Universidade de Brasília – UnB
Faculdade UnB Gama – FGA
Engenharia de Software

Desenvolvimento de software em Startups: Estudo de caso na empresa Rua Dois

Autor: Iasmin Santos Mendes
Orientador: Dr. Renato Coral Sampaio

Brasília, DF
2019

Iasmin Santos Mendes

Desenvolvimento de software em Startups: Estudo de caso na empresa Rua Dois

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Universidade de Brasília – UnB

Faculdade UnB Gama – FGA

Orientador: Dr. Renato Coral Sampaio

Brasília, DF

2019

Iasmin Santos Mendes

Desenvolvimento de software em Startups: Estudo de caso na empresa Rua Dois/ Iasmin Santos Mendes. – Brasília, DF, 2019-

46 p. : il. (algumas color.) ; 30 cm.

Orientador: Dr. Renato Coral Sampaio

Trabalho de Conclusão de Curso – Universidade de Brasília – UnB
Faculdade UnB Gama – FGA , 2019.

1. *Startups*. 2. Desenvolvimento de Software. I. Dr. Renato Coral Sampaio. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Desenvolvimento de software em Startups: Estudo de caso na empresa Rua Dois

CDU 02:141:005.6

Iasmin Santos Mendes

Desenvolvimento de software em Startups: Estudo de caso na empresa Rua Dois

Monografia submetida ao curso de graduação em Engenharia de Software da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Software.

Trabalho aprovado. Brasília, DF, 11 de dezembro de 2019:

Dr. Renato Coral Sampaio
Orientador

Dr. Fernando William Cruz
Convidado 1

Dr. John Lenon Cardoso Gardenghi
Convidado 2

Brasília, DF
2019

Dedico este trabalho a minha mãe, por sonhar e me incentivar a chegar até aqui. E ao meu pai, por todo o apoio e suporte dado durante esta caminhada.

Acreditamos no sonho e
construímos a realidade.

Roberto Marinho

Resumo

Nos últimos anos houve um crescente surgimento de empresas denominadas *startups*. Essas empresas possuem como objetivo buscar, no menor tempo possível, um modelo de negócios repetitível e escalável que gere inovação no mercado. Nesse contexto nasceu a *startup* Rua Dois com o propósito de inovar o mercado imobiliário. Contudo, durante a sua busca por inovação e escalabilidade, o processo de desenvolvimento de software foi conturbado. Levando a algumas decisões técnicas que dificultaram a evolução do software na mesma velocidade que era desejada pela a empresa. Consequentemente, foi necessário simplificar toda a arquitetura do software desenvolvido com o intuito de adaptar-se melhor ao atual contexto da *startup*. O presente trabaho realizará uma análise comparativa entre as duas arquiteturas adotadas na empresa com o intuito de avaliá-las e definir possíveis diretrizes que orientem a Rua Dois em relação as suas expectativas de desenvolvimento de software escalável.

Palavras-chave: Desenvolvimento de software. Escalabilidade. Estudo de caso. *Startup*.

Abstract

In the last years there was a growing emergence of companies called startups. These companies aim to seek, as soon as possible, a repeatable and scalable business model that creates market innovation. In this context was born the startup Rua Dois with the purpose of innovating the real estate market. However, during their quest for innovation and scalability the software development process was disturbed. Leading to some technical decisions that made it difficult for software to evolve at the same speed as the company wanted. Consequently, it was necessary to simplify the entire architecture of the software developed in order to better adapt to the current startup context. The present work will perform a comparative analysis between the two architectures adopted in the company in order to evaluate them and define possible guidelines that guide the Rua Dois regarding their expectations of scalable software development.

Key-words: Case study. Scalability. Software development. Startup.

Lista de ilustrações

Lista de quadros

Lista de abreviaturas e siglas

API Application Programming Interface.

AWS Amazon Web Services.

CRM Customer Relationship Management.

MVC Model-View-Controller.

TCC 2 Trabalho de Conclusão de Curso 2.

TCC 1 Trabalho de Conclusão de Curso 1.

TI Tecnologia da Informação.

Sumário

1 Introdução

Existe no mundo um novo cenário emergente no qual, empresas nascentes, denominadas *startups*, produzem inovação ao ponto de serem capazes de competir com empresas já estabelecidas no mercado (DULLIUS; SCHAEFFER, 2016). O ambiente de negócios está mudando devido a fatores como globalização, política e tecnologia. Neste novo contexto, tudo que se conhece dos negócios são hipóteses e um constante alto grau de risco (NAGAMATSU; BARBOSA; REBECCHI, 2013).

Nesse cenário, surgiu a *startup* Rua Dois Tecnologias. Com uma série de hipóteses sobre o mercado imobiliário, a empresa visa inovar esse ramo que é ainda muito burocrático e carente de melhorias (XU, 2019). E nessa euforia por inovação, a *startup* tem em seu modelo de negócios a constante busca pela validação rápida de ideias mas de uma forma que ela esteja preparada pra crescer e se expandir rapidamente, conforme os princípios de repetibilidade e escalabilidade[1] de Blank (2010) para *startups*.

Nessa busca por escalabilidade a empresa tomou uma série de decisões tecnológicas baseadas no que seria escalável, mas sem uma análise coerente com o sistema proposto e com o contexto de validação de ideias que a empresa vivenciava. A exemplo está a escolha de utilizar uma arquitetura de microsserviços sem conhecer de fato a demanda dos serviços que seriam prestados, ocasionando dificuldades para a empresa de modificar e evoluir esse sistema com a rapidez que é desejada para uma *startup*.

Assim surgem as indagações do presente trabalho. Neste contexto das *startups* o software produzido é incessantemente alterado em busca da melhor solução para um determinado problema, é possível preparar esse software de forma que ele possa ser escalável? Se for possível, quais práticas e recomendações podem ser aplicadas para alcançar este objetivo?

