Um Estudo Exploratório sobre Infraestrutura de Operação para uma Assistente Virtual Inteligente *

Aléxia C. S. da Silva[1], Thatiane de O. Rosa[2,3], Fernando F. Scattone[2],Alfredo Goldman[2]

[1]Escola de Artes Ciências e Humanidades - EACH/USP

[2]Instituto de Matemática e Estatística – IME/USP

[3]Instituto Federal do Tocantins – IFTO

{thatiane,ffs,gold}@ime.usp.br, ac.scheffer@usp.br

Abstract. *In order to execute a distributed complex system on the cloud, several tools and techniques were developed on the last few years, regarding scalability, orchestration, containers and microservices. This work aims to use these technologies to define and configure an operation infrastructure to the virtual assistant called A.D.A. (Advanced Distributed Assistant). We are researching several kinds of organization and communication for a microservice architecture, towards identifying better ways to integrate and provide services that make up the assistant. We hope that at the end of this study we have configured and tested a viable operating infrastructure to provide services and integrate the different parts that compose A.D.A..*

Resumo. *Várias ferramentas e técnicas foram desenvolvidas nos últimos anos para a execução de sistemas distribuídos na nuvem relacionadas a escalabilidade, orquestração, contêineres e microsserviços. Diante desse cenário, este trabalho tem o objetivo utilizar essas tecnologias para definir e configurar uma infraestrutura de operação da assistente virtual chamada A.D.A. (Assistente Distribuída Avançada). Para tanto, estão sendo estudadas diferentes maneiras de organização e comunicação da arquitetura de microsserviços, buscando identificar melhores formas de prover os serviços e integrar as partes que compõem a assistente em questão. Esperamos que ao final do estudo tenhamos configurado e testado uma infraestrutura de operação viável para prover serviços e integrar as diferentes partes que compõem a assistente A.D.A..*

1. Introdução

O progresso de ferramentas de arquitetura e aprendizado de máquina tem viabilizado o desenvolvimento de assistentes virtuais comandados por voz. Nesse cenário, tais assistentes têm se tornado um produto pessoal e rotineiro, que antes somente era idealizado ou problematizado. De acordo com [Mitchell et al.], uma assistente comandada por voz reconhece e interpreta comandos verbais e assim, realiza tarefas solicitadas pelo usuário, ocultando toda a parte computacional que viabiliza a sua execução. Contudo, para que uma assistente comandada por voz atinja o seu objetivo, é necessária uma infraestrutura

*Este projeto foi financiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e pelo grupo de extensão CodeLab uclab.xyz/site

que atenda adequadamente a escalabilidade de processos e a integração de diferentes dispositivos que compõem o ambiente do usuário.

Diante desse contexto, para lidar com alguns dos desafios relatados, apresentamos o projeto A.D.A. (Assistente Distribuída Avançada)[1], que é uma assistente pessoal distribuída inteligente que interage com o usuário por meio de comandos de voz. Tais comandos são executados em aparelhos conectados com a assistente, permitindo criar comandos personalizados. O objetivo do projeto é criar uma assistente de código aberto, cujo processamento seja feito de forma distribuída. Para que tal objetivo seja alcançado, serão estudados aspectos como processamento de voz em português, interação da assistente com dispositivos ambientados e maneiras de distribuir o processamento.

O projeto A.D.A. é subdividido em cinco projetos menores, que abordam desde conversões de voz para texto e comandos, interpretação e execução desses comandos em dispositivos IoT (*Internet of Things* – Internet das coisas), até o estudo de aspectos relacionados à infraestrtutura de operação. Este artigo está focado no projeto de infraestrutura de operação, que corresponde em compreender as diferentes maneiras de organização e comunicação da arquitetura de microsserviços, buscando identificar a melhor forma de prover os serviços e integrar as partes que compõem a assistente. Indo além, considera-se pesquisar diversas arquiteturas e modelagens de negócio para melhor funcionamento do sistema em conjunto a testes de integração.

