Arquitetura de Microsserviços
Conceituação, Projeto e Implementação em Java com o Framework Spring Boot

André Luis Jorge Paulino

Orientador: Prof. Me. Luiz Alberto Ferreira Gomes

Departamento de Ciência da Computação
Pontifícia Universidade Católica de Minas Gerais (PUC Minas)
37.701-355 – Poços de Caldas – MG – Brasil

andre.paulino@sga.pucminas.br

Abstract. *This work proposes to present the concept of microservices architecture, its application and functioning, in addition to showing its architectural model through an implementation in Java with the Spring Boot framework. To this end, its methodologies, advantages, and an implementation of this architectural model as a proof of concept were elaborated, enabling the demonstration of its behavior. Thus, it is concluded that the microservice architecture is a great progress in software development, because through it is possible to deal with issues relevant to the growth of the application.*

Resumo. *Este trabalho tem como proposta apresentar o conceito da arquitetura de microsserviços, sua aplicação e funcionamento, além de exibir seu modelo arquitetural mediante a uma implementação em Java com o framework Spring Boot. Para esse fim foram abordadas suas metodologias, vantagens, e elaborada uma implementação desse modelo de arquitetura como prova de conceito, possibilitando a demonstração de seu comportamento. Sendo assim, conclui-se que a arquitetura de microsserviços é um grande progresso no desenvolvimento de software, pois através dela é possível tratar questões relevantes ao crescimento da aplicação.*

Palavras-chave: *Microsserviços, Arquitetura, Software, Implementação.*

1. Introdução

O avanço da tecnologia também alavancou a necessidade de encontrar novas maneiras de se produzir aplicações, devido a isso, tem sido um grande desafio desenvolver softwares capazes de utilizar diferentes recursos e lidar com grandes demandas de requisições.

Em um cenário empresarial é importante atuar com ferramentas corporativas e lidar com regras de negócio, para que então, seja possível desenvolver um software que atenda todas as condições. Diante desse contexto e visando uma ampliação e/ou inovação do software, modelos convencionais podem trazer grandes barreiras, tornando necessária a melhoria de diversas questões como escalabilidade, produtividade, falhas, entre outros casos, que certamente impactariam nesse processo. A arquitetura de microsserviços surge nesse momento como uma evolução das arquiteturas tradicionais, tratando fatores essenciais ao crescimento e mudanças da aplicação.

Este trabalho tem como objetivo a conceituação da arquitetura de microsserviços, a apresentação do modelo e funcionamento da arquitetura, e a demonstração através de

uma implementação em Java com o framework Spring Boot, de modo a ser possível entender seu comportamento e atuação.

O artigo foi dividido da seguinte forma: (1): Introdução, que aborda o contexto de microsserviços, além do objetivo do trabalho. (2): Conceituação básica, que explica precisamente o termo e suas características. (3): Arquitetura, que detalha seu funcionamento e sua usabilidade, além de relacioná-lo com arquiteturas tradicionais. (4): Vantagens, que apresenta os benefícios de sua utilização. (5): Desvantagens, que alerta as precauções ao implantar microsserviços. (6): Trabalho desenvolvido, que descreve como foi implementado, juntamente com o modelo e as ferramentas utilizadas. (7): Considerações Finais, que conclui o objetivo do trabalho. (8): Referências, as fontes utilizadas para fundamentar a pesquisa.

2. Conceituação básica

Microsserviço é uma inovação na arquitetura de software, que corresponde a construção de aplicações dividindo-as em serviços independentes. Através deles há a possibilidade de utilização de métodos ágeis, integração e entrega contínua, entre outros, que permitem a criação de serviços otimizados em grande escala. Nesta seção será abordada de maneira precisa o conceito de microsserviços.

2.1 Conceito de microsserviços

O termo “microsserviço” foi discutido em um workshop de arquitetos de software em maio de 2011 para descrever o que os participantes viram como um estilo arquitetônico comum que muitos deles haviam explorado recentemente. Em maio de 2012, o mesmo grupo decidiu por “microsserviços” como o nome mais adequado. Segundo Martin Fowler e James Lewis (2014) o termo “Arquitetura de microsserviço” surgiu nos últimos anos para descrever uma maneira particular de projetar aplicativos de software como conjuntos de serviços implantáveis independentemente.