Com foco nessas questões, este trabalho almeja realizar um estudo de caso sobre o desenvolvimento de software na *startup* Rua Dois, visando avaliar as questões apresentadas e direcionar esta empresa para um caminho mais consolidade em relação aos seus objetivos de crescimento do sistema.

[1] Os princípios de Steve Blank sobre *startups* serão abordados com maior profundidade na **??**.

1.1 Justificativa

As metodologias ágeis[2] visam diminuir os custos com retrabalho mediante as constantes mudanças de requisitos no processo de desenvolvimento de software, não apenas acomodando essas mudanças, mas abraçando-as. Assim, o design do software é construído de forma contínua (HIGHSMITH; COCKBURN, 2001). Mas como garantir qualidade na construção desse design sendo que esses requisitos na verdade são hipóteses a serem comprovadas sobre o software? E em paralelo a validação dessas hipóteses, preparar esse software para atender a grande demanda que é almejada para as *startups*? Ries (2011) em seu livro *A Startup Enxuta*, traz a seguinte reflexão sobre esta realidade:

> Como sociedade, dispomos de um conjunto comprovado de técnicas para administrar grandes empresas, e conhecemos as melhores práticas para construir produtos físicos. No entanto, quando se trata de *startups* e inovação, ainda estamos atirando no escuro.

Portanto, este trabalho justifica-se por sua contribuição em buscar, neste cenário citado por Ries como obscuro, compreender as necessidades do desenvolvimento de software e como essas necessidades podem ser alinhadas ao ciclo de vida de uma *startup*.

1.2 Objetivos

1.2.1 Objetivo Geral

Este trabalho tem como objetivo geral realizar uma pesquisa exploratória acerca de arquitetura de software desenvolvendo uma análise comparativa baseada em revisão bibliográfica pretende-se construir uma hipótese sobre como o desenvolvimento de software escalável deve ser conduzido dentro de cada fase do ciclo de vida de uma *startup*.

1.2.2 Objetivos Específicos

As seguintes metas foram levantadas visando atingir o objetivo geral do presente trabalho:

1. Compreender o contexto e o ciclo de vida de uma *startup*;
2. Compreender as necessidades de uma *startup* em relação ao desenvolvimento de software;

[2] Métodos iterativos para desenvolvimento de software, que visam a reação a mudanças conforme sejam as necessidades do cliente. Vide <http://agilemanifesto.org/>

3. Compreender e elencar técnicas e boas práticas voltadas o desenvolvimento de software escalável;

4. Definir um método capaz de correlacionar as técnicas de escalabilidade com as fases do ciclo de vida de uma *startup*.

5. Aplicar o método definido e identificar os pontos de intercessão entre as técnicas de escalabilidade e as fases de uma *startup*, afim de construir a hipótese citada na subseção 1.2.1.

2 Fundamentação Teórica

Neste capítulo serão apresentadas as bases teóricas que permeiam o escopo do presente trabalho. Partindo do entendimento sobre o conceito de arquitetura de software e como ela reflete nos sistemas. Em seguida abordando estilos de arquitetura de software pertinentes para o contexto estudado. As seções estão dispostas em:

1. **Arquitetura de software:** definição do termo arquitetura dentro da área de software e suas características.
2. **Sistemas monolíticos:** descrição, vantagens e desvantagens da arquitetura monolítica.
3. **Microsserviços:** descrição, vantagens e desvantagens da arquitetura de microsserviços.

2.1 Arquitetura de Software

> Software systems are constructed to satisfy organizations' business goals. The architecture is a bridge between those (often abstract) business goals and the final (concrete) resulting system. While the path from abstract goals to concrete systems can be complex, the good news is that software architecture can be designed, analyzed, documented, and implemented using know techniques that will support the achievement of these business and mission goals. The complexity can be tamed, made tractable. (BASS; CLEMENTS; KAZMAN, 2015, tradução nossa)

Como retratado na citação acima, arquitetura de software representa a ponte entre os sistemas construídos e os objetivos de cada organização. A publicação *What is your definition of software architecture?* do *Software Engineering Institute* da *Carnegie Mellon University* (SEI, 2017) apresenta uma variedade de diferentes conceitos a respeito de arquitetura de software abordados pela bibliografia, exemplificando a dificuldade existente em compreender o conceito de arquitetura dentro da área de Tecnologia da Informação (TI).

Segundo Vogel et al. (2011) arquitetura é um aspecto implícito com o qual os desenvolvedores são confrontados diariamente e que não pode ser ignorado ou eliminado, ainda que eles nem sempre tenham consciência sobre a sua constante presença.

Bass, Clements e Kazman (2015) traz a seguinte definição a respeito de arquitetura de software:

> The software architecture of a system is the set of structures needed to reason about the system, which comprise software elements, relations among them, and properties of both. (BASS; CLEMENTS; KAZMAN, 2015, tradução nossa)

Com base nessa abordagem, os autores defendem que os sistemas de software são compostos por diversas estruturas e elementos, por como essas estruturas se relacionam entre si e pelas propriedades que os caracterizam. Assim, a arquitetura se torna uma abstração do sistema, destacando aspectos desejados do mesmo e omitindo outros aspectos, com o intuito de minimizar a complexidade com a qual o compreendemos.

A Figura 1 apresenta uma segunda visão abordada por Richards e Ford (2020) a respeito de arquitetura. Nessa abordagem eles defendem quatro dimensões as quais definem o que é arquitetura de software. A primeira dimensão consiste em estruturas: microsserviços, camadas e módulos são exemplos de conceitos abarcados por esse ponto, todavia, para eles descrever a arquitetura por meio de estruturas não é suficiente para contemplar todo o escopo acerca de arquitetura de software. Diante desta problemática, eles introduzem os conceitos de características, decisão e princípios de design referentes a arquitetura.