2. Arquitetura de Software

O desenvolvimento de *software* está em constante evolução, demandando cada vez mais flexibilidade e organização. Diante disso, a área de arquitetura de *software* mostra-se relevante, pois é considerada um fator crítico de sucesso no processo de desenvolvimento [Shaw and Garlan 1996]. Tal criticidade deve-se ao fato de a área de arquitetura de *software* auxiliar na identificação de melhores formas de gerenciamento, implementação e manutenção de sistemas complexos [Newman 2015]. Logo, com o objetivo de auxiliar na definição da arquitetura mais adequada, são adotados estilos arquiteturais, tais como monolítico, orientado a serviços e baseado em microsserviços. Considerando o contexto do projeto A.D.A., o estilo escolhido foi o de microsserviços.

2.1. Microsserviços

Seguindo a definição proposta por [Newman 2015], o estilo arquitetural de microsserviços é definido como vários serviços pequenos, autônomos, que trabalham de forma conjunta e buscam representar o contexto tecnológico. Serviços pequenos e autônomos são otimizados em diversos aspectos, pois é possível atingir áreas específicas da arquitetura. Portanto, para alterações no código, como adicionar novas funcionalidades ou fazer manutenção, os riscos e custos do processo são minimizados. Junto a isso, por conta da autonomia, é possível trazer heterogeneidade ao código, deixando implementações mais simples e ágeis ao permitir diversas ferramentas.

Entretanto, para alcançar as características desejadas, é necessário qualidade na criação e gerenciamento dos microsserviços. E para isso, [Newman 2015] destaca dois conceitos-chave: baixo acoplamento e alta coesão. Baixo acoplamento é descrito como o

[1]Mais informações sobre o projeto estão disponíveis em: `uclab.xyz/ada`

baixo grau de dependência entre os serviços, que resulta em grande autonomia entre eles. Já alta coesão, é a atribuição para cada serviço a uma única responsabilidade: executá-la da melhor forma possível e não interferir nas responsabilidades senão as dele.

Por fim, pela natureza dos microsserviços, [Newman 2015] destaca como benefícios: heterogeneidade, resiliência, escalabilidade, fácil implantação, times flexíveis e organizados, otimização de testes e reescrita de código. Assim, levando em consideração que cada componente do sistema A.D.A. demanda APIs (*Application Programming Interface*) e escalas variadas, a arquitetura do projeto é baseada em microsserviços.

2.1.1. Integração de Microsserviços

De acordo com [Newman 2015], garantir uma boa integração dos microsserviços é fundamental para o funcionamento escalável do sistema. Portanto, é recomendável seguir um protocolo, como: evitar a integração de banco de dados e interrupção de alterações; manter APIs adotadas independentes da evolução tecnológica; implementar serviços simplificados e ocultar detalhes de implementação interna. Outros aspectos relevantes para a integração de microsserviços e pontuados por [Newman 2015] são:

- Tipo de comunicação que deve ser feita – síncrona ou assíncrona. Para comunicação síncrona, é solicitada uma operação a um servidor remoto que é bloqueado durante o processo de conclusão de tal operação. Enquanto na comunicação assíncrona, não se espera a conclusão da operação e essa estratégia se torna útil para execuções longas ou de baixa latência. Para determinar a melhor opção, é necessário considerar os requisitos que a arquitetura deve cumprir;
- Modelo de arquitetura que será adotado – orquestrado ou coreografado. Para o modelo de orquestração, a condução do processo é centralizada em um único serviço, administrador dos serviços restantes. Já para o modelo coreografado, cada componente do sistema é informado sobre seu trabalho, colaborando com o todo. Arquiteturas coreografadas são mais desacopladas e flexíveis.

Diante disso, ao avaliar as limitações e objetivos do projeto A.D.A., a melhor escolha de estrutura de microsserviços trará autonomia e potencial de escalabilidade.

