JOSÉ RENATO ZACARONI BARBOSA

DESENVOLVIMENTO DE UM SISTEMA DE COMPRA DE PEÇAS UTILIZANDO METODOLOGIA ÁGIL

LAVRAS – MG

2019

JOSÉ RENATO ZACARONI BARBOSA

DESENVOLVIMENTO DE UM SISTEMA DE COMPRA DE PEÇAS UTILIZANDO METODOLOGIA ÁGIL

Trabalho de conclusão de curso, na modalidade Monografia, apresentado à Universidade Federal de Lavras, como parte das exigências do Curso de Engenharia de Controle e Automação, para a obtenção do título de Bacharel.

Prof. Dr. Danilo Alves de Lima

Orientador

LAVRAS – MG

2019

Ficha catalográfica elaborada pela Coordenadoria de Processos Técnicos

da Biblioteca Universitária da UFLA

José Renato Zacaroni Barbosa

Desenvolvimento de um Sistema de Compra de Peças Utilizando Metodologia Ágil / . 1^a ed. rev., atual. e ampl. – Lavras : UFLA, 2019.

46 p. : il.

TCC(graduação)–Universidade Federal de Lavras, 2019.

Orientador: Prof. Dr. Danilo Alves de Lima.

Bibliografia.

1. Desenvolvimento de Software. 2. Automação de processos 3. Metodologia ágil I. de Lima, Danilo Alves. II. Título.

CDD-808.066

JOSÉ RENATO ZACARONI BARBOSA

DESENVOLVIMENTO DE UM SISTEMA DE COMPRA DE PEÇAS UTILIZANDO METODOLOGIA ÁGIL

Trabalho de conclusão de curso, na modalidade Monografia, apresentado à Universidade Federal de Lavras, como parte das exigências do Curso de Engenharia de Controle e Automação, para a obtenção do título de Bacharel.

APROVADA em 14 de junho de 2019.

Prof. Dr. Danilo Alves de Lima UFLA

Prof. Dr. Danilo Alves de Lima
Orientador

LAVRAS – MG
2019

Aos meus pais, Renato e Solange, minha irmã, Ana Cláudia, e as meus avós, especialmente a falecida Maria Aparecida, pela base indispensável que me trouxe até aqui.

AGRADECIMENTOS

Agradeço ao professor Danilo pelo apoio e dedicação na orientação da escrita deste trabalho e a todos os outros professores que tive na vida, pois todos eles contribuíram de forma essencial para minha formação. Demonstro também minha gratidão a família, Diguinho, amigos e companheiros de curso, Gabriel, Rafael, Argenis, Homero, Godinho, Célio e Pierangeli pelo apoio nesta caminhada. Aos colegas de trabalho e também amigos, Marcelo, Bambirra, Andrezão, Mayer, Wellington, Francis e Enock que tornaram prazerosas as jornadas de serviço. Agradeço também a CAPES e CNPq pelos apoios financeiros durante os estudos acadêmicos. E por fim, a minha namorada Yasmim pelo amor e incentivos constantes demonstrados em singelas atitudes.

Um livro é a prova de que os homens são capazes de fazer magia.
Carl Sagan

RESUMO

Com a grande competitividade do mercado é natural que as empresas busquem melhorias nos seus processos internos. Um processo fundamental no funcionamento de um negócio é a gestão de fornecedores. O gerenciamento eficaz dos fornecedores é uma fonte de vantagem competitiva potencialmente sustentável para as organizações, e a integração entre fornecedor e comprador também desempenha um papel crucial para isto. Uma das maneiras de melhorar este processo é automatizando rotinas operacionais e repetitivas com softwares a fim de otimizar o tempo dos colaboradores de uma empresa. O objetivo deste trabalho é construir uma aplicação Web como solução de um problema no setor de compra de peças em uma empresa de aluguel de carros. Para atingir o objetivo foi utilizado a metodologia Scrum para o desenvolvimento e ferramentas como Visual Studio, ASP .NET MVC Framework, as linguagens C#, razor, CSS e JavaScript e banco de dados Sybase e SQL Server. Os resultados obtidos foram a entrega da primeria versão do software no prazo de 1 mês, aumento da produtividade da área de compra de peças devido a automação de parte do processo e consequente redução de custos operacionais.

Palavras-chave: Desenvolvimento de Software. Automação de processos. Metodologia ágil.

ABSTRACT

With the great competitiveness of the market, it is natural for companies to seek improvements in their internal processes. A key process in running a business is supplier management. Effective supplier management is a source of potentially sustainable competitive advantage for organizations, and supplier-buyer integration also plays a crucial role. One of the ways to improve this process is to automate operational and repetitive routines with software in order to optimize the time of a company's employees. The overall goal of this work is to build a Web application as a solution to a problem in the parts buying industry at a car rental company. To achieve the goal it was used the Scrum methodology for development and tools such as Visual Studio, ASP .NET MVC Framework, C# languages, razor, CSS and JavaScript and Sybase database and SQL Server. The results obtained were the delivery of the first version of the software within 1 month, an increase of productivity in the area of purchase of parts due to the automation of part of the process and consequently reduction of operation costs.