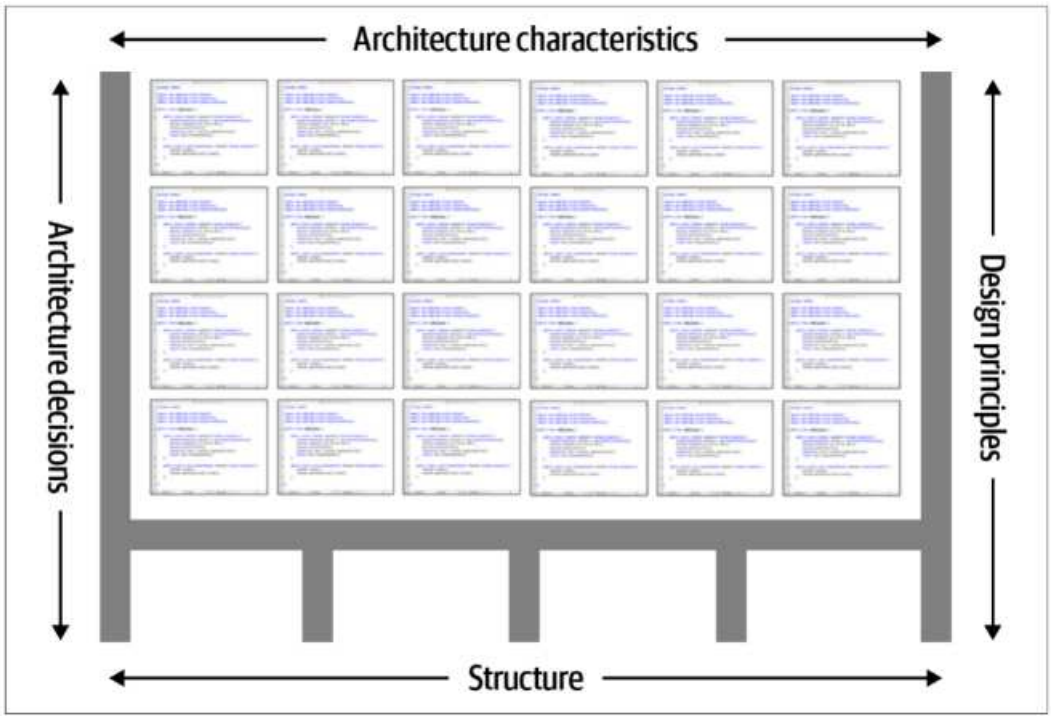

Figura 1 – Elementos que compõem a arquitetura de um software (RICHARDS; FORD, 2020)

As características da arquitetura, por sua vez, referem-se aos critérios de sucesso do sistema, trazendo pontos como disponibilidade, confiabilidade, segurança, etc., os quais são anteriores e independentes ao conhecimento acerca das funcionalidades do sistema.

O terceiro aspecto apresentado pelos autores consiste nas decisões de arquitetura responsáveis por definir as regras sob as quais o sistema deve ser construído. Isto diz ao time de desenvolvimento o que ele pode ou não fazer. A exemplo, em uma arquitetura Model-View-Controller (MVC)[1], a *View* é impossibilitada de consumir dados da *Model*, e cabe aos desenvolvedores garantir que isso seja respeitado.

Por fim, a última dimensão refere-se aos princípios de design. Estes consistem em um guia pelo o qual os desenvolvedores podem se orientar quando for necessário tomar alguma decisão. Diferentemente das decisões de arquitetura que devem ser sempre seguidas afim de garantir as características desejadas para a arquitetura, os princípios trazem maior flexibilidade e autonomia para o desenvolvedor, podendo este analisar o contexto do problema em mãos e decidir se seguir os princípios é a melhor abordagem para a funcionalidade desenvolvida, ou se deve seguir por um caminho diferente o qual ele julgue que os ganhos são maiores para a aplicação.

2.1.1 Leis da Arquitetura de Software

> **Tudo em arquitetura de software é um *trade-off*.** Primeira Lei da Arquitetura de Software
>
> ***Por que* é mais importantes do que *como*.** Segunda Lei da Arquitetura de Software
>
> [2] (RICHARDS; FORD, 2020, tradução nossa)

A Primeira Lei da Arquitetura de Software traz a tona a realidade do arquiteto de software, o qual precisa lidar constantemente com conflitos de escolha. O papel do arquiteto está intrinsecamente ligado a análise da situação e a tomada de decisão sobre qual caminho seguir. Para tal, é necessário que o mesmo esteja alinhado com uma série de fatores, como práticas de engenharia, *DevOps*[3], processos, negócio, etc (RICHARDS; FORD, 2020).

A segunda lei traz uma perspectiva que por diversas vezes é ignorada, na qual tendemos a olhar para a topologia dos sistemas, observando suas estruturas e como estas se relacionam, e não damos a devida atenção ao porquê das decisões arquiteturais tomadas(RICHARDS; FORD, 2020).

[1] *Model-View-Controller* (MVC) consiste em um estilo arquitetural adotado por várias aplicações. A *Model* representa a camada de dados, a *View* a camada de apresentação e a *Controller* é a camada de lógica.(MCGOVERN et al., 2004)

[2] Texto original: *Everything in software architecture is a trade-off. First Law of Software Architecture. Why is more important than how. Second Law of Software Architecture.*

[3] *DevOps* é um movimento cultural que altera como indivíduos pensam sobre o seu trabalho, dando suporte a processos que aceleram a entrega de valor pelas empresas ao desenvolverem práticas sustentáveis de trabalho.(DAVIS; DANIELS, 2016)

Nesse sentindo, o ponto defendido pelos autores busca olhar o por quê certas decisões são tomadas mediante os *trade-offs* enfrentados.

