Análise de Requisitos Não-Funcionais nos Processos de Migração de Sistemas Monolíticos para Microsserviços

Marcio Veronez e Ivonei Freitas da Silva

Universidade Estadual do Oeste do Paraná – Unioeste – Cascavel
Centro de Ciências Exatas e Tecnológicas – CCET
Programa de Pós-Graduação em Ciência da Computação – PPGComp
R. Universitária, 2069 – Universitário – CEP 85819–110 – Cascavel – PR – Brasil
{marcio.veronez, ivonei.silva}@unioeste.br

Resumo Requisitos não-funcionais como alta disponibilidade, redundância e escalabilidade são descritos na literatura como motivação e benefícios quando se migra de uma arquitetura monolítica para microsserviços. No entanto, não está claro quais os principais requisitos não-funcionais e seus *trade-offs* que engenheiros de requisitos e arquitetos de software devem considerar antes do processo de migração. Este trabalho descreve os principais requisitos não-funcionais e seus *trade-offs* do processo de migração de sistemas monolíticos para microsserviços. Para isto, primeiramente, as motivações e forças norteadoras que justificam as migrações de sistemas monolíticos para microsserviços são mapeadas para requisitos não-funcionais através de um mapeamento sistemático da literatura. Então, uma análise de similaridade e variabilidade entre estes requisitos não-funcionais é realizada com um modelo de *features*. Essa análise pode guiar os arquitetos de software nas decisões arquiteturais durante o processo de migração.

Keywords: microsserviço · monolítico · migração · requisitos não-funcionais. · modelo de *features*

1 Introdução

A migração de sistemas para a computação em nuvem potencializa diversos requisitos não-funcionais (RNFs), tais como alta disponibilidade, redundância, escalabilidade automática, facilidade no gerenciamento da infraestrutura e padrões de segurança [2]. Embora esses sistemas possam ser migrados com uma arquitetura monolítica, microsserviços (MS) é uma arquitetura *cloud-ready* [7] e uma alternativa para modernização de sistemas legados [8].

Um sistema monolítico é aquele onde todo o código do software é implantado como um processo único [12]. Mesmo em um sistema monolítico projetado de forma modular, ainda são compartilhados recursos como memória, banco de dados e arquivos [6]. A arquitetura monolítica pode ser uma solução viável em

vários casos, sendo que sua grande vantagem é a sua simplicidade [12]. Dentre suas principais limitações, estão a diminuição da manutenibilidade [6], implantação e escalabilidade ineficientes [6] e dificuldade na adoção de novas tecnologias [12].

Na arquitetura de microsserviços, um sistema é construído como um conjunto de pequenos serviços coesos, cada um sendo executado de forma independente e se comunicando através de protocolos leves [6]. Seu foco está na independência, facilidade de substituição e autonomia [11]. Microsserviços resolvem diversos problemas apresentados pelos sistemas monolíticos. Dentre seus principais benefícios estão a heterogeneidade de tecnologia, resiliência, escalabilidade, facilidade de implantação, reúso de serviços e facilidade de mudança [11].

Reestruturar a arquitetura de um sistema monolítico para MS pode habilitar RNFs inerentes à computação em nuvem, porém surgem novos desafios, tais como a descoberta de serviços em rede, gerenciamento da segurança, otimização da comunicação, compartilhamento de dados e performance [1], além das dificuldades com o aprendizado de novas tecnologias, o aumento da complexidade do sistema e a consequente necessidade de desenvolvedores com maior experiência [16].

Na literatura são descritas várias abordagens e experiências reais de migração para MS, as quais são motivadas e guiadas pelas denominadas "forças norteadoras"(FNs). Exemplos de FNs são: a necessidade do software ser escalável, mais seguro, coeso, e manutenível. Estas forças (ou objetivos) claramente confundem-se com RNFs (ou atributos de qualidade - AQ), os quais são definidos como requisitos que descrevem propriedades, características, restrições ou atributos que um software (sua arquitetura) precisa fornecer [10].