Keywords: Software development. Process automation. Agile methodology.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

HTML Sigla de *HyperText Markup Language*, expressão inglesa que significa "Linguagem de Marcação de Hipertexto". Consiste em uma linguagem de marcação utilizada para produção de páginas na web. 42

PO Sigla de *Product Owner*. 23, 27

XP Abreviação de *Extreme Programming*. 21, 22, 26

APD Sigla de Análise e Processamento de Despesas. 16, 18, 29

APO Abreviatura de Apoio ao Fornecedor. 16, 18

RSA Acrônimo dos sobrenomes de Ron Rivest, Adi Shamir e Leonard Adleman, criadores do algoritmo RSA. 34

TI Sigla de Tecnologia da Informação. 11

URL Sigla de *Uniform Resource Locator*, e em português é conhecido por Localizador Padrão de Recursos. 30, 34, 35

SUMÁRIO

1 INTRODUÇÃO

Na década de 1990 a evolução das tecnologias de informação, telecomunicação e transportes trouxe uma nova realidade para as organizações. O mercado, que antes era formado por "ilhas" de negócios praticamente isoladas, transformou-se em um único e complexo ambiente de negócio global (PENA, 2010). Com a globalização da economia tornou-se possível uma abertura de mercado, assim, como uma nova realidade a ser enfrentada pelas organizações nos mercados do mundo inteiro. Por esta razão, criou-se um ambiente caracterizado por uma acirrada concorrência empresarial, obrigando as empresas a serem cada vez mais competitivas (PORTER, 1999).

A Tecnologia da Informação (TI), que inclui a tecnologia digital mais a internet compondo a economia digital, tem desempenhado um papel importante na transformação da sociedade e na forma como as organizações têm feito seus negócios e estruturado seus processos (CASTELLS, 2009). Nos dias de hoje, a TI é o que a eletricidade foi na Era Industrial. Sendo assim, a internet pode ser equiparada a uma rede elétrica quanto ao motor elétrico, em razão de sua capacidade de distribuir a força da informação por todo o domínio da atividade humana. A internet passou a ser a base tecnológica para a forma organizacional da Era da Informação: a rede (CASTELLS, 2009). Deste modo, para qualquer empresa acompanhar o cenário atual do mercado, é necessário o investimento em tecnologia da informação.

Com a grande competitividade do mercado é natural que as empresas busquem melhorias nos seus processos internos. Um processo fundamental no funcionamento de um negócio é a gestão de fornecedores. O gerenciamento eficaz desses é uma fonte de vantagem competitiva potencialmente sustentável para as organizações, e a integração entre fornecedor e comprador também desempenha um papel crucial para isto (VAART; DONK, 2008). Por isso, realizar cotações ágeis e de maneira organizada fazem a diferença para qualquer negócio, de qualquer ramo.

Como uma maneira de reduzir custos e, consequentemente, seus preços de venda, certas empresas utilizam técnicas como *Downsizing* (redução de pessoal). Porém, há outras que preferem investir em programas relacionados à qualidade, investimentos tecnológicos e outras técnicas propostas para se atingir o objetivo desejado (MARINO, 2006).

Apesar do processo de cotação ser demorado e burocrático, é possível utilizar tecnologias como o desenvolvimento de software para automatizar partes do processo e otimizar o fluxo de trabalho. De acordo com Intelipost (2017), investir em tecnologia permite que muitas

rotinas operacionais sejam concluídas de forma automática e diminui a quantidade de tarefas repetitivas e burocráticas sob a responsabilidade dos colaboradores. Assim, consegue-se alcançar economia de tempo, fazendo com que as equipes concluam um volume maior de demandas em menos tempo.

Entretanto, o desenvolvimento do software não é um processo fácil. Nos anos de 1990, a maioria dos projetos utilizavam metodologia baseada no modelo conhecido como "*Waterfall*". Neste modelo, as mudanças de requisitos são muito difíceis de serem aceitas em fases posteriores ao planejamento, pois exige uma regressão para a fase de concepção do plano (TOMÁS, 2009). Em contraposição a este modelo, nos anos 2000, surgiu a metodologia ágil que enfatiza um fluxo de trabalho iterativo e a entrega incremental de produtos de software em iterações curtas (PATANAKUL; HENRY; LEACH, 2015).

O estudo de caso deste trabalho envolve uma empresa de aluguel de carros que atualmente é líder de mercado na América Latina. Avaliada em 12,8 bilhões de reais, ela opera uma plataforma de negócios absolutamente sinérgicos: aluguel de carros para pessoas físicas, franchising de aluguel de carros, gestão de frotas que alugam veículos para pessoas jurídicas e uma rede de pontos de vendas de carros desativados da frota para consumidores finais.

No final de 2017, a empresa apresentava uma frota com aproximadamente 180 mil veículos e, atualmente no ano de 2019, este número ultrapassou os 200 mil. Com uma frota crescente, o setor interno que cuida da manutenção de veículos tem uma demanda cada vez maior de melhoria no gerenciamento dos fornecedores de peças.

Até o início deste projeto, a empresa contava apenas com um sistema de aprovação de orçamento de manutenção onde os fornecedores enviavam seus orçamentos para a manutenção de um veículo e um colaborador aprovava a melhor opção. Muitas vezes a manutenção exigia compra de peças e, inicialmente, ficava a cargo do próprio fornecedor comprar estas peças de um segundo e repassar à empresa com um adicional acarretando maior custo.

Para reduzir o custo, a empresa adaptou o sistema para gerar orçamentos de compra de peças, porém, o preenchimento do orçamento era de domínio de um colaborador da área. Para registrar no sistema as cotações, ele precisava entrar em contato, via telefone ou e-mail, com cada fornecedor para obter as informações. E somente depois, quando atingisse um certo número de cotações, tomar a decisão de compra levando em consideração o preço e o tempo de entrega.

Considerando esses elementos, neste trabalho será apresentado uma solução de automação para gestão de compras desta empresa. Nele será abordado o desenvolvimento de um sistema capaz de permitir que o próprio fornecedor preencha seus valores de cotação. Uma vez que o colaborador não precise mais ligar para o fornecedor, seu tempo de trabalho será otimizado e haverá melhora na produtividade, não precisando aumentar o número de funcionários para se comprar mais peças. Além disso, para desenvolvimento de tal sistema será utilizado a metodologia ágil afim de se ter uma entrega do produto incremental, isto é, o produto será entregue em pacotes menores agilizando sua entrada em produção.