2.2 Sistemas monolíticos

A Arquitetura Monolítica é um padrão de desenvolvimento de software no qual um aplicativo é criado com uma única base de código, um único sistema de compilação, um único binário executável e vários módulos para recursos técnicos ou de negócios. Seus componentes trabalham compartilhando mesmo espaço de memória e recursos formando uma unidade de código coesa (SAKOVICH, 2017).

Esse padrão arquitetural permite um desenvolvimento fácil por ser de conhecimento da maioria dos desenvolvedores de software e apresentar baixa complexidade para a execução de tarefas como *deploy*, confecção de testes e compartilhamento do código. Contudo, a longo prazo começam a surgir dificuldades para manter essa arquitetura, entre elas estão:

1. Dificuldade em entender e alterar o código que ao longo do tempo se torna muito extenso e coeso;
2. Limitação na agilidade de atualização do software, uma vez que para cada pequena alteração é necessário reimplantar o código por completo;
3. Necessita de muito esforço para adotar uma nova tecnologia, sendo necessário adaptar todo o código da aplicação para a nova ferramenta;
4. O sistema perde a confiabilidade pois a medida que cresce um *bug* em uma única parte do código pode interromper todo o software já que todos os componentes estão conectados.

2.3 Microsserviços

Segundo Newman (2015), microsserviços são pequenos serviços autônomos que trabalham juntos. O objetivo desses serviços é tornar o código coeso e focado em resolver as regras de negócio dentro de um limite específico. As vantagens de adotar essa arquitetura consistem em:

1. Uso de tecnologias heterogêneas;
2. Maior resiliência do sistema;
3. Facilidade em escalar pequenas partes do sistema, ao invés do sistema por completo;

4. Facilidade e maior estabilidade no processo de *deploy*;
5. Maior produtividade da equipe, uma vez que se passa a adotar equipes menores com bases de código menores;
6. Reusabilidade dos serviços;
7. Otimização da substituibilidade: organizações de médio e grande porte costumam possuir sistemas legados, com uma base de código enorme funcionando com tecnologias antigas tanto de software quanto de hardware. A substituição desses códigos é um processo custoso e arriscado. Com serviços pequenos realizar essa substituição se torna um processo mais fácil de gerenciar.

Para o desenvolvimento de microsserviços é necessário pensar sobre o que deve ser exposto e o que deve ser ocultado. Se houver muito compartilhamento os serviços de consumo se acoplam às nossas representações internas, diminuindo a nossa autonomia e consequentemente a nossa capacidade de realizar alterações (NEWMAN, 2015).

3 Metodologia

3.1 Levantamento bibliográfico

Para referenciamento deste trabalho foram realizados estudos acerca de *startups*, desenvolvimento de software e aspectos relacionados a escalabilidade de um sistema por meio de livros, Google Scholar, dentre outras fontes. Os tópicos abordados dentro de cada tema foram selecionados com base na sua respectiva importância dentro do assunto e no seu envolvimento com o contexto aprensentado. Individualmente, os temas citados são bem explorados e abordados pela comunidade acadêmica, contudo, quando relacionados os artigos e estudos a respeito tendem a ser mais escassos.

Publicações e palestras de empresas de referencia voltadas ao desenvolvimento de software no contexto de *startups* também foram utilizadas com o intuito de contribuir com perspectivas mais atuais a respeito do tema.

3.2 Metodologia de pesquisa

A proposta deste trabalho é correlacionar técnicas de desenvolvimento de software escalável com as fases de vida de uma *startup*. Para tal, foi aplicado o METODO X, visando estabelecer essa correlação. Esta seção visa descrever os objetos de análise, o método aplicado e as etapas envolvidas no desenvolvimento de toda a pesquisa.

3.2.1 Objetos de análise

Os objetos de análise do presente estudo consistem em:

Ciclo de vida de uma *startup* Fases pelas quais uma *startup* passa desde o seu nascimento até a sua estabilização no mercado.

Técnicas de escalabilidade Técnicas e boas práticas aplicadas no mercado em busca da construção de sistemas escaláveis e/ou com a capacidade para tal.

3.2.2 Método de correlação

3.2.3 Etapas do desenvolvimento

A etapas adotadas na construção da seguinte pesquisa foram:

Etapa 1 Definição do método de análise a ser aplicado;

Etapa 2 Coleta de dados a respeito dos objetos de análise;

Etapa 3 Aplicação do método de análise sobre os dados coletados;

Etapa 4 Análise dos resultados obtidos;

Etapa 5 Classificação das técnicas de escalabilidade;

Etapa 6 Validação ??.

A seguir segue a descrição de cada etapa.

3.2.3.1 Etapa 1: Definição do método de análise a ser aplicado

Nesta etapa, foi definido o método de análise utilizado, o qual tem como objetivo de estabelecer a correlação entre entre as técnicas de escalabilidade elencadas e as fases do cicla de vida de uma *startup*.

3.2.3.2 Etapa 2: Coleta de dados a respeito dos objetos de análise

Mediante o método de análise escolhido, foram coletados por meio de pesquisa bibliográfica as informações necessárias acerca dos objetos de análise definidos na subseção 3.2.1. Dessa forma obteve-se as características sobre os objetos de análise pertinentes a aplicação do método.

3.2.3.3 Etapa 3: Aplicação do método de análise sobre os dados coletados

Com os dados em mãos, nessa etapa aplicou-se o méotodo de análise proposto visando obter a correlação entre os objetos analisados.

3.2.3.4 Etapa 4: Análise dos resultados obtidos

A partir dos resultados obtidos na etapa anterior, realizou-se uma análise afim de compreender os mesmos e analisar a coerência com o que era esperado.

3.2.3.5 Etapa 5: Classificação das técnicas de escalabilidade

Por fim, classificou-se as técnicas de escalabilidade elencadas em relação as fases da *startup* usando como base os resultados obtidos nas etapas anteriores. Obtendo assim, as recomendações almejadas no objetivo geral deste trabalho, o qual está disposto na subseção 1.2.1.