Precisamente o conceito básico é: microsserviço é uma pequena aplicação que executa uma única tarefa e o realiza com eficiência. Ele é um pequeno componente que pode ser facilmente substituído, sendo desenvolvido e implantado de forma independente. Jon Eaves, do RealEstate.com.au na Austrália, expressa que microsserviço é: “algo que possa ser reescrito em duas semanas”, um princípio muito conectado ao seu cenário. Os microsserviços são componentes separados que trabalham juntos para realizar as mesmas tarefas e por esse fato, a implantação e a escalabilidade podem ser tratadas de acordo com a demanda. Toda essa funcionalidade permite que os serviços sejam escritos em diferentes linguagens de programação e diferentes tecnologias, além de que cada serviço pode ser gerenciado por diferentes times de desenvolvimento, possibilitando a formação de equipes especializadas.

Na tradicional abordagem monolítica toda a aplicação é criada como um único bloco, há uma dependência entre os serviços. Essa dependência dificulta as alterações e atualizações, pois ao realizar mudanças em partes específicas da aplicação há riscos de ocasionar falhas no funcionamento de outros serviços. A utilização de microsserviços se mostra como um avanço comparado a modelos tradicionais, sanando fatores que até então ofereciam ameaças.

3. Arquitetura

Arquitetura de microsserviços e arquitetura monolítica são maneiras usadas para expor dois métodos de construção de aplicações e softwares. Para debater microsserviços é importante confrontá-lo com a arquitetura monolítica.

3.1 Arquitetura monolítica

Essa arquitetura trabalha como um sistema único, não dividido, que roda em um único processo, uma aplicação de software em que diferentes componentes estão ligados a um único programa dentro de uma única plataforma.

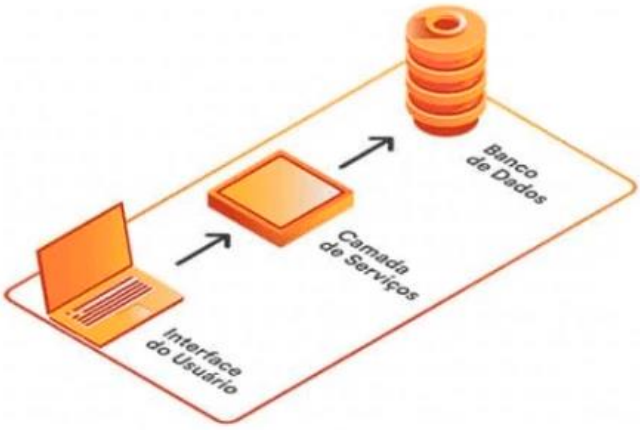

Figura 1 – Arquitetura monolítica (Fonte: supero.com.br)

Sistemas construídos sobre o padrão arquitetural monolítico são compostos basicamente por módulos ou funcionalidades agrupadas que, juntas, compõem o software. Sistemas ERP (Enterprise Resource Planning), como Tasy, Protheus, fazem uso dessa arquitetura. Estes possuem, normalmente, módulos para controle do setor financeiro, contábil, compras, entre outros, agrupados numa única solução que acessa o banco de dados.

> As aplicações empresariais são muitas vezes construídas em três partes principais: uma interface de usuário do lado do cliente (consistindo de páginas HTML (HyperText Transfer Protocol) e javascript executadas em um navegador na máquina do usuário) um banco de dados (consistindo de várias tabelas inseridas em um gerenciamento de banco de dados comum e geralmente relacional) e uma aplicação do lado do servidor. O aplicativo do lado do servidor tratará pedidos HTTP, executará lógica de domínio, recuperará e atualizará dados do banco de dados e selecionará e preencherá as visualizações HTML para serem enviadas para o navegador. 126 Esta aplicação do lado do servidor é um monolito – um único executável lógico (LEWIS; FOWLER, 2014).