1.1 Objetivos

Este trabalho tem como objetivo construir uma aplicação Web como solução de automação do setor de compra de peças numa empresa de aluguel de carros. Para tanto, os seguintes elementos foram considerados:

- Estudar o problema de automação e as restrições do processo;
- Aplicar a metodologia Scrum para desenvolvimento do software para que possa ter entregas incrementais e o produto entrar em produção de maneira mais rápida;
- Mostrar a influência que a nova tecnologia teve na produtividade do setor de compra de peças.

1.2 Contribuições

Foi desenvolvido um software utilizando metodologia ágil para reestruturar o processo de compra de peça e automatizar parte deste. Tal sistema automatizou o envio de e-mail para fornecedores preencherem cotação, transferido o tarefa de preencher as informações de cotação de peças do colaborador do setor para os próprios fornecedores interessados em vendê-las. Com isso, os estudos realizados ao longo deste projeto podem auxiliar como motivação na área de desenvolvimento de software e automação de processos, porque evidencia as entregas rápidas proporcionadas pela metodologia ágil e o rápido retorno de investimento com a diminuição de custos de produção devida à automação.

1.3 Estrutura do trabalho

O trabalho está dividido em quatro capítulos, como segue: O primeiro capítulo apresentou a introdução deste trabalho, com a motivação, problemas e objetivos esperados. No Referencial Teórico (Capítulo 2), têm-se o intuito de contextualizar o leitor nos conceitos utilizados nos capítulos posteriores. No Capítulo 3, são apresentadas as ferramentas e metodologias utilizadas durante o trabalho. Em seguida, os resultados são exibidos e discutidos no Capítulo 4 para que os dados obtidos sejam explicados e entendidos por completo. Finalmente, o Capítulo 5 traz as considerações finais e trabalhos futuros a respeito dos objetivos pretendidos e alcançados.

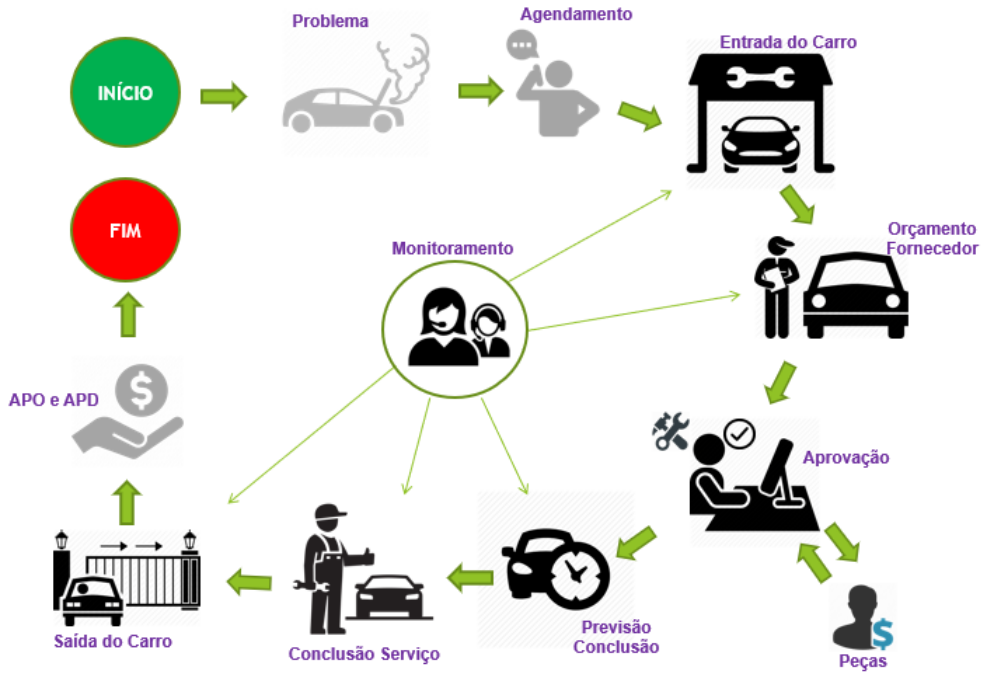
2 REFERENCIAL TEÓRICO

Este capítulo é dedicado à criação de uma base de conhecimento, fazendo com que a compreensão dos capítulos subsequentes seja mais fluída. Inicialmente, o leitor será contextualizado do problema proposto a ser resolvido por este trabalho e posteriomente será feita a discussão a respeito de processos de desenvolvimento de software até chegar no processo que foi utilizado.

2.1 Contextualização do problema

O processo de manutenção de veículos na empresa em questão acontece conforme o fluxograma da Figura 2.1. Inicialmente, tem-se a necessidade de manutenção de um carro, sendo ele levantado por um colaborador da área operacional ou por um cliente com o carro alugado. A manutenção do carro é agendada no sistema de atendimento 24 horas da própria empresa. Na situação em que o colaborador relata, ele mesmo faz o agendamento, no segundo caso, um atendente do 24 horas realiza o agendamento a partir de uma ligação de cliente.

Figura 2.1 – Processo de manutenção de carro na empresa.