3.3 Planejamento das atividades

Esta seção destina-se a descrever o planejamento realizado visando atingir as etapas definidas na subseção 3.2.3 e com base na metodologia de pesquisa descrita na seção 3.2.

Mediante o escopo da pesquisa a ser realizada, planejou-se a execução para um total de 7 semanas, iniciando na metade do mês de Março/2020 e indo até a primeira semana de Maio/2020. No Quadro 1 estão dispostas as atividades que pretende-se cumprir ao longo deste período.

Quadro 1 – Cronograma das atividades

Semana	**Atividades**
1	Definição do método de análise a ser aplicado
2	Coleta de dados a respeito dos objetos de análise
3	Aplicação do método de análise sobre os dados coletados
4	Análise dos resultados obtidos e classificação das técnicas de escalabilidade
6	Revisão e escrita do documento
7	Apresentação

3.4 Ferramentas

As ferramentas que pretende-se usar para a execução deste trabalho são:

Draw.io editor gráfico online que permite a construção de processos e desenhos relevantes ao conteúdo apresentado;

Google Planilhas ferramenta para construção de tabelas e gráficos. A qual pretende-se usar para realizar análises sobre os dados coletados.

Trello quadro interativo que auxilia na organização de projetos. O mesmo foi utilizado para gerenciar as etapas e tarefas alocadas para cada semana de execução do projeto, permitindo acompanhar o andamento do mesmo.

4 Desenvolvimento

Esta seção tem por propósito apresentar o estudo realizado sobre a *startup* Rua Dois Tecnologias, sendo esta o objeto de análise central do presente trabalho. O escopo deste capítulo abrangerá desde o entendimento sobre o contexto da empresa e sua proposta de valor até aspectos mais técnicos como a arquitetura do sistema, sua evolução durante o primeiro ano da empresa e as dificuldades enfrentadas pela *startup* nessa arquitetura.

As seções estão dispostas em:

1. **Rua Dois Tecnologias:** breve apresentação sobre a empresa, seus propósitos e os serviços oferecidos;

2. **Arquitetura do sistema:** descrição da arquitetura de software adotada dentro da Rua Dois, no seu primeiro ano de desenvolvimento, e possíveis soluções discutidas dentro da empresa a respeito da escalabilidade.

4.1 Rua Dois Tecnologias

No artigo *Modernizing Real Estate: The Property Tech Opportunity* [1], a autora Xu (2019) aborda uma visão histórica sobre o desenvolvimento de tecnologias para o mercado imobiliário fazendo um comparativo com a realidade atual. Nesse histórico de desenvolvimento, ela relata que desde 1980 são empregados diversos esforços na área buscando melhorarias por meio de software, mas mesmo assim, continuamos em uma ramo altamente burocrático e carente de melhorias.

Mediante essas limitações, surgiu em outubro de 2018 a Rua Dois Tecnologias, com o propósito de solucionar as dificuldades enfrentadas tanto por parte das imobiliárias quanto por parte dos locatários [2] no processo de locação de imóveis.

A proposta[3] da empresa consiste em atuar como um facilitador para as imobiliárias ingressarem no meio digital e consequentemente propiciar seu crescimento. Para tal, eles buscam aumentar a eficiência do processo de locação por meio da digitalização, proporcionando melhor integração entre operação e tecnologias. Assim, com um processo mais digital, os recursos humanos podem ser melhores empregados no atendimento do cliente favorencendo a experiência do mesmo nesse mercado tão burocrático.

[1] Disponível nest link.
[2] Aquele que mora em um imóvel que não lhe pertence, mediante um contrato de locação.
[3] Mais informações a respeito da proposta e visão da Rua Dois podem ser encontradas neste *podcast*.

4.1.1 Serviços prestados

Visando o processo de locação de imóveis, a Rua Dois conta com quatro serviços sendo desenvolvidos dentro da empresa, afim de agregar valor aos seus clientes, sendo eles:

1. **Captação de Imóveis:** serviço destinado à atrair proprietários de imóveis que desejam divulgar o imóvel para locação;
2. **Visitas:** serviço de visitas aos imóveis, no qual os interessados em locar o imóvel agendam um horário para conhecer o mesmo;
3. **Propostas:** serviço no qual os interessados, após passarem pela etapa de visitação, avaliam o imóvel e fazem uma proposta para o proprietário do imóvel referente ao valor a ser pago pela locação e possíveis ajustes que desejam que sejam feitos na propriedade;
4. **Contrato:** etapa final, destinada a automatizar a avaliação por parte das seguradoras, envio de documentos e assinatura do contrato.

Cada serviço é vendido de forma separada para as imobiliárias, de forma que cada uma adapte ao seu contexto os serviços que melhor se encaixam. Atualmente, o serviço mais desenvolvido é o de visitas e a equipe vem trabalhando com o intuito de estabilizar os demais serviços.

4.2 Arquitetura do sistema

O início do desenvolvimento do software da Rua Dois se deu em outubro de 2018 com base na proposta de criar um sistema usando de uma arquitetura de microsserviços, de forma que esse sistema já fosse preparado pra escalar desde seu início. Contudo, a empresa enfrentou uma série de dificuldades[4] em relação a esta arquitetura e ao seu contexto no respectivo momento. Mediante essas dificuldades a Rua Dois optou em seu processo de desenvolvimento migrar essa arquitetura de microsserviços para um sistema monolítico.

Esta seção visa descrever essas duas fases arquiteturais dentro da empresa com suas respectivas motivações e desafios encontrados. Assim, visando facilitar o entendimento, cada fase será nomeada da seguinte forma:

Fase 1 referente a fase inicial de desenvolvimento na qual foi adotada uma arquitetura de microsserviços;

Fase 2 referente a segunda fase de desenvolvimento na qual foi adotada uma arquitetura monolítica.

