitlab CI, Github Action, etc., que permitem automatizar o processo de desenvolvimento e implantação de aplicações *serverless*.

Além disso, o uso de contêineres tem crescido significativamente. Isso se deve ao fato de que eles proporcionam uma forma rápida e eficiente de replicar ambientes de desenvolvimento ou teste local com precisão em ambientes remotos. Um exemplo disso no ecossistema Amazon Web Services (AWS) é o Amazon ECR (*Elastic Container Registry*), que oferece funcionalidades para armazenar e versionar imagens de contêineres, facilitando a gestão e o *deployment* de aplicações a partir de imagens de projetos de *software* (*blueprints*) existentes.

4.2 Exemplos de casos de uso e aplicações práticas

Por seus muitos *usos,* a arquitetura *serverless* é aplicada, na contemporaneidade, em aplicações web e móveis, *Application Programming Interfaces* (APIs) e processamento de dados em tempo real. A Netflix, por exemplo, usa essa arquitetura para codificar e processar vídeos, o que permite escalar rapidamente para atender à demanda dos usuários (Castro et al., 2019). Essa flexibilidade é importante para otimizar recursos e reduzir custos.

A Coca-Cola também adotou a arquitetura *serverless* em sua campanha Coca-Cola Freestyle, que introduziu o derramamento de refrigerantes por um dispenser sem contato físico do usuário para essa finalidade. Essa solução integra a infraestrutura do Amazon Web Services (AWS) e utiliza *WebSocket*, garantindo uma boa experiência do usuário. Com o AWS *Lambda* para computação e o *Amazon API Gateway* para conexões seguras e rápidas, os usuários podem escanear um código QR (*Quick Response*) e acessar uma interface móvel, escolhendo sua bebida e servindo-a em menos de um segundo. Essa configuração reduz atrasos, evitando transbordamentos e garantindo a satisfação do cliente.

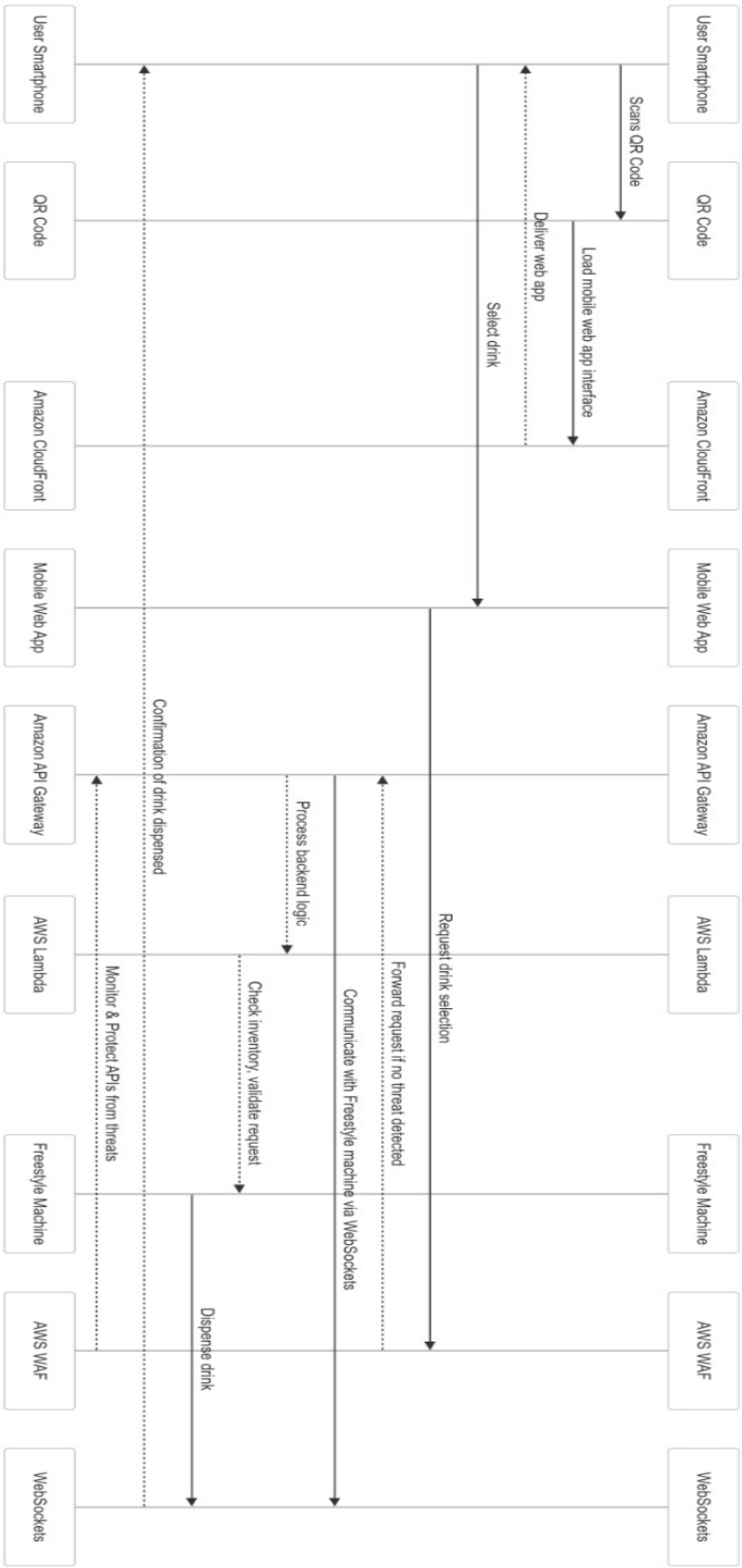
A Figura 2, elaborada pelos autores, ilustra o fluxo da funcionalidade de servir refrigerantes aos usuários na perspectiva da arquitetura de sistemas. Por meio de um diagrama de sequência que representa os fluxos internos da Coca-Cola Freestyle, observa-se que a fase do AWS Lambda é particularmente eficaz em termos de escalabilidade, permitindo que picos de carga de trabalho e altos volumes de requisições sejam atendidos quase instantaneamente, com o mínimo de downtime ou lentidão (Aws, 2020).