Fonte: Do autor

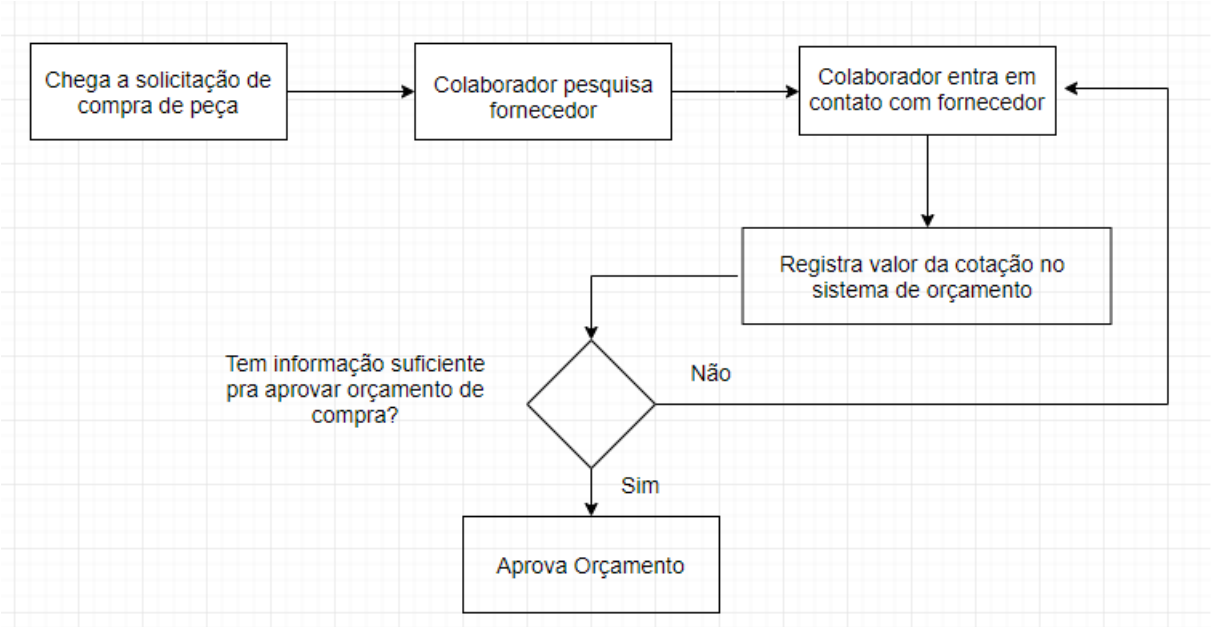
Uma vez que o carro deu entrada no fornecedor, inicia-se o procedimento de análise de orçamento. Tal análise é feita por um técnico, que aprova ou não o orçamento de manutenção, levando em considerações os preços de serviços e mão de obra. Em várias ocasiões, tal

manutenção exige a compra de peças e o técnico se vê comumente em duas situações: ou o fornecedor tem a peça e é muito cara, ou o fornecedor não tem a peça. Em ambas as situações, gera-se a necessidade de compra de peças.

Após a conclusão do serviço, há a saída do carro e a solicitação de serviço é enviada para as áreas de Apoio ao Fornecedor (APO) e Análise e Processamento de Despesas (APD), onde, neste último citado, os fornecedores serão pagos pelos serviços prestados. Todo procedimento da entrada até a saída do carro é monitorado pelo sistema de monitoramento. Este consiste em colaboradores que ligam diariamente perguntando a situação do serviço e cobrando o prazo para conclusão deste e, consequentemente, a saída do veículo.

Inicialmente, o sistema utilizado para aprovação de orçamento foi adaptado para gerir as compras de peças da seguinte maneira: Quando a manutenção exigia compra de peças, era marcado no orçamento da manutenção a necessidade de compra. Com isso, era criado um novo orçamento para aprovação que era a solicitação de compra de peça. A partir daí, o processo seguia conforme ilustra na Figura 2.2, onde o colaborador pesquisava um fornecedor na região onde o carro estava, muitas vezes pelo próprio *Google*, entrava em contato por *e-mail* ou telefone e registrava o valor da cotação no sistema. Se tinha um número adequado de cotações para tomar a decisão de compra, o orçamento era aprovado e a compra era feita.

Figura 2.2 – Processo de compra de peça a partir do sistema de aprovação de orçamento.



Fonte: Do autor

A partir do processo mapeado pela Figura 2.2, foram levantados, em conjunto com os colaboradores da área, os principais gargalos do sistema, sendo eles:

- Processo moroso: O fato de o colaborador ter que pesquisar contato de fornecedores e pedir informações da peça e o próprio colaborador registrar no sistema tomava muito tempo.

- Não ser possível registrar tempo de entrega: O prazo para entrega é uma informação importante, uma vez que quanto mais tempo o carro ficar parado, mais custo isto gera para a empresa (Estima-se que 1 dia do carro parado custa-se 100 reais). Até então, o campo de "previsão de conclusão" no sistema de orçamento era utilizado para este controle, mas muitas vezes não era preenchido e quando era, tornava difícil a análise devido a necessidade de comparar datas mentalmente.

- Falta de uma interface intuitiva: O sistema de aprovação de orçamento era dividido em muitas abas e era lento, isto é, não era possível ver todos os valores cotados para uma peça em uma única tela, o que tornava o processo ainda mais moroso.

Para contornar tal situação, foi considerada a possibilidade de ser comprado algum software pronto no mercado. Dentre as soluções, foram avaliados os softwares da *Ticketlog* (TICKETLOG, 2019) e *Ultracar* (ULTRACAR, 2019).

Ticketlog oferece um sistema para autogestão ou terceirização da operação com técnicos especializados, que garante agilidade nas cotações e preço negociado das peças. O sistema conta com clientes já cadastrados que cobrem 93% do território nacional, fornece acompanhamento do ciclo de manutenção com detalhamento técnico das peças e mão de obra. Além disso, a empresa fornece automação financeiro-fiscal com pagadoria centralizada e apoio com integração dos dados (TICKETLOG, 2019). Apesar da possibilidade de compra de peça e integração com o sistema da empresa de aluguel de carros, o software oferecido é mais específico para o ciclo todo de manutenção, algo que a empresa onde este trabalho foi realizado já executava. Como a necessidade era a compra de peça, tal possibilidade foi descartada.