Percebe-se que por mais que a aplicação possua diferentes módulos, por trás há apenas uma aplicação realizando essa ponte entre interface e banco de dados, ou seja, uma única solução de acesso, conforme a Figura 1.

“A aplicação monolítica pode ser difícil de se manter e modificar. Isso é especialmente verdadeiro quando o aplicativo está ficando maior. Com o crescimento da

aplicação, é difícil adicionar novos desenvolvedores ou substituir membros da equipe.” (DMITRY; MANFRED, 2014, p. 24).

Apesar de diversos fatores complicadores, a arquitetura monolítica pode ser bem-aceita em aplicações menores pois possui formas mais simples de gerir a composição do software. A implantação pode ser realizada com facilidade visto que o banco progredirá de acordo com as funcionalidades, também há melhor aproveitamento do código em razão de todo o código fazer parte da mesma unidade. Porém, conforme a aplicação tome proporções maiores será altamente prejudicada em novas integrações, a base de código se tornará complexa e influenciará diretamente na manutenção e produtividade, além de que existirá um único ponto de falha.

3.2 Arquitetura de microsserviços

Como já colocado, a arquitetura de microsserviços é composta por vários serviços em que cada serviço se comporta de forma autônoma e com diversas funcionalidades, e usualmente realizam a comunicação via APIs. Essa independência permite que esses serviços sejam implementados em diversas linguagens e tecnologias, além do mais, é permitido instalar em diferentes estações.

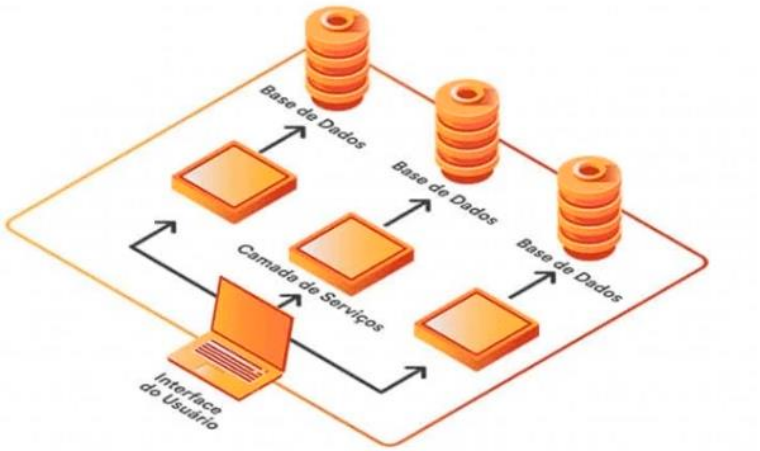

Figura 2 – Arquitetura de microsserviços (Fonte: supero.com.br)

De modo simples, a arquitetura de microsserviços trata a decomposição de aplicações monolíticas em suíte de serviços independentes, possibilitando o controle de forma autônoma. Isso garante descentralização e disponibilidade, já que não é necessário parar toda a aplicação para ajustes. Utilizar microsserviços permite facilidade na compreensão e documentação pois estão concentrados no domínio do programa, garantindo assim um melhor aproveitamento de equipe, além da possibilidade de utilização de novas ferramentas.

Em um sistema monolítico inovar em técnicas pode acarretar grandes riscos, visto que qualquer mudança é capaz de afetar boa parte da aplicação. Com microsserviços isso não é um problema, uma vez que há diversos serviços, possibilitando a escolha de um único serviço para utilização da nova tecnologia, de forma que não traga ameaças ao sistema. Arquiteturas monolíticas tendem a ficar cada vez mais difícil de serem otimizadas, em contrapartida as arquiteturas de microsserviços facilitam a escalabilidade e agilizam o desenvolvimento de aplicativos, permitindo a inovação e acelerando o tempo de introdução desses novos recursos no mercado.