Além disso, o *Amazon CloudFron*t melhora a entrega de conteúdo, aumentando a velocidade e eficiência do serviço. Essa inovação não só atende às necessidades de higiene, especialmente em tempos de pandemia, mas também melhora a interação do usuário em um processo de seleção de bebidas que é divertido e rápido (Aws, 2020).

Retomando a revisão bibliográfica aplicada sobre o motor de processamento de vídeos da Netflix, conforme destacado por Castro et al. (2019), a plataforma emprega um *workflow* inovador que utiliza funções *serverless* para o processamento de vídeos, enviando-os eficientemente para o Amazon S3. Esse processo gera eventos que ativam funções *Lambda*, as quais desempenham um papel crucial na divisão e transcodificação simultânea dos vídeos em diversos formatos, conforme ilustrado de maneira clara na Figura 3.

RECA
REVISTA ELETRÔNICA DE COMPUTAÇÃO APLICADA
Vol. 5 Nro. 1 2024

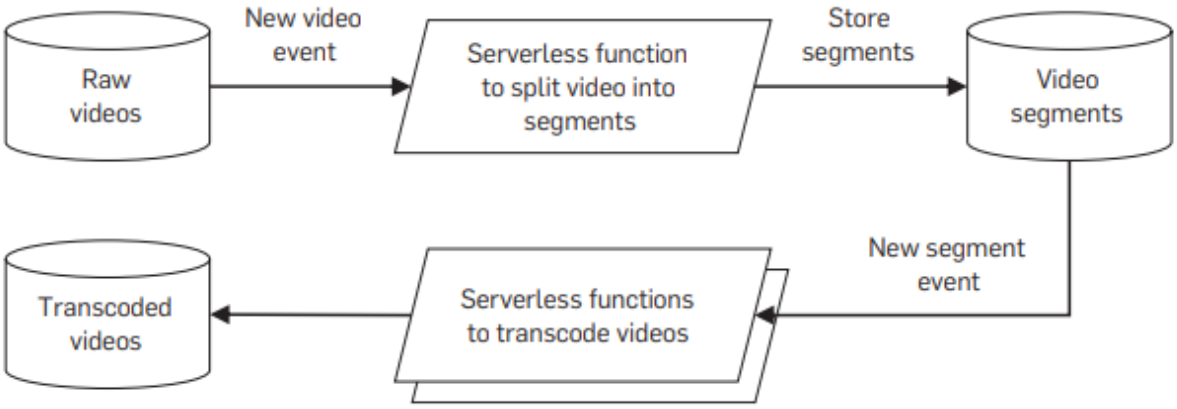
Figura 2: Diagrama de sequência do totem usado na campanha Coca-Cola Freestyle.