Ultracar também apresenta um sistema que realiza toda a gestão financeira e acompanhamento em tempo real das movimentações de compras. Além disso, oferece relatórios que auxiliam nas tomadas de decisões do negócio (ULTRACAR, 2019). Ademais, conta também com fornecedores cadastrados, não necessitando do cadastro inicial manual por parte dos colaboradores. A área de compra de peça fez o teste durante um mês com este software e levantou as seguintes desvantagens: a usabilidade era difícil, o valor da peça era caro, pois era cobrado um percentual a mais para *Ultracar*, as cotações demoravam a serem respondidas e não tinha

integração com os sitemas internos da empresa de aluguel de carros. Com isto, era necessário cadastrar todas as compras realizadas manualmente nos sitemas internos. Além disso, tem a limitação das cotações serem apenas de fornecedores que o software oferece.

De forma geral, a solução de comprar um software pronto teria a vantagem de ter uma entrega imediata. Entretanto, não seria possível integrar as compras com os outros sistemas da empresa como o cadastro de fornecedores, monitoramento, APO e APD citados anteriormente. O monitoramento é responsável por supervisionar os serviços e chegada de peças compradas e os últimos pelo pagamento dos fornecedores. Isto é, a maioria das vantagens oferecidas por estes produtos, a empresa já dispunha com seus outros sistemas internos, e além destes sistemas citados anteriormente, havia também um sistema de cadastro de fornecedores e peças e uma base de dados pronta com essas informações para serem utilizadas. Portanto, foi proposto a construção de um software customizado que possibilitasse a integração com os outros sistemas da empresa e a utilização de um processo de desenvolvimento que agilizaria a entrega amenizando a desvantagem de não ter uma entrega imediata. Além disso, tal software seria construído especificamente para atender as necessidade da área, gerando mais valor para o segmento. O processo utilizado para construção de tal software, bem como sua implementação, será descrito nas seções e capítulos posteriores.

2.2 Modelos de Desenvolvimento de Software

Nesta seção serão abordados os principais modelos de desenvolvimento de software. Primeiramente será abordado o modelo em cascata que será comparado com as metodologias ágeis, descritas posteriormente.

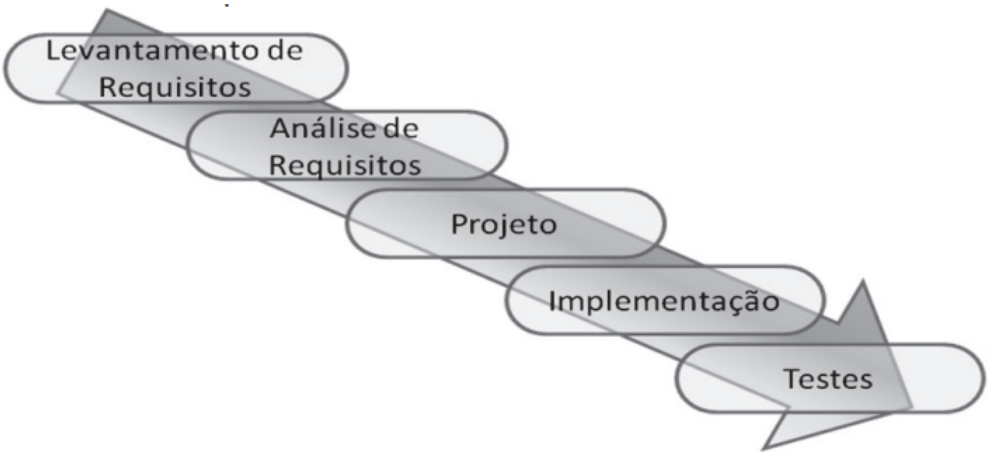
2.2.1 Modelo em Cascata

O modelo em cascata, ou *waterfall*, de desenvolvimento de software é derivado de processos mais gerais da engenharia de sistema, e tem esse nome por causa do encadeamento entre uma fase e outra. O modelo em cascata é um exemplo de um processo dirigido a planos, em princípio, deve-se planejar e programar todas as atividades do processo antes de começar a trabalhar nelas (SOMMERVILLE, 2011).

Os principais estágios do modelo são levantamento de requisitos, análise dos requisitos, projeto, implementação e testes conforme ilustra a Figura 2.3.

Sommerville (2011) descreve cada estágio da seguinte maneira:

Figura 2.3 – Esquemático do modelo Cascata.



Fonte: (CORTÉS, 2013)

1. Levantamento dos requisitos: Nesta etapa são definidos os requisitos de serviços, restrições e metas do sistema, estabelecidos por meio de consulta aos usuários.

2. Análise dos Requisitos: Onde os requisitos são definidos em detalhes e funcionam como uma especificação do sistema.

3. Projeto: O processo de projeto de sistemas aloca os requisitos tanto para sistemas de hardware como para sistemas de software, por meio da definição de uma arquitetura geral do sistema. O projeto de software envolve identificação e descrição das abstrações fundamentais do sistema de software e seus relacionamentos.

4. Implementação: Durante esse estágio, o projeto do software é desenvolvido como um conjunto de programas ou unidades de programa. O teste unitário envolve a verificação de que cada unidade atenda a sua especificação.

5. Teste: As unidades individuais do programa ou programas são integradas e testadas como um sistema completo para assegurar que os requisitos do software tenham sido atendidos. Após o teste, o sistema de software é entregue ao cliente.

Em suma, este modelo trata o desenvolvimento de software como um modelo linear, isto é, somente se passa para próxima fase depois da aprovação da anterior. Com isso, as mudanças de requisitos são muito difíceis de serem aceitas em fases posteriores ao planejamento, pois exige uma regressão para a fase de concepção do plano (TOMÁS, 2009). Além disso, a versão executável do software fica apenas disponível em uma etapa avançada do desenvolvimento.

Como consequência, o risco do sistema não atender completamente às necessidades do cliente fica latente até o final do desenvolvimento (CORTÉS, 2013).