4. Vantagens

Certamente, uma das maiores vantagens é a alta produtividade da equipe. Por se tratar de pequenos serviços, a base de código também é menor, com isso se torna fácil o entendimento e desenvolvimento por novos membros. Outra grande característica é a possibilidade de lançamento no mercado com mais rapidez. Como os serviços trabalham de formas autônomas, a implantação é mais acelerada e não depende de outros serviços, não exigindo que todo o software seja restabelecido. Isso faz com o que o processo seja veloz e ainda possibilite o uso de tecnologias ágeis.

Mais tolerante a falhas, pois se um elemento falhar, o restante da aplicação permanece em funcionamento, diferentemente do modelo monolítico. Além disso, permite o uso de diferentes tecnologias, dessa forma existe um potencial de escolhas que garantem o melhor desempenho em determinado serviço, que também gera um aumento no open source, por se ter liberdade diante de tantas linguagens e tecnologias.

Bancos de dados descentralizado, cada serviço pode ter o seu próprio banco de dados, proporcionando escolhas que atendam e melhoram o desempenho de cada serviço, aumentando ainda mais sua independência. Microsserviços também são altamente escaláveis, à medida que a demanda por determinados serviços aumenta, é possível fazer implantações em vários servidores e infraestruturas para que as necessidades sejam atendidas. Juntamente, proporcionam desacoplamento entre serviços, o escalonamento de recursos pode ser otimizado, uma vez que é permitido ajustar os serviços de acordo com sua capacidade de processamento.

5. Desvantagens

Apesar das grandes vantagens, existem diversos pontos de atenção que podem ser tornar desvantagens.

A equipe precisa lidar com sistemas distribuídos, sendo obrigada encarar suas complexidades. Implantar pode ser uma tarefa difícil, pois ao possuir muitos serviços, também pode haver inúmeras instancias, tornando a elaboração e gerenciamento um tanto quanto complexo. O monitoramento deve ser robusto para gerenciar efetivamente toda a infraestrutura. A descentralização dos bancos é uma questão importante, pois manter sua consistência é algo árduo. Além do mais, é necessário incorporar a cultura de DevOps, pois se torna essencial o controle pelo provisionamento de serviços e as falhas.

6. Trabalho desenvolvido

Foi desenvolvida uma aplicação composta por vários microsserviços que se comunicam de forma transparente, escalável e com balanceamento de carga. A autenticação e autorização também foram práticas utilizadas, além de testes, que deixam os microsserviços aptos para implantação. Desta forma foi possível explorar um cenário relevante sobre microsserviços, o qual será detalhado nesta seção.

6.1 Implementação

O objetivo da implementação é montar um ecossistema de microsserviços com API gateway, servidor de Discovery, configuração centralizada e microsserviços escaláveis com balanceamento de carga. Além do Spring Boot também foram utilizadas ferramentas

do Spring Cloud, como Feign para requisições de API, Ribbon para balanceamento de carga, servidor Eureka para registro dos microsserviços, entre outras.