Fonte: Os autores.

RECA
REVISTA ELETRÔNICA DE COMPUTAÇÃO APLICADA
Vol. 5 Nro. 1 2024

Figura 3: *Workflow* do tratamento de dados de vídeo para serviços *streaming* da netflix.



Fonte: Castro et al. (2019, p. 7).

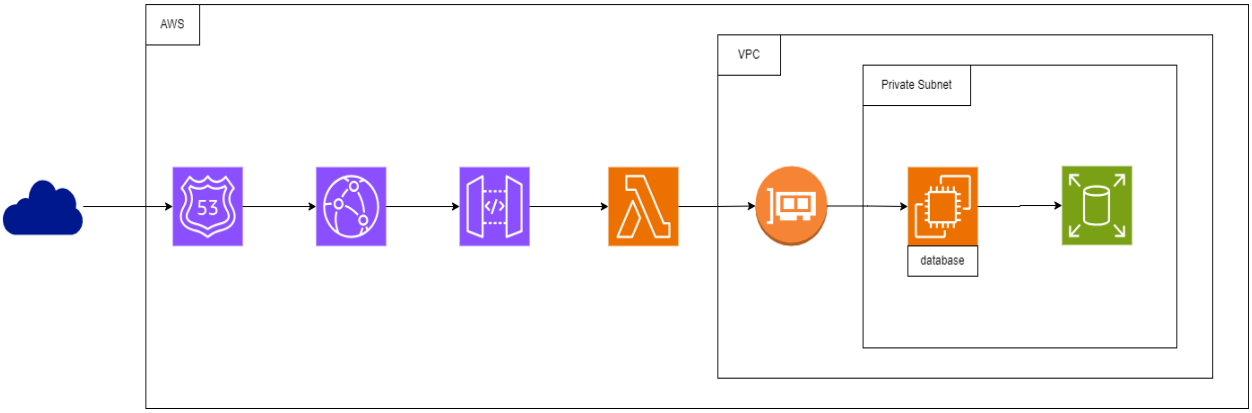
Essas funções se destacam por serem sem estado e idempotentes, o que significa que, em caso de falhas, como eventuais problemas de rede, sua reexecução não resulta em efeitos indesejados, garantindo robustez e confiabilidade ao sistema (Castro et al. 2019). Embora essa abordagem possa parecer simples à primeira vista, a combinação de funções *serverless* com outros serviços de nuvem abre um leque de possibilidades para o desenvolvimento de aplicações extremamente complexas e dinâmicas. Isso inclui desde processamento de fluxo e filtragem de dados em tempo real até a criação de *chatbots* interativos e aplicações *web* envolventes.

5 Comparativo e Análise de Performance

Na estrutura tradicional, as aplicações são desenvolvidas em servidores (físicos, virtuais ou em nuvem) geridos pela própria organização. No modelo serverless, porém, o provedor assume a administração e aloca recursos de forma dinâmica conforme o consumo do cliente (Baldini et al., 2017). Esse modelo oferece vantagens significativas em escalabilidade e eficiência operacional, mas também impõe desafios, como o gerenciamento do estado da aplicação e a latência de inicialização a frio após períodos de inatividade (Eivy, 2017).

