INTERFACE TECNOLÓGICA

ARQUITETURA DE MICROSSERVIÇOS

MICROSSERVICES ARCHITETURE

Danilo Takeo Kanizawa – danilot.kanizawa@gmail.com
Faculdade de Tecnologia de Taquaritinga (Fatec) – Taquaritinga – SP – Brasil

Giuliano Scombatti Pinto – giuliano.pinto@fatec.sp.gov.br
Faculdade de Tecnologia de Taquaritinga (Fatec) – Taquaritinga – SP – Brasil

DOI: 10.31510/infa.v19i2.1538
Data de submissão: 01/09/2022
Data do aceite: 28/11/2022
Data da publicação: 20/12/2022

RESUMO

A arquitetura de microsserviços é uma alternativa às arquiteturas monolíticas no ramo de Engenharia de software, onde cada contexto que compõe as funcionalidades do sistema é um microsserviço independente. Essa abordagem permite que cada microsserviço seja desenvolvido utilizando os recursos e linguagens que fazem mais sentido dentro do seu próprio contexto, além dessa flexibilidade no momento de desenvolver os microsserviços, sistemas desenvolvidos utilizando a arquitetura de microsserviços são mais escaláveis e tem maior manutenibilidade. Microsserviços permitem também uma curva de aprendizado menor das funcionalidades, uma vez que em geral cada microsserviço está sob responsabilidade de um time diferente e muitas vezes especializado naquele contexto da aplicação. A comunicação, o entendimento da arquitetura e os recursos utilizados para fazer com que os microsserviços trabalhem de maneira sinérgica e agregativa, no entanto são pontos onde o uso da arquitetura de microsserviços é mais trabalhosa e de difícil aprendizado em relação a uma arquitetura monolítica. O uso da arquitetura de microsserviços com sucesso depende de requisitos como o tamanho do sistema, a necessidade de variabilidade tecnológica e a capacidade dos membros dos times que vai empregar a arquitetura de assimilar e empregar de maneira correta os padrões e boas práticas desse modelo.

Palavras-chave: Microsserviços. Arquitetura. Funcionalidades. Contexto.

ABSTRACT

The microservices architecture is an alternative to monolithic architectures in the field of Software Engineering, where each context that composes the system's functionalities is an independent microservice. This approach allows each microservice to be developed using the resources and languages that make the most sense within its own context, in addition to this flexibility when developing microservices, systems developed using the microservices architecture are more scalable and allow for greater maintainability. Microservices also allow a lower learning curve of functionality, since in general each microservice is under the responsibility of a different team and often specialized in that application context.

Communication, understanding of the architecture and the resources used to make microservices work in a synergistic and aggregative way, however, are points where the use of the microservices architecture is more laborious and difficult to learn in relation to a monolithic architecture. The successful use of the microservices architecture depends on requirements such as the size of the system, the need for technological variability and the ability of the members of the times that will employ an assimilation architecture and correctly employ its standards and best practices.

Keywords: Microservices. Architecture. Functionalities. Context.

1 INTRODUÇÃO

A arquitetura de microsserviços é uma abordagem arquitetural que se põe como uma alternativa a abordagem arquitetural monolítica, diferente da mesma, a abordagem de microsserviços trata de isolar um único contexto da aplicação em uma unidade autossuficiente de software, com seus próprios recursos, como por exemplo banco de dados, serviços de cache etc.

Segundo Fowler (2014) o termo Arquitetura de Microsserviços ganhou espaço nos últimos tempos como uma forma alternativa de estruturar e construir aplicações como unidades de serviços independentes e autônomas. Não existe uma definição formal que descreva essa maneira de arquitetar aplicações, porém existem algumas características comuns que giram em torno das aplicações que utilizam essa arquitetura.

Para entender o padrão arquitetural de microsserviços é necessário primeiro entender o padrão ao qual ele se apresenta como alternativa, o monolítico. O padrão monolítico é uma abordagem onde todos os contextos necessários para o funcionamento da aplicação estão encapsulados em uma única entidade indivisível que será executada em sua totalidade. A abordagem monolítica não é pior ou melhor do que a abordagem de microsserviços, mas possui um contexto diferente que se encaixa melhor em escopos menores e menos complexos, afinal não é necessário construir uma estrutura de microsserviços complexa e onerosa em questões de desenvolvimento se a aplicação é pequena e simples.