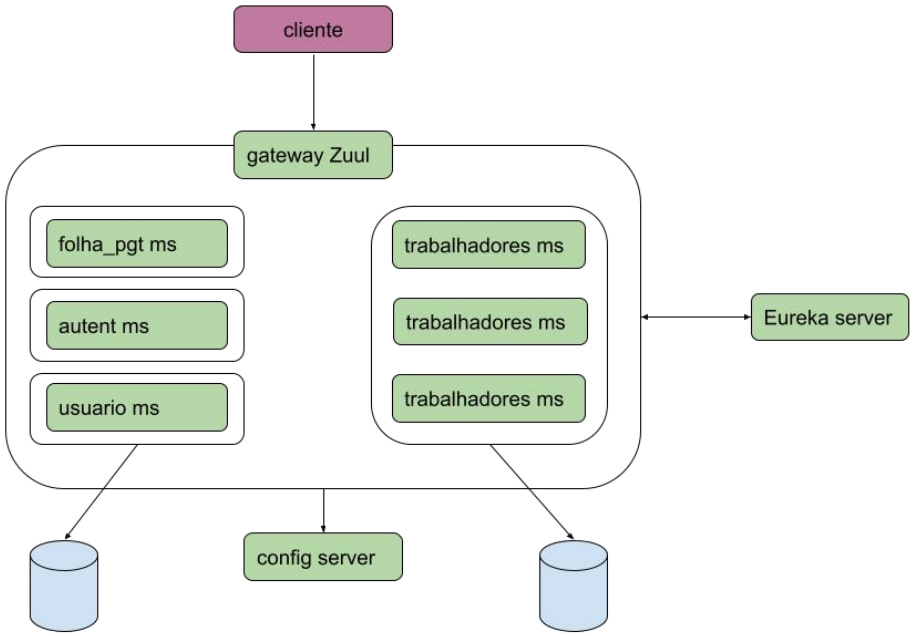

Figura 3 – Visão geral da aplicação (Fonte: elaborada pelo autor)

Há um serviço de trabalhadores que está conectado ao banco de dados e este possui 3 instâncias, ou seja, desta forma pode-se escalar o microsserviço conforme a necessidade. Existe um microsserviço de folha de pagamento, que precisa de informações do microsserviço de trabalhadores parar gerar a folha. Foi criado um microsserviço de usuários que também está conectado ao banco de dados, com cadastros de usuários e seus papéis. Por último, um microsserviço de autorização no qual será utilizado o protocolo OAuth e os tokens JWT, que são dois padrões muito populares de indústrias. Os microsserviços são registrados em um servidor de discorvery, no caso o Eureka, e foi utilizado o API Gateway para rotear os microsserviços. A Figura 3 mostrada acima exemplifica a visão da aplicação.

Nesta implementação será possível visualizar grandes características, como a facilidade de criação de instâncias, microsserviços autoescaláveis, balanceamento de carga, dentre outras. De modo geral, se torna uma infraestrutura bastante interessante.

6.2 Modelo conceitual

A ideia é montar a infraestrutura de microsserviços, com a comunicação, configuração automática, escala automática e balanceamento de carga.

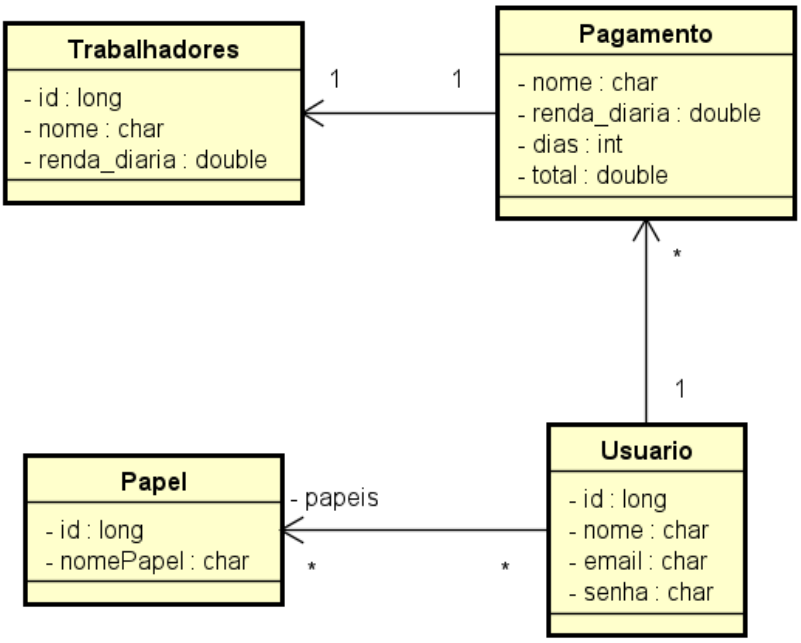

Figura 4 – Modelo conceitual (Fonte: elaborada pelo autor)

Através da entidade Trabalhadores, a entidade Pagamento consegue criar os registros de pagamento. Há também as entidades Usuários e Papéis, que são responsáveis por verificar o usuário e seu acesso.