RECA
REVISTA ELETRÔNICA DE COMPUTAÇÃO APLICADA
Vol. 5 Nro. 1 2024

Figura 4: Arquitetura *Serverless* com o uso de Funções Lambda.



Fonte: Os autores.

Como apresentado na Figura 4, ao utilizar um *API Gateway* no ecossistema Amazon Web Services (AWS), a arquitetura da aplicação passa a ser orientada a eventos, resultando em uma estrutura mais distribuída. Devido a essa natureza orientada a eventos, não é viável adotar padrões como *singleton* ou manter toda a aplicação dependente de uma conexão persistente (*connection pool*). Nesse contexto, o gerenciamento pode ser eficazmente realizado pelo próprio *gateway*, pois ele permite a configuração de políticas de segurança e controle de taxa (*rate limiting*) diretamente.

Todos os serviços, como o Sistema de Nomes de Domínio (DNS), Rede de Distribuição de Conteúdo (CDN), Sistema de Gerenciamento de Banco de Dados (DBMS) e rede, são dispostos de maneira semelhante ao modelo tradicional de cliente-servidor. No entanto, a diferença no âmbito da aplicação é que, no modelo serverless, todos os recursos necessários para a carga de trabalho (workload) devem ser inicializados no momento da chamada. Isso significa que, ao contrário do modelo tradicional, onde recursos podem ser pré-alocados e mantidos em execução, no ambiente serverless os recursos são criados e configurados sob demanda, durante a execução de cada solicitação.

5.1 Avaliação de desempenho e escalabilidade

A arquitetura *serverless* é inerentemente escalável. Ela aumenta ou diminui automaticamente para atender à demanda. Um ambiente *serverless* só ativará os containers quando as funções forem invocadas. Como tal, ele responde automaticamente a aumentos ou diminuições de uso (Roberts, 2016).

No entanto, diagnosticar problemas de desempenho ou uso excessivo de recursos numa solução *serverless* pode ser mais difícil do que com código tradicional de servidor, porque normalmente não se tem acesso à infraestrutura em que esses serviços são executados (Eivy, 2017). Portanto, é importante monitorar e otimizar o desempenho dessas funções para garantir que elas atendam às expectativas de desempenho e escalabilidade.

Conforme observado por Roberts (2016) e Eivy (2017), toda a equipe de desenvolvimento envolvida no projeto deve possuir um certo grau de conhecimento no

provedor de nuvem utilizado. Isso implica um aumento nos custos, pois é necessário manter profissionais com essa especialização na equipe, garantindo a eficácia e a eficiência na utilização dos serviços de nuvem.

5.2 Custos e Casos de Uso em Grandes Corporações

Com o modelo *serverless*, o pagamento é feito exclusivamente pelo uso efetivo dos recursos. Isso elimina a necessidade de manutenção de servidores inativos ou subutilizados, uma vez que a cobrança é baseada no tempo de execução das funções e na quantidade de recursos consumidos (Roberts, 2016; Baldini et al., 2017). Tal abordagem gera uma redução substancial nos custos operacionais (Eivy, 2017). Contudo, é fundamental considerar que, embora o modelo serverless ofereça uma alternativa econômica para algumas aplicações, ele pode não ser a opção mais custo-efetiva para todas as cargas de trabalho. Aplicações que demandam longos períodos de execução ou uso intensivo de CPU, por exemplo, podem incorrer em custos mais elevados em uma arquitetura *serverless*.

A abordagem *Serverless* pode ser usada para uma ampla variedade de casos: aplicações *Web* e *Mobile*. Um dos casos de uso mais comuns para *Serverless*, é dado pelo uso de *Application Programming Interfaces* (APIs) de *back-end* que atendem a aplicativos da *Web* e móveis. A equipe de Engenharia de *Software* Comercial (CSE) da Microsoft fez uma parceria com um varejista global para implantar uma solução sem servidor altamente disponível nas plataformas de nuvem do Azure e do Amazon Web Services (AWS), usando a estrutura sem servidor (Castro *et al.*, 2019).