O objetivo desse artigo é dissertar principalmente sobre a arquitetura de microsserviços, mas também abordar um pouco da arquitetura monolítica a fins comparativos. Para isso foram realizadas pesquisas bibliográficas em artigos, websites e livros.

2 MODELOS ARQUITETURAIS DE SOFTWARE

A abordagem de construção de aplicações em microsserviços nada mais é do que um modelo arquitetural de software que tem como objetivo guiar as pessoas envolvidas no desenvolvimento, quais são os padrões e boas práticas utilizadas para que haja sinergia e coesão no desenvolvimento das funcionalidades e quais são os componentes computacionais que integram o sistema para gerar valor ao usuário.

A definição de Shaw (1996) sobre arquitetura de software diz que ela define a estrutura do sistema em termos de componentes computacionais e que ainda é necessário definir como estes componentes interagem entre si.

Jazayeri em 2000 complementou a definição dada por Shaw dizendo que a arquitetura de software é uma ferramenta para tratar a dificuldade que um software imprime durante seu desenvolvimento e frisa que além de cumprir o papel dito por Shaw ela deve atingir os requisitos funcionais e não funcionais do sistema.

Existem muitos modelos e submodelos arquiteturais de software utilizados no mercado e cada abordagem responde melhor a um determinado tipo de situação.

2.1 O modelo arquitetural monolítico

Em um sistema de arquitetura monolítica geralmente existe uma única aplicação que é responsável por todas as funcionalidades. Essa característica faz com que quando for realizada uma atualização ou correção em uma parte do sistema, todas as demais partes precisem ser reescritas.

Grande parte das aplicações utilizadas hoje ainda são monolíticas pela facilidade que essa abordagem traz de desenvolver sem demandar tanto esforço da equipe na hora de refletir e decidir questões de arquitetura, pode-se dizer que é um modelo mais simplista de se desenvolver aplicações. Outra característica que corrobora para o uso da arquitetura monolítica é a simplicidade na hora de realizar os testes e de encontrar falhas nos processos de debug.

Um sistema baseado em uma arquitetura monolítica em geral possui apenas um banco de dados para todas as funcionalidades o que faz com que o sistema tenha uma manutenção facilitada e uma etapa de *deploy* mais simples também.

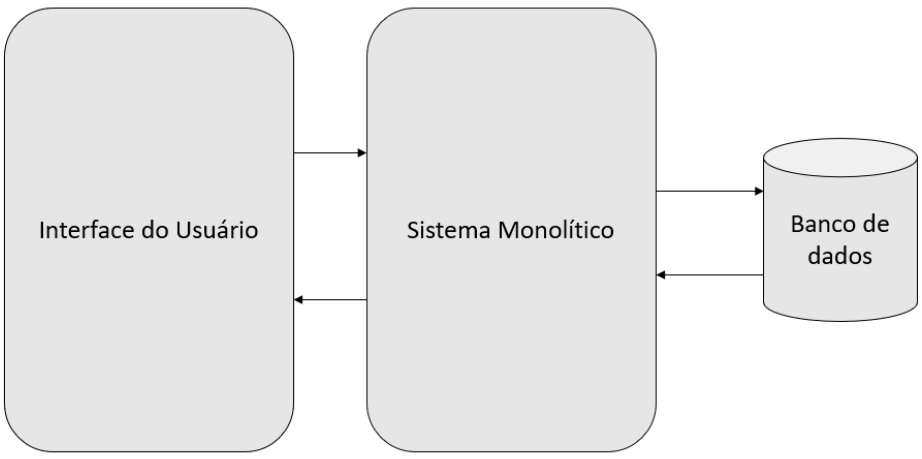
INTERFACE TECNOLÓGICA

Segundo Richardson (2016) uma das vantagens de se trabalhar com a arquitetura monolítica além das já citadas é a abundância de recursos e frameworks que se têm para facilitar o desenvolvimento enquanto a arquitetura de microsserviços ainda encontra uma escassez desses mesmos recursos e muitas vezes torna os desenvolvedores reféns das mesmas ferramentas quase que de forma inflexível.