Como visto nesses trabalhos, há diversos RNFs indicados como importantes para o processo de migração para microsserviços o que pode potencializar conflitos entre eles. Os engenheiros de requisitos ou arquitetos de software precisam analisar os denominados *trade-offs* entre esses requisitos (ou atributos de qualidade) para tomar decisões como: "migrar ou não migrar"e "definir a arquitetura adequada para atender os RNFs". A análise de similaridades e variabilidades [3] entre esses RNFs poderia ser realizada para entender quais requisitos são obrigatórios para todos os cenários de migração para microsserviços, quais requisitos seriam opcionais e qual RNF requer a presença (ou ausência) de outro requisito. Além disso, essa análise permite especificar configurações de RNFs (obrigatórios e opcionais selecionados) as quais podem ser adequadas para cenários específicos de migração, reduzindo riscos e potencializando benefícios.

O objetivo geral deste estudo é descrever os requisitos não-funcionais e seus *trade-offs* que motivam e guiam a migração de sistemas monolíticos para microsserviços. Os objetivos específicos são: (i) descrever as FNs que guiam e motivam a migração para MS; (ii) mapear os RNFs com as FNs; (iii) analisar as similaridades e variabilidades entre esses requisitos e (iv) descrever os *trade-offs* existentes entre os RNFs.

Este artigo está estruturado da seguinte forma: a seção 2 traz os passos metodológicos, na seção 3 os resultados preliminares são apresentados e discutidos,

na seção 4, são descritas as contribuições deste estudo, seguida da seção 5, onde são apresentados os trabalhos relacionados. Finalmente, a seção 6 traz as considerações adicionais.

2 Metodologia de Pesquisa

Para atingir o objetivo deste trabalho, foram definidos quatro passos, os quais são descritos na sequência.

Passo 1: Identificação das FNs Através da análise de artigos, foram extraídas as forças norteadores que motivaram ou guiaram a migração de sistemas monolíticos para MSs. Para isso, foi utilizada uma metodologia baseada no mapeamento sistemático, conforme proposto em [13], tendo como objetivo a identificação de estudos sobre abordagens e experiências de migração de sistemas monolíticos para MSs. Inicialmente, foram feitas buscas em seis bases de dados científicas utilizando as seguintes palavras-chave: (i) "monolith", (ii) "migrate, transform, refactor, decompose, extract, partition" e (iii) "microservice". O retorno inicial foi de 422 estudos. Após a aplicação de critérios de inclusão e exclusão, restaram 61 artigos. Com o objetivo de validar a seleção, foi feita uma comparação com a seleção realizada em quatro estudos importantes sobre migração para microsserviços: [7], [5], [4] e [14]. Foi possível verificar que 42 artigos considerados nestes estudos secundários também foram listados no mapeamento realizado em nossa pesquisa, sendo que os demais estudos não listados eram contemporâneos ou mais recentes do que os quatro artigos considerados na comparação. Como produto da execução deste passo, tem-se uma lista das FNs citadas nos estudos selecionados.

Passo 2: Mapeamento das FNs para requisitos não-funcionais Para cada um dos artigos selecionados foram mapeados os RNFs relacionados. Esse mapeamento foi feito através da análise textual de cada FN obtida no passo anterior. A lista de forças norteadores foi complementada com os respectivos RNFs, os quais podem ser vistos como os requisitos que devem ser satisfeitos para que se alcance os objetivos estabelecidos (FNs).

Passo 3: Categorização dos RNFs identificados Os RNFs foram analisados e categorizados nos seguintes grupos: *escalabilidade, independência, manutenibilidade, implantação, confiabilidade, modularidade, gerenciabilidade, performance, reusabilidade, portabilidade, agilidade* e *segurança*. Tais grupos foram obtidos a partir do estudo proposto em [1]. Os RNFs obtidos no passo 2 foram classificados de forma que cada RNF poderia estar associado apenas a uma única categoria principal. Como resultado, tem-se uma tabela de cruzamento entre as categorias e os RNFs que fazem parte de cada categoria.