6 Desafios e Perspectivas Futuras

Além dos tópicos já discutidos por Eivy (2017) sobre custos, integração e usabilidade em arquiteturas serverless, que podem afetar o desempenho e a experiência do usuário, especialmente em aplicações sensíveis ao tempo, é importante considerar o desafio do gerenciamento de estado. Conforme destacado por Baldini et al. (2017), em uma arquitetura serverless, cada função é executada de forma independente e não retém estado entre as execuções, o que pode acrescentar complexidade ao gerenciamento do estado da aplicação.

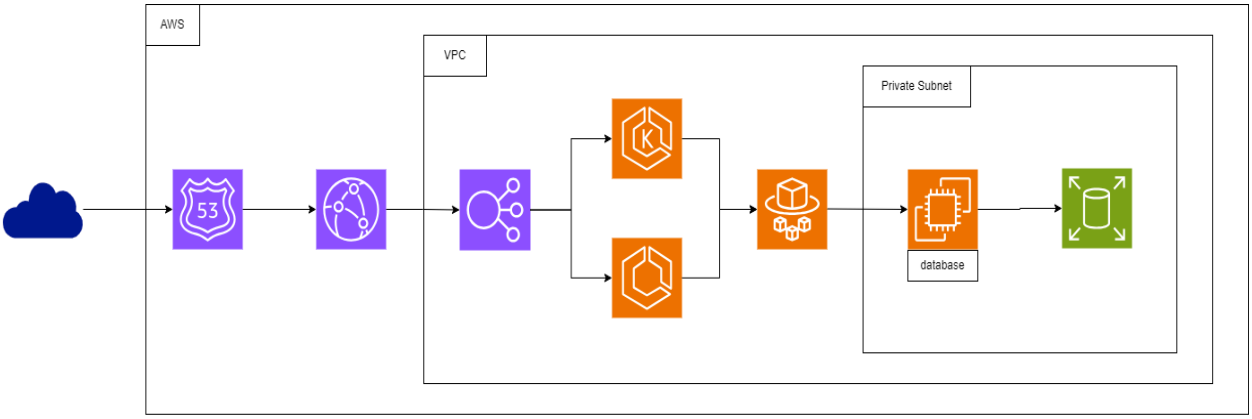
Outro desafio é a falta de visibilidade e controle sobre a infraestrutura subjacente. Isso pode tornar difícil para os desenvolvedores diagnosticar e resolver problemas de desempenho ou uso excessivo de recursos (Eivy, 2017). Além disso, questões de segurança e conformidade podem surgir, pois os dados são armazenados e processados em servidores gerenciados pelo provedor de serviços em nuvem (Roberts, 2016).

6.1 Tendências e evoluções futuras

As arquiteturas serverless estão evoluindo rapidamente, com novas tendências e tecnologias surgindo constantemente. Uma dessas tendências é o aumento do uso de contêineres em combinação com *serverless*, permitindo que os desenvolvedores empacotem suas aplicações juntamente com todas as dependências necessárias, o que pode melhorar a portabilidade e a eficiência (Baldini et al., 2017). Essa situação é melhor ilustrada na Figura 5, que mostra o uso da ferramenta AWS Fargate em conjunto com o EKS e/ou ECS. Essa combinação orquestra automaticamente os serviços de contêiner

após o cumprimento das políticas de escalonamento configuradas, sendo ideal para sistemas críticos que possuem expectativas de consumo e/ou taxas de *throughput* definidas, antes de contatar as camadas de persistência dos dados.

Figura 5: Arquitetura *serverless* com o uso de *containers*.



Fonte: Os autores.

Além disso, espera-se que as plataformas *serverless* se tornem cada vez mais integradas com serviços de inteligência artificial e aprendizado de máquina, permitindo que os desenvolvedores criem aplicações mais inteligentes e avançadas. A computação de borda também é uma área de crescimento, com *serverless* desempenhando um papel chave na habilitação de aplicações de IoT (Internet das Coisas) e móveis de baixa latência (Castro *et al.*, 2019).

Apesar dos serviços *serverless* já terem algum tempo desde seu lançamento no mercado, pelos *cloud providers*, eles ainda não são amplamente adotados como solução para a grande maioria dos negócios que tem como principal requisito alta disponibilidade em períodos de pico.