Mesmo com todas essas vantagens e facilidades citadas que a arquitetura monolítica traz, um sistema monolítico muito grande, com muitas funcionalidades e contextos pode causar problemas de manutenção e de aprendizado sobre a implementação de suas regras de negócio, tornando muitas vezes a abordagem inviável. Não existe um número mágico de contextos ou de funcionalidades que viabilize uma abordagem arquitetural de microsserviços ou monolítica, isso deve ser analisado com parcimônia dentro de cada caso específico e a equipe desenvolvedora deve entender qual é a melhor abordagem para tratar o desenvolvimento e ainda que o sistema seja desenvolvido a princípio de maneira monolítica, existem diversas abordagens de como migrar um sistema monolítico para a arquitetura de microsserviços depois de já implementado.

A Figura 1 mostra como é o funcionamento de um sistema monolítico em geral, onde toda a responsabilidade lógica de acesso e distribuição dos dados para a interface do usuário está concentrada em um único sistema.

Figura 1



Fonte: Elaborada pelo autor (2022)

A abordagem arquitetural monolítica também tem demonstrado algumas dificuldades de bom funcionamento em tecnologias mais atuais como *Cloud Computing* e *Containers*, segundo Fowler (2014) aplicativos monolíticos podem ser bem-sucedidos, mas cada vez mais as pessoas estão se frustrando com eles - especialmente à medida que mais aplicativos estão sendo implantados em nuvem.

2.2 Arquitetura de Microsserviços

A arquitetura de microsserviços é uma alternativa relativamente recente às abordagens monolíticas, a ideia central dessa arquitetura é isolar os contextos que compõe as funcionalidades do sistema em pequenos serviços que isolados não tem muita utilidade, mas quando integrados eles conseguem gerar o valor esperado para os usuários do sistema.

Segundo Stoiber (2017) um microsserviço é uma única unidade autocontida que, juntamente com muitas outras, compõe um grande aplicativo. Ao dividir seu aplicativo em pequenas unidades, cada parte dele é implantável e escalável de forma independente, pode ser escrita por diferentes equipes e em diferentes linguagens de programação e pode ser testada individualmente.

Newman (2015) cita que microsserviços são uma forma de desenvolver sistemas distribuídos que fazem uso de serviços divididos com suas próprias funções e ciclos de vida que quando funcionando de maneira sinérgica geram valor para o usuário.

Uma aplicação pode ser desenvolvida desde o princípio em microsserviços (*Microservice First*) ou também é possível decompor uma aplicação monolítica já pronta em microsserviços (*Monolith First*). Hoje já existem diversos padrões e estudos a respeito da migração de uma arquitetura monolítica para uma arquitetura de microsserviços graças a popularidade que a abordagem de microsserviços ganhou nos últimos anos com o avanço das tecnologias de *Cloud Computing*.