6.3 Ferramentas utilizadas

6.3.1 REST

De forma simples, a utilização da arquitetura de microsserviços se torna possível com REST. Ela é baseada em recursos e utiliza métodos HTTP, que por sua vez, são encarregados de ocasionar modificações nos recursos, são elas: GET – recupera, POST – cria, PUT – atualiza e DELETE – apaga).

6.3.2 Spring Framework

Ele é um framework open source, a partir dele é possível implementar diversas funções, como persistência de dados, injeção de dependência, entre outras. Desta forma, ele permite facilidade no desenvolvimento em Java.

6.3.3 Spring Boot

Baseado no Spring Framework, ele é um recurso para construção de microsserviços em Java que reduz a dificuldade e proporciona mais agilidade na implementação de microsserviços. Ele é construído em módulos/starters, que são estruturas que se comunicam e são utilizadas para conceder funcionalidade a uma aplicação.

7. Considerações finais

Através do trabalho realizado conclui-se que a arquitetura de microsserviços traz muitos benefícios para o futuro do desenvolvimento de software, em razão de que contribui com serviços escaláveis, se torna acessível já que permite facilidade na integração, melhora o

desempenho e possibilita códigos mais estruturados. Deste modo, microsserviços podem gerar ganhos às empresas, que por sua vez trabalham com grandes desafios devido a utilização de arquiteturas convencionais.

Contudo, a utilização dessa arquitetura nem sempre é indicada, pois dependendo da aplicação seu uso pode ser falho. Ela gera maior complexidade no processo de implantação já que são necessárias numerosas implementações, é fundamental ser realizada a quebra por serviços e integração, além de que é preciso automatizar todo o processo da aplicação. Para sistemas monolíticos simples não há problemas em manter a arquitetura tradicional, desde que ele se mantenha desta maneira e não se expanda.

É importante analisar no planejamento de um software se há sinais de que a aplicação atingirá com o tempo níveis elevados de complexidade, para que então seja criada com a concepção de favorecer possíveis modificações no futuro. Ainda assim, caso a arquitetura de microsserviços seja utilizada de forma correta, ela pode trazer diversas vantagens frente às dificuldades impostas atualmente.

Por meio da aplicação implementada foi possível entender o funcionamento da arquitetura de microsserviços, assim como assimilar com seu conceito e o que de fato seu modelo propõe.

8. Referências

O que são microsserviços? | AWS. Amazon Web Services, Inc. Disponível em: <https://aws.amazon.com/pt/microservices/>.

MARCOSGMACHADO. Micro Serviços: Qual a diferença para a arquitetura Monolítica? | OPUS. Opus Software. Disponível em: <https://www.opus-software.com.br/micro-servicos-arquietura-monolitica/>.

MICROSERVICES. Building Microservices. O'Reilly Online Learning. Disponível em: <https://www.oreilly.com/library/view/building-microservices/9781491950340/ch01.html>.

GEISON DURÃES. Microsserviços: o que é, como fazer e por onde começar? | Blog da TecnoSpeed. Blog da TecnoSpeed. Disponível em: <https://blog.tecnospeed.com.br/microsservicos-o-que-e/>.

Microservices. martinfowler.com. Disponível em: <https://martinfowler.com/articles/microservices.html>.

NEWMAN, Sam. Building Microservices: Designing fine-grained systems. 1. ed. Califórnia: O'Reilly Media, 2015.

WESLEYWILLIANS. Microservices: O que são, e como funcionam? - Blog School of Net. Blog. Disponível em: <https://blog.schoolofnet.com/o-que-sao-microservices/>.

PRABATH SIRIWARDENA. Building Microservices - FACILELOGIN. Medium. Disponível em: <https://medium.facilelogin.com/building-microservices-designing-fined-grained-systems-d37b57a97c4e>.

DAILY CODE BUFFER. Microservices using SpringBoot | Full Example. Disponível em: <https://www.youtube.com/watch?v=BnknNTN8icw>.

JAVA TECHIE. Microservice Using Spring Boot & Spring Cloud | 2 Hours Full Course | JavaTechie. Disponível em: <https://www.youtube.com/watch?v=tljuDMmfJz8>.