Passo 4: Elaboração do modelo de features Tendo como base a categorização obtida no passo anterior, será feita uma análise de similaridades, variabilidades

e interações para a construção de um modelo de *features*, utilizando-se o software *FeatureIDE*[1]. Este modelo foi escolhido, pois permite definir quais requisitos são ou não obrigatórios e quais requisitos impactam positivamente ou negativamente os demais. Permite também visualizar os requisitos de forma hierárquica, partindo do nível mais geral até o nível mais detalhado dos RNFs, auxiliando na compreensão, a partir da seleção de uma categoria específica, de quais requisitos devem ser satisfeitos e quais os demais requisitos serão impactados. A metodologia para construção do modelo foi a seguinte: *(i) inicialmente estabeleceu-se o domínio, identificado como "MigracaoMS", composto pelos artigos sobre migração avaliados e de onde foram extraídas as forças norteadoras e respectivos RNFs; (ii) as categorias de RNFs foram inseridas no modelo como filhas do domínio principal; (iii) os RNFs foram inseridos como filhos de suas respectivas categorias, baseando-se na classificação realizada anteriormente; (iv) finalmente, os RNFs foram analisados transversalmente com o objetivo de descobrir quais tinham impacto nas demais.*

Passo 5: Avaliação dos resultados Com o objetivo de avaliar se a análise realizada neste estudo pode ser útil no processo de tomada de decisão sobre a migração para MS, será aplicado um *survey*, conforme proposto em [15]. O público-alvo desta etapa serão engenheiros de requisitos e/ou arquitetos de software com experiência na migração para MSs.

3 Resultados Preliminares

Através do mapeamento sistemático realizado, foram identificadas as forças norteadoras que motivam e guiam as migrações de sistemas monolíticos para microsserviços, as quais foram mapeadas para requisitos não-funcionais. Foram encontrados 75 RNFs, os quais são listados na Tabela 1.

Em uma análise inicial de similaridades e variabilidades (Figura 1), baseada no estudo de [9], foi possível identificar algumas interações entre RNFs: um alto nível de *coesão* e um *baixo acoplamento* favorecem a *escalabilidade*, a *implantação independente* e a *redução de custos*, pois MSs menores podem ser escalados de forma individual, permitindo obter eficiência na utilização dos recursos computacionais. A implementação de *segurança* em MSs pode prejudicar a *performance*, pois a inclusão de uma camada de controle pode gerar gargalos na execução dos MSs. A *testabilidade* favorece a *performance*, *disponibilidade* e *segurança*: a arquitetura de MS tem como característica a facilidade de mudanças e cada alteração pode gerar impactos nestes três RNFs, os quais podem ser mitigados através da execução de testes.

Nestes casos de análise de interação entre os RNFs é necessário entender quais atributos ou variáveis mais operacionais de um RNF-a estão impactando nos de um RNF-b. Esse entendimento permitirá analisar se a interação na análise de variabilidade será de *inclusão*, ou seja, o RNF-a requer que também seja incluído

[1] Disponível em https://featureide.github.io

Tabela 1. Requisitos não-funcionais mapeados através das forças norteadoras

Adaptabilidade	Facilidade de implantação	Melhor customização de software
Agilidade	Facilidade de implementação	Menor complexidade de desenvolvimento
Alta capacidade	Facilidade de incorporar Devops	Modularidade
Alta coesão	Flexibilidade de configuração	Monitorabilidade
Alta concorrência	Flexibilidade de mudanças	Multi-tenancy
Atomicidade	Flexibilidade de tecnologia	Organização dos times
Auditabilidade	Gerenciabilidade	Performance
Autonomia	Gerenciamento automatizado	Portabilidade
Baixo acoplamento	Governança de dados descentralizada	Produtividade
Balanceamento de carga	Governança heterogênea	Rápida prototipagem
Clusterabilidade	Implantação automatizada	Rastreabilidade
Colaborativo	Implantação contínua	Reconfigurabilidade
Composabilidade	Implantação independente	Redução da complexidade dos times
Compreensibilidade	Implantação rápida	Redução de custos
Confiabilidade	Independência	Redução do tamanho dos times
Disponibilidade	Independência de gerência	Redução no time to market
Distributividade	Independência de times	Replicabilidade
Eficiência de processamento	Independência do ciclo de vida	Resiliência
Eficiência no uso de recursos	Independência dos serviços	Reusabilidade
Elasticidade	Inovação	Robustez
Escalabilidade	Inovação rápida	Segurança
Evolução independente	Integração contínua	Substituível
Evolucionalidade	Interoperabilidade	Sustentabilidade
Extensibilidade	Isolamento dos dados	Testabilidade
Facilidade de gerenciamento	Manutenibilidade	Tolerância a falhas