6.2 Considerações sobre a adoção de serverless em ambientes corporativos

A adoção de arquiteturas *serverless* em ambientes corporativos deve ser cuidadosamente considerada. Embora *serverless* possa oferecer benefícios significativos em termos de escalabilidade, eficiência operacional e redução de custos, também apresenta desafios únicos que as organizações devem estar preparadas para enfrentar (Roberts, 2016).

Por exemplo, a migração para uma arquitetura *serverless* pode exigir uma mudança significativa na forma como as equipes de desenvolvimento operam, pois elas precisarão se adaptar a novos modelos de programação e ferramentas. Além disso, questões de segurança e conformidade devem ser cuidadosamente avaliadas, especialmente para organizações que lidam com dados sensíveis (Eivy, 2017).

Ao considerar o conceito de delegar responsabilidades, no uso de uma aplicação front-end que dispensa servidores, o armazenamento de conteúdo estático no Amazon S3 pode ser interpretado como uma abordagem *serverless*. Embora o S3 não seja originalmente um serviço *serverless*, com expresso na página de vendas do recurso na

Amazon Web Services (AWS), é possível integrá-lo com o Route 53 e o CloudFront, oferecendo acesso eficiente e de baixo custo para hospedagem de sites e indexação de arquivos estáticos.

Finalmente, é importante lembrar que *serverless* pode não ser a solução certa para todas as aplicações. Por exemplo, aplicações que requerem longos períodos de execução ou uso intensivo de CPU podem ser mais caras para executar em uma arquitetura *serverless* (Baldini *et al.*, 2017). Assim, uma abordagem híbrida, que combina elementos de arquiteturas *serverless* e tradicionais, pode representar a solução mais eficaz em certos cenários, especialmente na delegação de atividades, como exemplificado pelo uso do recurso Amazon S3 no parágrafo acima.

7 Revisão bibliográfica

A Nordstrom, uma renomada rede de lojas de departamento de moda dos Estados Unidos, enfrentava desafios críticos em sua infraestrutura de Tecnologia da Informação (TI), especialmente durante eventos sazonais de vendas, como *Black Friday* e *Cyber Monday*. Nesses períodos de alta demanda, a infraestrutura monolítica e provisionada de forma estática da Nordstrom falhava em acompanhar o aumento exponencial no tráfego e a necessidade de processamento em tempo real de grandes volumes de dados para análises e tomadas de decisão.