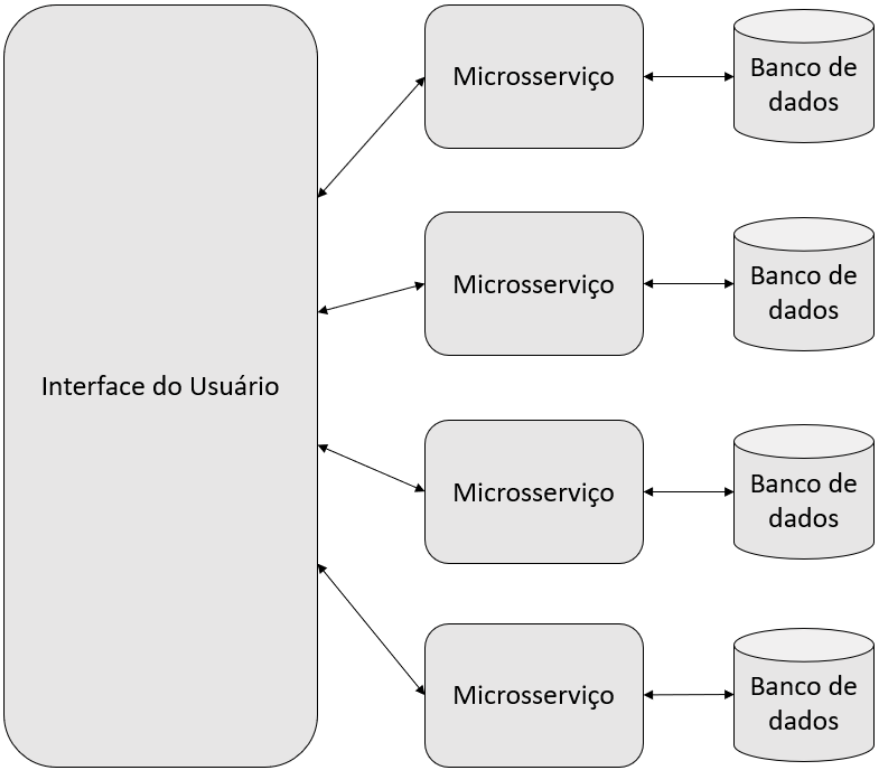
Outra diferença notável entre a arquitetura monolítica e a arquitetura de microsserviços está na camada de dados, um sistema monolítico possui um único banco de dados que centraliza os registros para todos os contextos do sistema, na arquitetura de microsserviços, cada microsserviço possui o seu próprio banco de dados.

Embora teoricamente pareça uma diferença simples e de fácil compreensão, a arquitetura de microsserviços na prática demanda mais tempo, energia e expertise tanto das pessoas que desenvolvem quanto das pessoas que modelam a arquitetura do sistema, fazer

vários sistemas isolados se integrarem e interagir da maneira correta é um desafio maior do que parece apenas vendo as diferenças teóricas entre as abordagens, muitas adversidades surgem quando se pensa em um software orientado pela arquitetura de microsserviços.

A Figura 2 ilustra o funcionamento de uma aplicação desenvolvida para funcionar com base em microsserviços.

Figura 2



Fonte: Elaborada pelo autor (2022)

2.2.1 – Motivação da arquitetura de microsserviços

Mesmo que o emprego da arquitetura de microsserviços tenha vários benefícios, é importante ressaltar que ela não é uma solução para todos os problemas e não deve ser utilizada em todos os casos.

Atkisson (2017) cita que os desafios da arquitetura monolítica já são bem compreendidos pela comunidade e que durante o desenvolvimento de uma aplicação monolítica devemos pensar em migrar para a arquitetura de microsserviços se encontrarmos os seguintes problemas ou bloqueios:

- **Tamanho:** se a aplicação está ficando muito grande a ponto de dificultar o entendimento ou até mesmo o carregamento em ambiente de desenvolvimento, deve-se pensar em decompor essa aplicação em microsserviços.
- ***Stack:*** se os recursos da *stack* utilizada para desenvolver o sistema em um modelo monolítico estão encontrando limitações ou dificuldades, deve-se pensar em migrar para o modelo de microsserviços já que nesse caso pode-se utilizar a *stack* adequada para cada serviço.
- **Escalabilidade:** em um sistema monolítico só é possível realizar uma escalabilidade vertical, quando é preciso replicar um recurso dentro do ambiente só é possível replicar todos os recursos ao mesmo tempo mesmo sem necessidade de alguns, isso gera um custo elevado computacional, em uma aplicação de microsserviços pode-se replicar apenas o serviço que é necessário.

2.2.2 – Desafios da arquitetura de microsserviços

Embora o emprego da arquitetura de microsserviços traga muitos benefícios existem alguns desafios a se enfrentar para que essa arquitetura não se torne um problema a longo prazo. Um sistema baseado em qualquer arquitetura não deve ser construído pensando apenas no tempo de desenvolvimento e sim levando em conta também características como a manutenibilidade a longo prazo no tempo em que o sistema já se encontra em produção.