[4] Estas dificuldades serão abordadas com maior clareza na subseção 4.2.1

4.2.1 Arquitetura do sistema: Fase 1

A primeira fase de desenvolvimento da *startup* foi marcada pela premissa de que o sistema deveria ser escalável, e com base nessa premissa foram tomadas uma série de decisões visando preparar o sistema para tal. Assim o sistema foi dividido em dois microsserviços principais e em uma série de microsserviços auxiliares distribuídos nos lambdas da Amazon Web Services (AWS). Os dois serviços principais são:

r2service responsável por gerir o cadastro de imóveis e as propostas de locação do imóvel;

r2visit responsável por gerir a lógica de agendamento de visitas a um imóvel.

A Figura 2 visa ilustrar a organização desse sistema. Nela os dois serviços principais estão representados pelos círculos maiores roxos e as demais representações são outros serviços que auxiliam no ciclo de vida do produto. Adiante segue descrito o fluxo com o qual todo o sistema é acionado, seguindo a enumeração disposta na imagem:

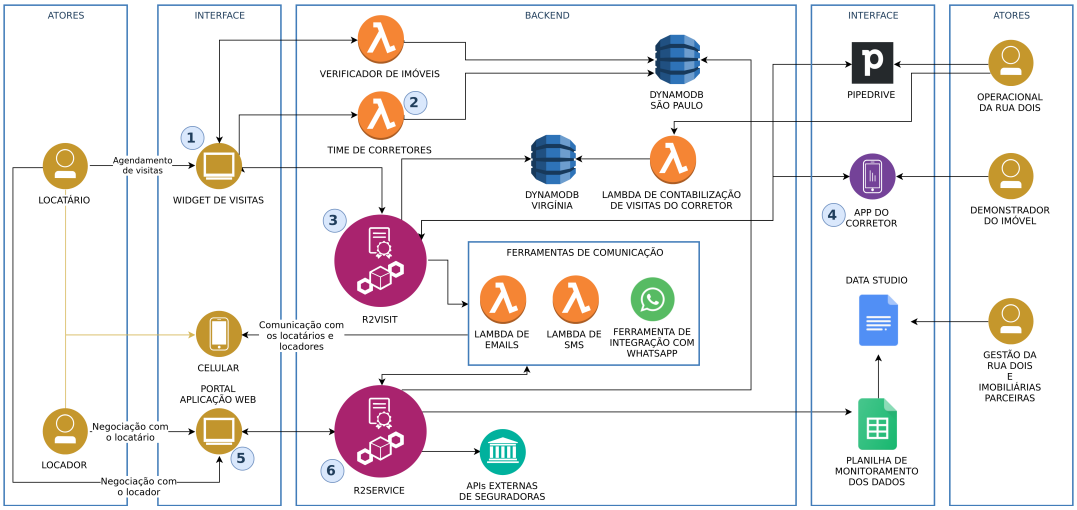

Figura 2 – Arquitetura do sistema da Rua Dois na Fase 1

1. O *Widget* é um componente utilizado pela Rua Dois que roda no *frontend* dos sites das imobiliárias que aderem ao serviço de visitas. Ele se comunica com um lambda responsável por verificar se o imóvel apresentado no site da imobiliária está disponível no banco de dados da Rua Dois. Uma vez que esteja disponível, o *widget* exibe um botão que disponibiliza o agendamento da visita *online* para o interessado;

2. O locatário ao clicar nesse botão, dispara uma requisição ao lambda que identifica o time de corretores responsável por atender as visitas ao imóvel selecionado. E em seguida, uma nova requisição é disparada ao serviço *r2visit* solicitando o calendário com os horários disponíveis do respectivo time;

3. Após o locatário ter selecionado o horário da visita, uma nova requisição é lançada para registrar a visita no *r2visit*, que por sua vez dispara uma série de mensagens por meio das ferramentas de comunicação e atualiza o Pipedrive[5];

4. Após a visita ser realizada, o corretor sinaliza por meio do aplicativo a conclusão da visita, disparando novamente no *r2visit* os eventos de atualização do Pipedrive e as ferramentas de comunicação, para que se possa dar início a nova fase de Negociação, referente ao produto de propostas;

5. Nessa nova fase, locatário e locador trocam mensagens por meio do portal da Rua Dois, o qual é gerido pelo *r2service*. Ao final da negociação, os clientes entram na fase de contrato, na qual seu dados são recolhidos e enviados por meio do *r2service* para Application Programming Interfaces (APIs) externas de seguradoras as quais realizam uma avaliação de crédito do locatário. Quando concedida a avaliação de crédito, o contrato é gerado automaticamente com os dados já recolhidos do locatário, locador, da imobiliária e do próprio imóvel;

6. Por fim, o *r2service* é responsável por atualizar uma planilha do Google Sheets[6] que serve de insumo para geração de relatórios dentro do Data Studio [7].