3. Metodologia

Diante do apresentado, o método de pesquisa da infraestrutura da assistente A.D.A será através da pesquisa exploratória, visando familiaridade com o problema e levantamento bibliográfico, e estudo de caso, buscando conhecimento detalhado do projeto. Portanto, para realizar satisfatoriamente o estipulado, considerou-se as etapas:

1. Revisão bibliográfica sobre arquitetura de microsserviços;
2. Escolha de ferramentas e tecnologias de infraestrutura;
3. Configuração de *pipeline* de integração contínua;
4. Execução de experimentos sobre escalabilidade e coleta de resultados.

Esta pesquisa não está finalizada, e até o momento apenas a Etapa 1 foi concluída e a Etapa 2 está em andamento. Durante a Etapa 1, realizamos uma pesquisa bibliográfica com o objetivo de compreender definição, vantagens e desvantagens, modelos, armadilhas

e integração do estilo arquitetural baseado em microsserviços. Durante essa etapa foram consultados livros e artigos científicos que discutem sobre microsserviços.

Durante a Etapa 2, para pôr em prática os conhecimentos teóricos adquiridos e implementar os sistemas que compõem a A.D.A, foi escolhida a ferramenta Docker[2], que serve para isolar o ambiente de execução do microsserviço. Também, realizou-se um curso para entender melhor o funcionamento do Docker. Além disso, se fez necessário estudar um sistema operacional favorável para o gerenciamento da arquitetura e o desenvolvimento de microsserviços em Docker, sendo escolhido o Debian Linux/GNU[3] por ter uma comunidade ativa e bem ampla de usuários.

Durante a Etapa 3, trabalharemos junto com os outros responsáveis pelo desenvolvimento das demais partes da A.D.A. na comunicação e execução na nuvem dos componentes distintos do projeto. Pretendemos pesquisar e utilizar os serviços de integração contínua oferecidos por plataformas de repositórios de código aberto (como GitLab[4]) para facilitar o desenvolvimento, execução de testes automatizados e manutenção do código. Ao final do projeto, pretende-se executar um experimento com todas as partes da A.D.A. funcionando concomitantemente e a assistente funcionando de forma completa. Durante o experimento coletaremos medidas de latência na comunicação entre os componentes para comparar com o desempenho de outros assistentes virtuais.

4. Considerações Finais

A partir do estudo realizado até o momento, tem-se como hipótese que o estilo arquitetural de microsserviços, contribui para a heterogeneidade tecnológica, facilita a distribuição do processamento das atividades realizadas e aumenta a elasticidade da assistente virtual A.D.A.. Além disso, a partir da revisão bibliográfica foi possível justificas Docker como melhor opção para isolamento de execução [Felter et al. 2015] e Kubernetes para a orquestração de instâncias na nuvem [Truyen et al. 2019]. Esperamos que os resultados finais mostrem as vantagens do uso de microsserviços para sistemas complexos.

Referências

Felter, W., Ferreira, A., Rajamony, R., and Rubio, J. (2015). An updated performance comparison of virtual machines and linux containers. In *2015 IEEE International Symposium on Performance Analysis of Systems and Software*, pages 171–172.

Mitchell, T., Caruana, R., Freitag, D., Mcdermott, J., and Zabowski, D. Experience With a Learning Personal Assistant 1. Technical report.

Newman, S. (2015). *Building Microservices: Designing Fine-Grained Systems*. O'Reilly Media, Sebastopol - USA, first edition.

Shaw, M. and Garlan, D. (1996). *Software architecture: perspective on an emerging discipline*. Alan Apt.

Truyen, E., Landuyt, D. V., Preuveneers, D., Lagaisse, B., and Joosen, W. (2019). A comprehensive feature comparison study of open-source container orchestration frameworks. *Applied Sciences*, 9(5):931.

[2] `docker.com`
[3] `debian.org`
[4] `gitlab.com`