Para que isso seja garantido Felipe (2020) cita alguns dos desafios ao utilizar a arquitetura de microsserviços no desenvolvimento de aplicações:

- **Consistência dos dados:** quando temos dados sendo acessados e registrados por vários serviços diferentes podemos ter inconsistências de dados entre os serviços, se o modelo de replicação dos dados entre os serviços não for adequado ou se houver atraso na publicação das alterações os usuários poderão ter inconsistências nos dados da aplicação.
- **Comunicação entre os sistemas e entre as equipes:** para que não haja inconsistência nas regras de negócio tratadas pelos microsserviços é necessária

uma comunicação bem estruturada tanto em termos síncronos quanto em termos assíncronos entre os microsserviços, o mesmo vale para a comunicação entre as equipes que tratam e implementam as regras de negócio.

- **Monitoramento:** monitoramento é importante em sistemas de qualquer arquitetura, porém quando se trata de microsserviços é algo crucial para garantir o bom funcionamento do sistema. Quando se tem vários microsserviços funcionando em sinergia para gerar valor é imprescindível que se saiba quais microsserviços estão funcionando e como estão funcionando. A forma como é feito o monitoramento dos serviços é crucial para a tomada de decisões e para realizar correções em maus funcionamentos.
- **Testes:** por envolver diversos serviços e camadas, a arquitetura de microsserviços possui um fluxo de dados muito mais complexo e de difícil entendimento as vezes, por isso realizar ou escrever testes para sistemas baseados na arquitetura de microsserviços é uma tarefa muito mais difícil do que nas aplicações monolíticas.

A visão de Atkisson (2017) também complementa os desafios na implementação da arquitetura de microsserviços com:

- **Construção:** construir microsserviços demanda uma grande capacidade analítica para identificar a dependência entre os serviços, uma vez que uma alteração em um serviço pode acarretar alterações em vários outros serviços.
- **Versionamento:** decidir como controlar a versão de uma aplicação em microsserviços não é uma tarefa fácil, é necessário analisar se isso será realizado em um repositório único ou se cada serviço terá seu próprio repositório. Cada microsserviço tem sua própria versão, mas a agregação de serviços também pode gerar uma versão própria. Fica muito mais difícil pensar na compatibilidade entre versões anteriores.
- ***Deploying:*** A etapa de deploy de uma aplicação de microsserviços deve ser obrigatoriamente automatizada, a complexidade dessa etapa é tão grande que se torna inviável realizá-la manualmente.

3 PROCEDIMENTOS METODOLÓGICOS

Para esse estudo foram utilizadas consultas bibliográficas a livros, artigos, monografias, websites e outros materiais acadêmicos a fim de levantar informações e sintetizar a compreensão da arquitetura de microsserviços e suas aplicações na atualidade.

4 RESULTADOS E DISCUSSÃO

Analisando a estrutura da arquitetura de microsserviços é possível visualizar que existem muitos pontos positivos, mas também muitas adversidades e desafios ao implementar o modelo arquitetural em um projeto.

Também é possível observar que existem pontos negativos e questões onde a arquitetura de microsserviços leva algumas desvantagens em relação a arquitetura monolítica, então é necessário pensar com parcimônia e analisar se realmente vale o esforço de utilizar ou migrar para a arquitetura de microsserviços.

4.1 Pontos positivos da arquitetura de microsserviços

Entre as características da arquitetura de microsserviços Felipe (2020) observa:

- **Heterogeneidade tecnológica:** utilizando a arquitetura de microsserviços é possível utilizar várias tecnologias, o que abre um leque de opções para utilizar a tecnologia adequada para cada contexto da aplicação.
- **Escalabilidade:** quando os contextos da aplicação estão separados em microsserviços é possível escalar as instancias dos serviços de maneira mais precisa o que gera um melhor aproveitamento dos recursos computacionais dos servidores ou clusters.
- **Facilidade na publicação:** ter serviços menores e autônomos facilita o processo de *build*, uma vez que não é necessário realizar o *build* de toda a aplicação, o que gera um processo de publicações e *rollbacks* mais rápidos.
- **Resiliência:** ter serviços independentes e autônomos aumenta a resistência a falha do sistema como um todo, em um sistema monolítico a falha de uma componente pode levar a falha de todo o sistema, com microsserviços bem desenvolvidos, cada serviço lida com a falha de maneira isolada sem comprometer toda a aplicação.