Nesse fluxo nota-se que existe uma grande diversidade de elementos interagindo entre si e a forma como isto foi organizado gerou as seguintes dificuldades dentro da empresa:

1. Dificuldade de organização dos dados da empresa uma vez que não houve um bom planejamento em relação a interação entre os serviços *r2service* e *r2visit*, o que ocasionou a duplicação da tabela referente aos dados dos imóveis - uma tabela no banco de dados da Virgínia e a outra tabela no banco de dados de São Paulo;

2. Dificuldade de gerir as associações necessárias entre os dados armazenados no banco. A empresa optou pelo uso do banco não relacional DynamoDB, visando novamente a escalabilidade, mas sem planejar corretamente o armazenamento desses dados nessa estrutura não relacional. Isso impactou diretamente o desenvolvimento, visto que existe uma necessidade da empresa de tratar esse dado como relacional mas fez isso em cima de uma tecnologia inadequada para tal;

3. Dificuldade de manter essa infraestrutura, tanto pela complexidade na forma como o sistema está organizado quanto pela escolha de utilizar a infraestrutura da AWS, que

[5] O Pipedrive é uma ferramenta de Customer Relationship Management (CRM) utilizada pela empresa para realizar o gerenciamento operacional e monitoramento dos resultados obtidos. Saiba mais em: <https://www.pipedrive.com>

[6] Saiba mais em: <https://www.google.com/sheets/about/>

[7] Saiba mais em: <https://datastudio.google.com>

é mais árdua de trabalhar do que outras soluções disponíveis no mercado (KAVYA, 2019);

4. Dificuldade de inserção de novos membros na equipe de desenvolvimento, visto que entender como esses vários serviços se relacionam não é uma tarefa simples;

5. Dificuldade de mudança e inserção de novas funcionalidades. Nessa primeira fase a Rua Dois encontrava-se fortemente no período de validação da ideia dentro de uma *startup* o que exigia uma rápida reação da equipe em relação aos *feedbacks* recebidos. Contudo, nessa arquitetura se tornou exaustivo a realização de constantes mudanças vistos todos os problemas relatados nos tópicos anteriores.

4.2.2 Arquitetura do sistema: Fase 2

Mediante as dificuldades enfrentadas na arquitetura adotada na Fase 1 e a incerteza sobre a escalabilidade desse sistema, em junho de 2019 a Rua Dois optou por migrar para uma arquitetura monolítica com a perspectiva de tornar o sistema mais estável e mais apto a realização de mudanças. Nessa mudança os serviços *r2service* e *r2visit* se tornaram um único sistema, agora chamado de *novo r2service*, alguns lambdas foram incorporados por esse novo sistema ou simplesmente foram descartados juntamente com alguns serviços auxiliares que não seriam mais usados. A Figura 3 exemplifica como está o sistema nessa nova configuração.

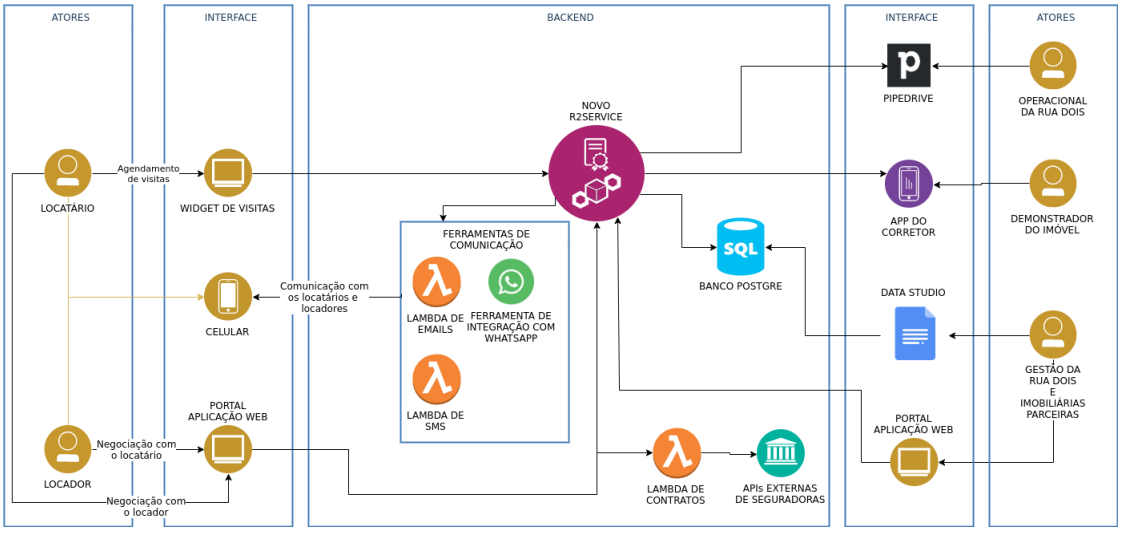

Figura 3 – Arquitetura do sistema da Rua Dois na Fase 2

5 Considerações Finais

Neste capítulo serão apresentadas considerações referentes ao que foi realizado neste trabalho até o presente momento e as expectativas e metas para a próxima fase.

As seções estão dispostas em:

1. **Primeira entrega parcial** considerações acerca dos resultados entregues na primeira etapa do respectivo trabalho, referente ao Trabalho de Conclusão de Curso 1 (TCC 1);
2. **Entrega final** considerações acerca das expectativas almejadas para a realização da segunda etapa do respectivo trabalho, referente ao Trabalho de Conclusão de Curso 2 (TCC 2);
3. **Cronograma** Distribuição planejada das fases de execução deste trabalho durante o período subsequente.

5.1 Primeira entrega parcial

Nesta primeira fase foi realizado o levantamento bibliográfico a respeito dos assuntos relacionados ao tema central com o intuito de explorar as visões dispostas na literatura sobre o mesmo. Foram encontrados diversos relatos a respeito de *Startups*, desenvolvimento de software escalável desde o início, métodos de desenvolvimento de software, etc. No entanto, a respeito de desenvolvimento de software em *startups* e desenvolvimento de software escalável em *startups* o assunto ainda é muito escasso. Nesse sentido, foi proposto uma metodologia de pesquisa afim de explorar essas duas questões dentro do contexto da Rua Dois. Também foi realizado uma análise inicial a respeito das arquiteturas de software adotadas na empresa com o intuito de compreender melhor as problemáticas enfrentadas pela *startup* e o estado atual.