Nos dias de hoje, as empresas operam em um ambiente global, com mudanças rápidas. Assim, precisam responder a novas oportunidades e novos mercados, a mudanças nas condições econômicas e ao surgimento de produtos e serviços concorrentes. Por esse motivo, muitas empresas estão dispostas a trocar a qualidade e o compromisso com requisitos do software por uma implantação mais rápida do software de que necessitam (SOMMERVILLE, 2011). Para atender a estas necessidades, surgiu a metodologia ágil, a qual será descrita na subseção subsequente.

2.2.2 Metodologia Ágil

Nos anos de 1990, começaram a surgir processos alternativos de desenvolvimento de software, em resposta àqueles tradicionais, considerados excessivamente regrados, lentos, burocráticos e inadequados à natureza da atividade. Estes novos processos passaram a ser chamados posteriomente de metodologias ágeis (GOMES; WILLI; REHEM, 2014).

Metodologia ágil é uma forma de planejamento e execução de projetos que visa um menor tempo em todo processo de desenvolvimento do mesmo. Enquanto as abordagens tradicionais normalmente enfatizam o processo sequencial, a partir da coleta de requisitos, planejamento, projeto, desenvolvimento de código, teste e implementação, as metodologias ágeis enfatizam um fluxo de trabalho iterativo e a entrega incremental de produtos de software em iterações curtas (PATANAKUL; HENRY; LEACH, 2015).

Em 2001, foi feita uma reunião onde participaram 17 profissionais influentes que já praticavam métodos ágeis. Eles observaram os pontos em comum de projetos que tiveram sucesso em suas metodologias e com base nesses pontos criaram o Manifesto para Desenvolvimento Ágil de Software, hoje conhecido como Manifesto Ágil. Além dos doze princípios, encontra-se no Manifesto (BECK, 2001) as linhas orientadoras que fundamentam esta nova abordagem. Os conceitos chave são:

- Indivíduos e interações em vez de processos e ferramentas;
- Software que funciona em vez de documentação abrangente;
- Colaboração do Cliente em vez de negociação de contratos;
- Resposta a modificações em vez de seguir um plano.

Existem já algumas abordagens de modelos ágeis de processo, que seguem esses princípios. Entre eles tem-se o *XP* (*Extreme Programming*) e o *Scrum* (TOMÁS, 2009) que serão discutidos nas subseções posteriores.

2.2.2.1 Extreme Programming

Extreme Programming, também conhecido como *XP*, é uma metodologia de desenvolvimento de software com foco em agilidade de equipes e qualidade de projetos. Ela se apoia em valores como simplicidade, comunicação e *feedback* (MEDEIROS, 2006). Para cumprir com esses valores e os princípios da metodologias ágeis, o *XP* aborda uma série de práticas descritas nessa subseção.

Em Extreme Programming, os requisitos são expressos como cenários (chamados de estórias do usuário), que são implementados diretamente como uma série de tarefas (SOMMERVILLE, 2011). Uma estória de usuário é uma indicação mnemônica de uma conversa em andamento a respeito de um requisito. É uma ferramenta de planejamento que o cliente usa para agendar a implementação de um requisito, com base em sua prioridade e em seu custo estimado (MARTIN, 2011).

Antes de iniciar o desenvolvimento é feito o jogo de planejamento. Nele, o cliente julga o quanto que um recurso é importante e os desenvolvedores estimam quanto custará para implementar esse recurso. Com base no orçamento de todos os recursos estimados, o cliente escolhe aqueles que cabem no orçamento da iteração (MARTIN, 2011).

O desenvolvimento é incremental e sustentado por meio de pequenos e frequentes versões/*releases* do sistema. Os requisitos são baseados em cenários ou em simples estórias de usuário, usadas como base para decidir a funcionalidade que deve ser incluída em um incremento do sistema (SOMMERVILLE, 2011). Para diminuir o erro na especificação é necessária uma constante disponibilidade do cliente para colaborar em dúvidas, alterações e prioridades em um escopo (MEDEIROS, 2006).

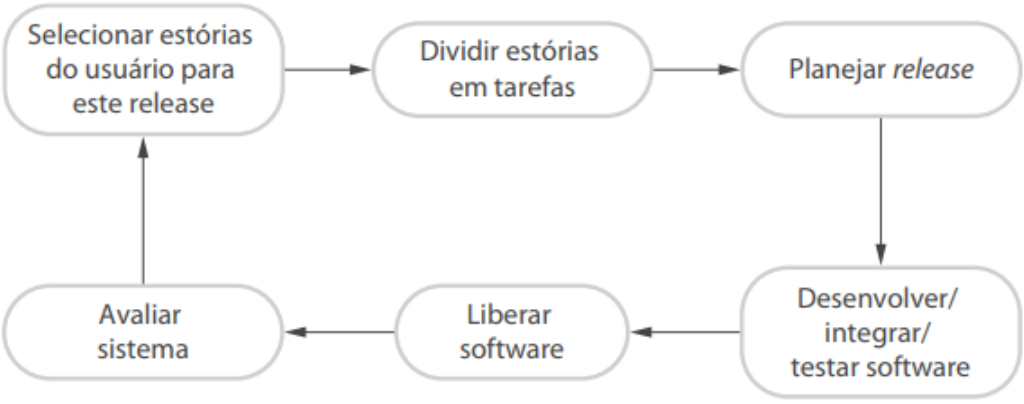
Durante a implementação, o código pode ser escrito em pares de programadores juntos na mesma estação de trabalho, onde um membro de cada par comanda o teclado e digita o código. O outro membro observa o código que está sendo digitado procurando erros e melhorias. Os dois interagem intensamente e os papéis de cada um mudam com frequência (MARTIN, 2011).