o RNF-b ou de *exclusão*, onde o RNF-b não pode ser selecionado caso RNF-a tenha sido considerado pelo engenheiro de requisitos.

4 Contribuições Esperadas

No contexto do processo de migração para microsserviços, este trabalho pretende guiar engenheiros de requisitos e arquitetos de software na escolha da configuração adequada de RNFs que deverão ser satisfeitos durante o processo de migração, auxiliando também na tomada de decisão sobre migrar ou não para microsserviços. Engenheiros e Arquitetos poderão analisar cenários possíveis (configurações identificadas na análise de variabilidade) de migração de acordo com os RNFs selecionados.

5 Trabalhos Relacionados

Em [17] são identificadas as principais motivações da migração para microsserviços, no entanto, elas não são relacionadas aos requisitos não-funcionais. Em nosso estudo, as motivações (FNs) descritas nos artigos analisados foram mapeadas para RNFs.

No trabalho de [9] é feita uma revisão sistemática da literatura com o objetivo de identificar e sintetizar os principais estudos sobre atributos de qualidade

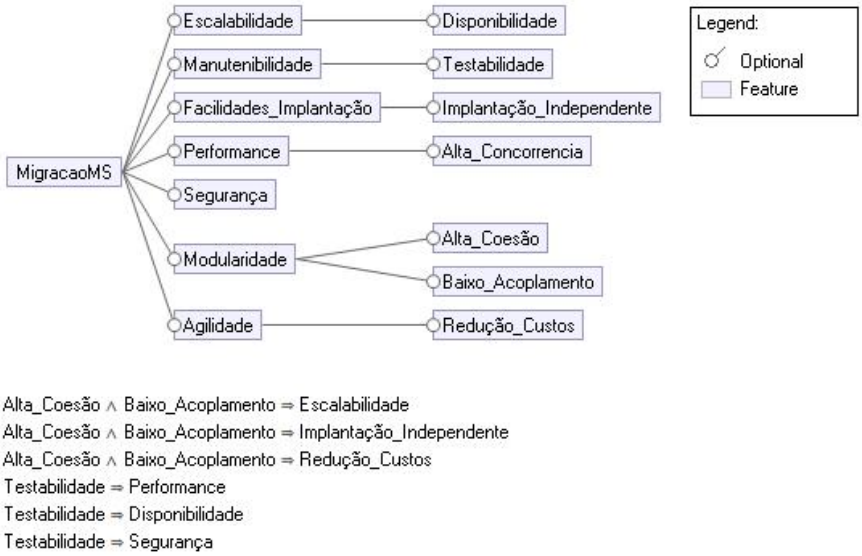

Figura 1. Modelo de *features* inicial

relacionados aos MSs. Como resultado, são apresentados os atributos de qualidade mais abordados e respectivas táticas que podem ser utilizadas para alcançar tais atributos. Nosso estudo pode complementar o trabalho de [9], permitindo que arquitetos de software tenham uma visão de quais RNFs são importantes antes de iniciar um processo de migração, baseado nos RNFs específicos de sua organização.

Em [10] é feita uma análise sistemática da literatura sobre três aspectos dos RNFs: definição e terminologia, tipos e relevância em diferentes domínios de aplicação. No entanto, não foi abordado especificamente o domínio de aplicação orientado a serviços, tão pouco microsserviços. Nosso estudo pode preencher a lacuna existente em [10], na categorização dos requisitos relevantes no domínio de microsserviços.