5.2 Entrega final

Espera-se realizar a coleta dos dados propostos e com base nos resultados elaborar a análise comparativa entre as duas fases da Rua Dois. Assim, a partir da análise comparativa realizada propõem-se levantar diretrizes que orientem o desenvolvimento de software escalável na empresa mediante as experiências já vivenciadas pela mesma. A validação dessas diretrizes será constituída com base no que é encontrado na literatura a respeito do tema.

5.3 Cronograma

Esta seção visa apresentar o cronograma para desenvolvimento do presente trabalho. Este cronograma está dividido em duas etapas: Primeira Entrega Parcial (TCC 1) e Entrega Final (TCC 2).

O cronograma referente a Primeira Entrega Parcial é apresentado no Quadro 2. Em relação a atividade de *Pesquisa e definição da metodologia*, foi definido o **??** apresentado na **??** que será aplicado para elencar quais métricas e informações serão coletadas durante a Etapa 2: Coleta de dados a respeito dos objetos de análise. Contudo, a execução desse processo irá definir aspectos como métricas e tecnologias que ainda não estão descritos na Metodologia. Por este motivo, esta atividade foi considerada como parcialmente concluída.

Quadro 2 – Cronograma: primeira entrega parcial

Atividade	**Agosto**	**Setembro**	**Outubro**	**Novembro**	**Status**
Levantamento e definição do tema	X	X			Concluído
Definição do objetivo geral e dos específicos		X			Concluído
Pesquisa e levantamento bibliográfico		X	X		Em andamento
Pesquisa e definição da metodologia			X		Parcialmente concluído
Análise inicial da arquitetura				X	Concluído

O Quadro 3 apresenta o cronograma previsto para a execução da segunda etapa do trabalho, conforme à metodologia e ao planejamento definido no Capítulo 3.

Quadro 3 – Cronograma: entrega final

Atividade	**Fevereiro**	**Março**	**Abril**	**Maio**
Estudo acerca da arquitetura adotada	X			
Coleta e levantamento de dados referentes a qualidade do código	X	X		
Coleta e levantamento de dados referentes as práticas de desenvolvimento de software	X	X		
Coleta e levantamento de dados referentes ao conhecimento da equipe		X	X	
Coleta e levantamento de dados referentes as tecnologias utilizadas		X	X	
Coleta e levantamento de dados referentes a escalabilidade			X	X
Coleta e levantamento de dados referentes ao ciclo de vida da *startup*			X	X
Análise comparativa				X
Definição e validação das diretrizes				X
Escrita do relatório final	X	X	X	X

Referências

BASS, L.; CLEMENTS, P.; KAZMAN, R. *Software Architecture in Pratic*. 3. ed. Massachusetts, USA: Pearson Education, Inc., 2015. ISBN 978-0-321-81573-6. Citado 2 vezes nas páginas 25 e 26.

BLANK, S. *What's a Startup? First Principles.* Steve Blank, 2010. Disponível em: <https://steveblank.com/2010/01/25/whats-a-startup-first-principles>. Acesso em: 29.06.2019. Citado na página 21.

DAVIS, J.; DANIELS, R. *Effective DevOps: Building a Culture of Collaboration, Affinity, and Tooling at Scale*. O'Reilly Media, 2016. ISBN 9781491926437. Disponível em: <https://books.google.com.br/books?id=6e5FDAAAQBAJ>. Citado na página 27.

DULLIUS, A. C.; SCHAEFFER, P. R. As capacidades de inovação em startups: contribuições para uma trajetória de crescimento. *Revista Alcance*, Santa Catarina, v. 23, n. 1, 2016. Citado na página 21.

HIGHSMITH, J.; COCKBURN, A. Agile software development: The business of innovation. IEEE, v. 34, n. 9, 2001. Citado na página 22.

KAVYA. *DigitalOcean vs AWS EC2: 8 Factors to Decide Who's the Winner?* ServerGuy, 2019. Disponível em: <https://serverguy.com/comparison/digitalocean-vs-aws-ec2/>. Acesso em: 07.07.2019. Citado na página 39.

MCGOVERN, J. et al. *A Practical Guide to Enterprise Architecture*. Prentice Hall/Professional Technical Reference, 2004. (The Coad series). ISBN 9780131412750. Disponível em: <https://books.google.com.br/books?id=YGCNnnzeov0C>. Citado na página 27.

NAGAMATSU, F. A.; BARBOSA, J.; REBECCHI, A. Business model generation e as contribuições na abertura de startups. In: . São Paulo: II Simpósio Internacional de Gestão de Projetos, Inovação e Sustentabilidade, 2013. Citado na página 21.

NEWMAN, S. *Building Microservices: Designing Fine-Grained Systems*. USA: O'Reilly Media, Inc., 2015. Citado 2 vezes nas páginas 28 e 29.

RICHARDS, M.; FORD, N. *Fundamentals of Software Architecture: An Engineering Approach*. USA: O'Reilly Media, 2020. ISBN 9781492043423. Citado 3 vezes nas páginas 13, 26 e 27.

RIES, E. *The Lean Startup*. Nova Iorque: Currency, 2011. Citado na página 22.

SAKOVICH, N. *Monolithic vs. Microservices: Real Business Examples*. 2017. Disponível em: <https://www.sam-solutions.com/blog/microservices-vs-monolithic-real-business-examples/>. Acesso em: 07.11.2019. Citado na página 28.

SEI, S. E. I. What is your definition of software architecture? Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, n. 3854, Jan 2017. Citado na página 25.

VOGEL, O. et al. *Software Architecture: A Comprehensive Framework and Guide for Practitioners*. [S.l.]: Springer Berlin Heidelberg, 2011. ISBN 9783642197369. Citado na página 25.

XU, L. *Modernizing Real Estate: The Property Tech Opportunity*. Forbes, 2019. Disponível em: <https://www.forbes.com/sites/valleyvoices/2019/02/22/the-proptech-opportunity>. Acesso em: 06.07.2019. Citado 2 vezes nas páginas 21 e 35.