Além da programação em pares, em *XP*, o desenvolvimento é orientado a testes, isto é, o teste para o requisito a ser implementado é construído antes do código (SOMMERVILLE, 2011). Ademais, o desenvolvimento do teste é incremental a partir de cenários, a iteração entre escrever casos de testes e código é muita rápida, cerca de um minuto (MARTIN, 2011).

Um problema geral com o desenvolvimento incremental é que ele tende a degradar a estrutura do software. Desse modo, as mudanças para o software tornam-se cada vez mais difíceis de serem implementadas. Por isso, *XP* sugere que o software deve ser constantemente refatorado. Isso significa que a equipe de programação deve estar focada em possíveis melhorias para o software e em implementação imediata destas. Quando um membro da equipe percebe que o código pode ser melhorado, essas melhorias são feitas, mesmo quando não existe necessidade imediata destas (SOMMERVILLE, 2011).

Em resumo, o ciclo de uma iteração em *XP* funciona conforme ilustra a Figura 2.4. O cliente seleciona as estórias de usuário com base no orçamento estimado, as estórias são dividas em tarefas mais detalhadas com critérios de aceitação e com isso o *release* é planejado. Depois é feito o desenvolvimento seguindo as práticas abordadas nesta subseção e o software é liberado para teste dos critérios de aceitação formulado pelo cliente. As correções e possíveis alterações nos requisitos surgidas durante a avaliação do sistema entregue entra para próxima iteração como novas estórias de usuário.

Figura 2.4 – Ciclo de um *release* em Extreme Programming



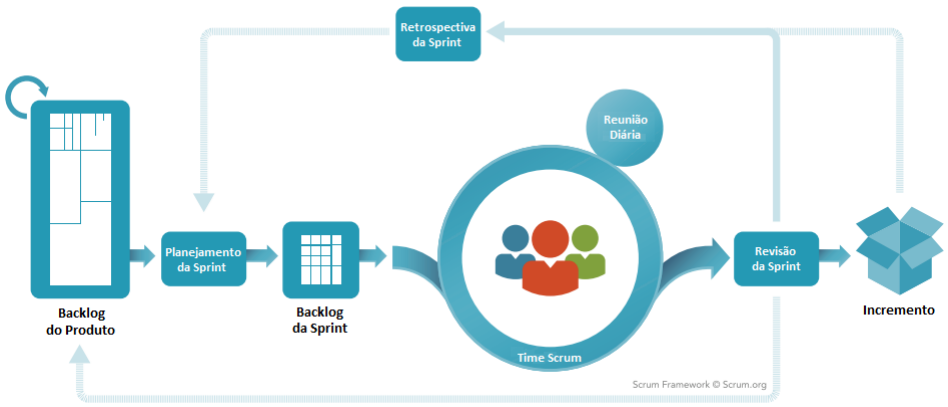
Fonte: (SOMMERVILLE, 2011)

2.2.2.2 Scrum

O *Scrum* é uma estrutura de gerenciamento para desenvolvimento incremental de produtos usando uma ou mais equipes multifuncionais e auto-organizadas (JAMES, 2014). A estru-

tura do *Scrum* enfatiza o *feedback* empírico, a autogestão da equipe e o esforço para construir incrementos de produtos testados adequadamente em iterações curtas (SCHWABER, 2004). Segundo Lei (2015), o *framework* Scrum consiste de times Scrum associados a papéis, eventos, artefatos e regras. As regras são essenciais para unir equipes, eventos e artefatos durante o projeto. Além disso, elas também fornecem uma estrutura agradável para resolver conflitos dentro de um projeto. Na Figura 2.5 é apresentada a representação esquemática de um processo baseado em Scrum.

Figura 2.5 – Processo baseado em Scrum



Fonte: Adaptado de Schwaber e Sutherland (2012)

Nas subseções seguintes, serão detalhados o time Scrum, os eventos e os artefatos.

2.2.2.2.1 Time Scrum

O time Scrum é um modelo de equipe desenvolvido para otimizar a flexibilidade, a criatividade e a produtividade. A equipe tem que ser auto-organizada escolhendo a melhor forma de realizar seu trabalho, em vez de serem dirigidas por outras pessoas fora da equipe. Além disso, são multifuncionais e têm todas as competências necessárias para realizar o trabalho sem depender de outras pessoas que não fazem parte da equipe (SCRUM ORG, 2019). O time Scrum é composto pelo *Product Onwer* (*PO*, ou dono do produto em tradução livre), o *Scrum Master* (Mestre Scrum em tradução livre) e os membros da equipe de desenvolvimento (LEI, 2015). Em Schwaber (2004), esses papéis são detalhados da seguinte maneira:

- **Dono do Produto:** É responsável por gerenciar o *Product Backlog*, que é a lista de requisitos do produto, e maximizar o valor do projeto. Suas funções também incluem a explicação dos itens do *Backlog* do Produto e as metas do projeto para a Equipe de

Desenvolvimento, garantindo que a equipe entenda essas metas e tenha um desempenho de alto nível;

- **Mestre Scrum:** É responsável por garantir que o Scrum seja entendido e aplicado e se comunica com a equipe para garantir que a equipe entenda os planos de longo prazo do projeto. Além disso, age como facilitador removendo impedimentos e um mediador em prováveis conflitos;

- **Equipe de desenvolvimento:** É responsável por implementar e entregar o produto liberável ao final de cada "*Sprint*", para criar um incremento utilizável do produto. Os membros da Equipe de Desenvolvimento gerenciam seu próprio trabalho e são auto-organizados, e não estão agrupados em subequipes.