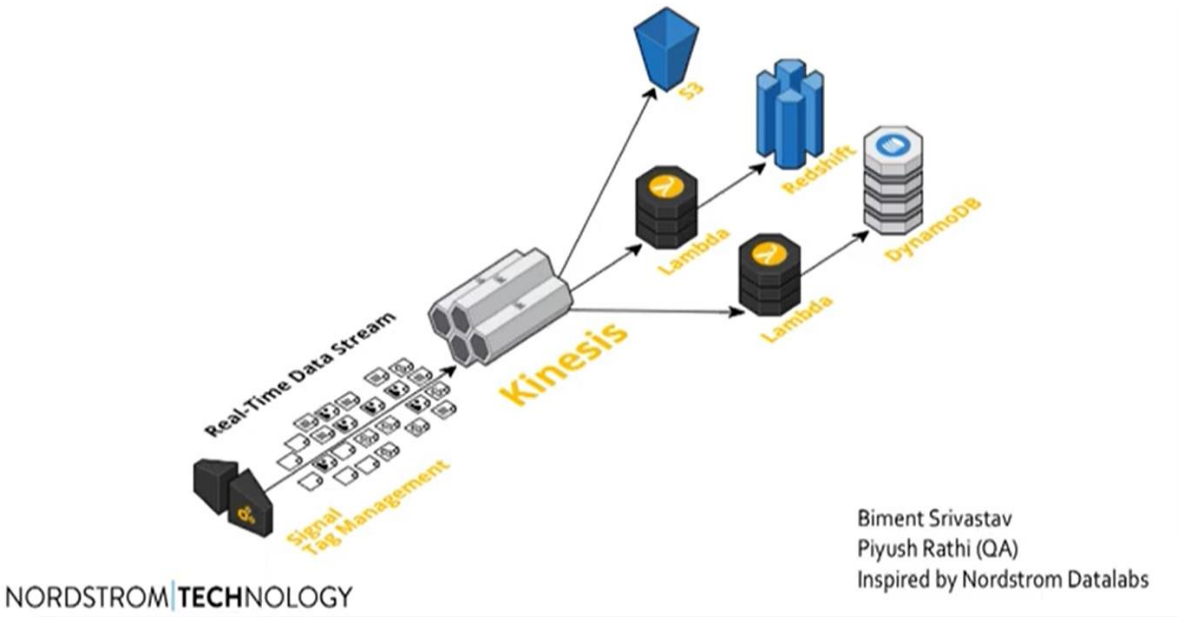
7.1 Descrição da arquitetura utilizada

Para resolver esse grave problema de latência, como expresso pelo canal de telecomunicações Serverless (2017), em uma de suas publicações, a Nordstrom optou por reestruturar parte de sua arquitetura utilizando uma abordagem *serverless*, com foco em AWS Lambda, parte do ecossistema de serviços gerenciados da Amazon Web Services (AWS). A migração envolveu a reescrita de componentes críticos da *pipeline* de análise de dados para serem executados como funções Lambda, aproveitando o modelo de execução *event-driven* (baseado em eventos), onde funções são acionadas por gatilhos específicos.

A solução foi desenhada com base em uma arquitetura de microsserviços e eventos distribuídos (Figura 6), em que as funções Lambda foram encapsuladas para realizar operações discretas e de curta duração, como processamento de lotes de dados ou execução de cálculos analíticos. Essas funções eram disparadas automaticamente por eventos no Amazon Kinesis (um serviço de processamento de dados em tempo real) ou por novos dados sendo armazenados no Amazon S3. Além disso, o uso do Amazon DynamoDB para armazenamento NoSQL proporcionou uma baixa latência e alta disponibilidade dos dados, complementando o *pipeline* de análises em tempo real (Serverless, 2017).

Ao adotar uma arquitetura baseada em Lambda, a Nordstrom eliminou a necessidade de gerenciar infraestrutura física ou virtual, já que o AWS Lambda escala automaticamente, com base na demanda e na quantidade de eventos recebidos. Essa escalabilidade horizontal instantânea foi um fator chave para o sucesso da aplicação durante os picos de tráfego.

Figura 6: Arquitetura *serverless* para processamento de dados volumétricos da empresa Nordstrom.



Fonte: Serverless (2017).

7.2 Resultados obtidos e lições aprendidas

Com a adoção da computação *serverless*, a Nordstrom, assim como empresas como Netflix e Coca-Cola, obteve os seguintes resultados:

1. **Elasticidade automática:** a capacidade inerente de escalar horizontalmente com base na demanda permitiu que a Nordstrom processasse milhões de eventos por segundo durante os picos sazonais, sem a necessidade de provisionar servidores ou ajustar manualmente a capacidade da infraestrutura, ou também com políticas de *auto-scaling* baseadas no fluxo de acesso por determinado tempo.
2. **Eficiência de custos:** o modelo de faturamento do AWS Lambda é baseado no *pay-as-you-go*, ou seja, cobra-se apenas pelo tempo de execução das funções e pela quantidade de memória utilizada. Isso resultou em uma redução dos custos operacionais, especialmente fora dos períodos de pico, já que não era necessário manter a infraestrutura ociosa com a expectativa de que certos graus de *throughput* sejam atingidos.
3. **Agilidade no desenvolvimento:** a abstração da infraestrutura proporcionada pelo lambda permitiu aos engenheiros da Nordstrom focarem mais no desenvolvimento de novas funcionalidades e menos na manutenção de servidores ou na configuração de ambientes. Isso resultou em ciclos de desenvolvimento mais rápidos, promovendo uma cultura de DevOps e automação contínua.
4. **Tolerância a falhas:** o uso de serviços gerenciados com alta disponibilidade e redundância, como DynamoDB e Kinesis, aumentou a resiliência do sistema como um todo. A Nordstrom também implementou práticas de *retry* automático e

detecção de erros nas funções Lambda para garantir maior confiabilidade durante falhas temporárias.