6 Considerações Adicionais

As atividades no curso de mestrado foram iniciadas em 2020 e deverão ser concluídas em março de 2022. Informações adicionais sobre o estudo podem ser consultadas em `https://github.com/marcioveronez/nfrmsmigration.git`.

Referências

1. Alshuqayran, N., Ali, N., Evans, R.: A systematic mapping study in microservice architecture. Proceedings - 2016 IEEE 9th International Conference on

Service-Oriented Computing and Applications, SOCA 2016 pp. 44–51 (2016). https://doi.org/10.1109/SOCA.2016.15
2. Bass, L., Weber, I., Liming, Z.: DevOps: A Software Architect's Perspective. Addison-Wesley (2015)
3. Bontemps, Y., Heymans, P., Schobbens, P.Y., Trigaux, J.C.: Semantics of FODA Feature Diagrams. Proceedings SPLC 2004 Workshop on Software Variability Management for Product Derivation–Towards Tool Support (Theorem 2), 48–58 (2004)
4. Di Francesco, P., Lago, P., Malavolta, I.: Architecting with microservices: A systematic mapping study (2019). https://doi.org/10.1016/j.jss.2019.01.001
5. Di Francesco, P., Malavolta, I., Lago, P.: Research on Architecting Microservices: Trends, Focus, and Potential for Industrial Adoption. Proceedings - 2017 IEEE International Conference on Software Architecture, ICSA 2017 pp. 21–30 (2017). https://doi.org/10.1109/ICSA.2017.24
6. Dragoni, N., Giallorenzo, S., Lafuente, A.L., Mazzara, M., Montesi, F., Mustafin, R., Safina, L.: Microservices: Yesterday, today, and tomorrow. Present and Ulterior Software Engineering pp. 195–216 (2017)
7. Fritzsch, J., Bogner, J., Zimmermann, A., Wagner, S.: From Monolith to Microservices: A Classification of Refactoring Approaches, vol. 11350. Springer International Publishing (2019). https://doi.org/10.1007/978-3-030-06019-0
8. Knoche, H., Hasselbring, W.: Using Microservices for Legacy Software Modernization. IEEE Software (2018)
9. Li, S., Zhang, H., Jia, Z., Zhong, C., Zhang, C., Shan, Z., Shen, J., Babar, M.A.: Understanding and addressing quality attributes of microservices architecture: A Systematic literature review. Information and Software Technology **131**, 106449 (2021). https://doi.org/10.1016/j.infsof.2020.106449, `https://doi.org/10.1016/j.infsof.2020.106449`
10. Mairiza, D., Zowghi, D., Nurmuliani, N.: An investigation into the notion of non-functional requirements. Proceedings of the ACM Symposium on Applied Computing (May), 311–317 (2010). https://doi.org/10.1145/1774088.1774153
11. Newman, S.: Building Microservices. 1st edn. (2015)
12. Newman, S.: Monolith to Microservices. O'Reilly Media (2020)
13. Petersen, K., Vakkalanka, S., Kuzniarz, L.: Guidelines for conducting systematic mapping studies in software engineering: An update. Information and Software Technology **64**, 1–18 (2015). https://doi.org/10.1016/j.infsof.2015.03.007, `http://dx.doi.org/10.1016/j.infsof.2015.03.007`
14. Ponce, F., Marquez, G., Astudillo, H.: Migrating from monolithic architecture to microservices: A Rapid Review. Proceedings - International Conference of the Chilean Computer Science Society, SCCC **2019-November** (2019). https://doi.org/10.1109/SCCC49216.2019.8966423
15. Shull, F., Singer, J., Sjøberg, D.I.: Guide to advanced empirical software engineering (2008). https://doi.org/10.1007/978-1-84800-044-5
16. Taibi, D., Lenarduzzi, V., Pahl, C., Janes, A.: Microservices in agile software development pp. 1–5 (2017). https://doi.org/10.1145/3120459.3120483
17. Taibi, D., Systä, K.: From monolithic systems to microservices: A decomposition framework based on process mining. CLOSER 2019 - Proceedings of the 9th International Conference on Cloud Computing and Services Science (Closer), 153–164 (2019). https://doi.org/10.5220/0007755901530164