4.2 Pontos negativos da arquitetura de microsserviços

- **Complexidade:** a arquitetura de microsserviços é muito mais complexa do que a arquitetura monolítica, demandando uma maior expertise e maturidade do time, caso não existam razões claras para migrar a aplicação para microsserviços o mesmo não deve ser feito.
- **Comunicação entre os microsserviços:** definir mal qual vai ser a maneira com que os microsserviços vão se comunicar pode causar diversos problemas na aplicação, definir como e quando tratar as comunicações de maneira síncrona ou assíncrona é uma tarefa complexa que muitas vezes pode implicar em problemas para a experiência do usuário.
- **Comunicação entre os times responsáveis:** as equipes que trabalham nos microsserviços de uma aplicação devem estar bem alinhadas para que não haja problemas de governança e devem gerenciar bem os conflitos que houverem entre os interesses de cada parte, caso contrário o mau relacionamento entre as equipes será refletido no funcionamento do sistema.

5 CONSIDERAÇÕES FINAIS

Após a realização desse estudo é possível notar que a arquitetura de microsserviços surgiu para sanar problemas e limitações que começaram a surgir durante o uso da arquitetura monolítica. Isso não quer dizer que todas as aplicações monolíticas apresentam problemas ou que este é um modelo fadado ao fracasso e sim que ele possui limitações que devem ser avaliadas e que se caso haja sentido deve-se migrar para a arquitetura de microsserviços.

A arquitetura de microsserviços é muito mais complexa e demanda uma curva de aprendizado muito mais acentuada do que a arquitetura monolítica, por tanto ela não é uma arquitetura que deva ser utilizada por times inexperientes e imaturos.

Trabalhar com microsserviços é muito mais do que escrever código ou implementar soluções lógicas, para que um projeto de microsserviços prospere é necessário pensar em aspectos de negócio, infraestrutura e até mesmo de equipe, caso contrário o projeto pode fracassar por ter pontos que ainda não estão prontos para trabalhar com a arquitetura de microsserviços.

Quando todos os requisitos para trabalhar com microsserviços são atingidos esse modelo pode levar o projeto a um grau de maturidade muito elevado em que seria muito difícil atingir utilizando a arquitetura monolítica.

REFERÊNCIAS

ATKISSON, Brian. **The Truth about Microservices.** 4 mai. 2017. Disponível em: https://developers.redhat.com/blog/2017/05/04/the-truth-about-microservices?extIdCarryOver=true&sc_cid=701f2000001OH6fAAG. Acesso em 14 ago. 2022.

AWS. **O que são Microsserviços** Disponível em: https://aws.amazon.com/pt/microservices/. Acesso em 18 de ago. 2022

FELIPE, Luis. **Introdução a Microsserviços.** 26 set. 2020. Disponível em : https://www.luisdev.com.br/2020/12/26/introducao-a-microsservicos/?gclid=EAIaIQobChMIi-vBsfG0-gIVCj-RCh2_mg9dEAAYBCAAEgLEffD_BwE. Acesso em 13 ago. 2022

FOWLER, Martin. **Microservices - a definition of this new architectural term.** 25 mar .2014 Disponível em: https://martinfowler.com/articles/microservices.html. Acesso em 11 ago. 2022.

JAZAYERI, Mehdi. **Software Architecture for Product Families: Principles and Practice.** 15 jan. 2000. 1ª ed. Addison-Wesley, 2000

NEWMAN, Sam. **Building Microservices: Designing Fine-Grained Systems** 20 fev. 2015. 1ª ed. O'Reilly Media, 2015.

RICHARDSON, Chris. **Pattern: Microservice Architecture.** Disponível em: https://microservices.io/patterns/microservices.html. Acesso em 16 ago. 2022.

SHAW, Mary. **Software Architecture. Perspective on an Emerging Discipline.** 12 abr. 1996. 1ª ed. Prentice Hall, 1996.

STOIBER, Max. **Build your first Node.js microservice.** 11 jan. 2017. Disponível em: https://mxstbr.blog/2017/01/your-first-node-microservice/. Acesso em 16 ago. 2022.