Esta revisão bibliográfica ilustra como a adoção de uma arquitetura *serverless* permitiu à Nordstrom superar desafios relacionados à escalabilidade, desempenho e custo durante eventos de vendas sazonais, ao mesmo tempo em que aprimorou sua capacidade de inovar com agilidade e confiabilidade.

8. Considerações Finais

Em resposta ao questionamento apresentado na introdução deste artigo, optar por *serverless* em vez de serviços tradicionais de *back-end* oferece vantagens significativas, como escalabilidade automática e redução de custos, já que você paga apenas pelo uso efetivo.

No entanto, desafios como latência de inicialização e dificuldades na diagnose de problemas de desempenho devem ser considerados, pois funções que demandam configurações extensas devem ser modularizadas em pedaços menores para que caibam dentro do tempo máximo de execução oferecido pelos provedores desse serviço.

Assim, o uso de *serverless* deve levar em conta as necessidades específicas da aplicação atual e o contexto da empresa que deseja usar este novo padrão de arquitetura de *software*, como o acoplamento entre funcionalidades internas, bem como o fato que muitas companhias ainda deverão migrar, ao menos em partes, do estágio de nuvem privada para híbrida a fim de que possam usufruir dessas funcionalidades (provindas de ambientes remotos).

Conclui-se, portanto, que computação *serverless* proporciona escalabilidade e economia. Além disso, tendências emergentes, como o uso de contêineres e a integração com inteligência artificial, visam aprimorar o atual modelo. Com a revisão bibliográfica do caso Nordstrom é notório como *serverless* pode ser eficaz para lidar com picos de demanda, embora não seja a solução ideal para todas as aplicações (a depender dos requisitos de negócio). Uma avaliação cuidadosa é essencial para determinar a adequação do modelo para cada caso específico.

Referências

Andrikopoulos, V.; Binz, T.; Leymann, F.; Strauch, S. How to adapt applications for the cloud environment. Computing, v. 95, n. 6, p. 493-535, 2013.

Aws. Coca-Cola Freestyle Launches Touchless Fountain Experience in 100 Days Using AWS Lambda. Disponível em: https://aws.amazon.com/solutions/case-studies/coca-cola-freestyle/. Acesso em: 25 jul. 2024. Publicado em 2020.

Baldini, I. *et al.* Serverless Computing: Current Trends and Open Problems. In: Research Advances in Cloud Computing. [S.l.: s.n.], 2017.

Buyya, R.; Yeo, C. S.; Venugopal, S.; Broberg, J.; Brandic, I. Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation Computer Systems, v. 25, n. 6, p. 599-616, 2009.

Catteddu, D.; Hogben, G. Cloud computing: benefits, risks and recommendations for information security. European Network and Information Security Agency, v. 17, n. 7, p. 1-125, 2009.

Castro, P. *et al.* The rise of serverless computing. Communications of the ACM, v. 62, n. 12, p. 44–54, 2019.

Eivy, R. Be Wary of the Economics of “Serverless” Cloud Computing. IEEE Cloud Computing, v. 4, n. 2, p. 6–12, 2017.

Mell, P.; Grance, T. The NIST definition of cloud computing. NIST special publication, v. 800, n. 145, p. 7, 2011.

Roberts, M. Serverless Architectures. MartinFowler.com, 2016.

Serverless. Towards a serverless event-sourced Nordstrom -- Rob Gruhl. YouTube, 2017. Disponível em: https://www.youtube.com/watch?v=WcCErxLKR7g. Acesso em: 18 jun. 2024.

Zhang, Q.; Cheng, L.; Boutaba, R. Cloud computing: state-of-the-art and research challenges. Journal of Internet Services and Applications, v. 1, n. 1, p. 7-18, 2010.