2.2.2.2.2 Artefatos do Scrum

Os artefatos do *Scrum* representam trabalho ou valor que fornecem transparência e oportunidades de inspeção e adaptação. Eles são projetados especificamente para maximizar a transparência das principais informações de maneira que todos tenham o mesmo entendimento do artefato (SCRUM ORG, 2019). Os artefatos do Scrum são constituídos por *Product Backlog* (*Backlog* do produto em tradução livre), *Sprint Backlog* (Ou *Backlog* da *Sprint*) e Incremento. Lei (2015) define esses items como:

- ***Backlog* do Produto:** É a lista de requisitos, funções, aprimoramentos e correções necessárias no produto. O Dono do Produto é responsável pela criação da lista e pela explicação da perspectiva do projeto para a equipe. Além disso, o *backlog* do produto é dinâmico, pois evolui sempre que há avanço no desenvolvimento do projeto.

- ***Backlog* da *Sprint*:** É a lista de itens no *Backlog* do Produto selecionados para uma *Sprint* específica de acordo com a prioridade dada pelo Dono do Produto. Nele, a Equipe de Desenvolvimento esclarece as funcionalidades do produto que serão implementadas e o que é necessário fazer para atingir o objetivo da *Sprint*.

- **Incremento:** É o conjunto dos requisitos do *Backlog* do Produto que já foram completados no decorrer da *Sprint*.

2.2.2.2.3 Eventos do Scrum

Os eventos do *Scrum* são usados para criar regularidade e minimizar a necessidade de reuniões não definidas no Scrum. Além disso, todos eles têm *Time-Box* (tempo máximo). Quando o *Sprint* começa, sua duração é fixa e não pode ser reduzida ou aumentada. Já os restantes podem terminar sempre que o objetivo do evento for alcançado, garantindo que uma quantidade apropriada de tempo seja gasta sem permitir desperdício no processo (SCRUM ORG, 2019). Tais eventos são *Sprint Planning* (Planejamento da *Sprint*), *Sprint*, *Daily Scrum* (Reunião diária em tradução livre), *Sprint Review* (Revisão da *Sprint*) e *Sprint Retrospective* (Retrospectiva da *Sprint*). Eles podem ser detalhados como segue:

- **Planejamento da *Sprint*:** É uma reunião com *Time-Box* de 2 horas para cada semana de *Sprint*. Nela, o time Scrum monta o *Backlog* da *Sprint* considerando a capacidade da Equipe de Desenvolvimento, as restrições ao desenvolvimento e a prioridade dos itens do *Backlog* do produto (PATANAKUL; HENRY; LEACH, 2015).

- ***Sprint*:** São iterações de tamanho fixo que normalmente podem variar de 2 a 4 semanas de duração. Durante esse período, a equipe tem autoridade total de como completar com sucesso os requisitos selecionados no Planejamento da *Sprint* (COHN, 2009). No final, a equipe entrega um produto implantável e devidadamente testado e que entregue valor para o cliente (PATANAKUL; HENRY; LEACH, 2015).

- **Reunião diária:** É uma reunião realizada diariamente de aproximadamente 15 minutos e é organizada para avaliar o status da *Sprint*, relatar problemas e identificar futuras tarefas (PATANAKUL; RUFO-MCCARRON, 2018).

- **Revisão da *Sprint*:** É uma reunião que pode ser uma demonstração do produto para o Dono do Produto, ou ocasionalmente para os usuários também (LEI, 2015).

- **Retrospectiva da *Sprint*:** É uma reunião baseada nos conceitos de melhoria contínua e nela são analisadas todas as coisas que correram bem, ou não. No final, o time Scrum seleciona cuidadosamente as áreas de interesse de melhoria de processo e planejará a próxima *Sprint*.

2.2.2.3 Comparação entre Extreme Programming e Scrum

Embora ambas metodologias nasceram do movimento ágil e tem seu processo baseado em iterações, há diferenças nas abordagens *Extreme Programming* e *Scrum*. Conforme visto na Subseção 2.2.2.1, o Extreme Programming define uma metodologia ágil com foco nas atividades de desenvolvimento para pequenas e médias equipes, enquanto o Scrum define uma abordagem mais voltada para gerenciamento de projetos (DEVMEDIA, 2012). Valkenhoef et al. (2011) reforçam isso ao dizer que o *XP* oferece muito pouco suporte ao gerenciamento de projetos. Ademais, no que diz respeito a implementação do software, o Scrum, conforme visto na Subseção 2.2.2.2.1, prega que a equipe de desenvolvimento tem total autonomia para desenvolver, não deixando claro qual a melhor estratégia a ser realizada no desenvolvimento. Por isso, neste trabalho foi escolhido adotar o Scrum como metodologia de gerenciamento do projeto e algumas práticas do *XP* foram utilizadas durante a implementação como o desenvolvimento orientado a testes e a refatoração. Quanto à programação em pares, foram escolhidas a abordagem de programação individual e a revisão do código em dupla, de acordo com a sugestão de Sommerville (2011).

3 METODOLOGIA

Neste capítulo, objetiva-se transcrever os passos necessários para a concepção do projeto. Serão apresentados os materiais utilizados para o desenvolvimento do software, além da metodologia empregada para levantar os requisitos da aplicação desenvolvida.

3.1 Materiais

As ferramentas utilizadas para o desenvolvimento foram:

- Visual Studio como interface de desenvolvimento;
- ASP .NET MVC 6 como framework;
- Linguagens C#, Razor, CSS, JS;
- Banco de dados Sybase e SQL Server;
- Sistema Operacional Windows 10.

3.2 Criação do *Backlog* do produto

A metodologia abordada para levantar os requisitos e dar início ao desenvolvimento foi o Scrum. Sendo assim, o primeiro passo a ser feito é levantar o *Backlog* do produto. Para isso, foi mapeado o funcionamento do processo e levantados todos os gargalos do processo que deveriam ser eliminados com o novo sistema.

Com base nas informações levantadas na Seção 2.1, construiu-se o *backlog* do produto exibido na Tabela 3.1 em conjunto com o *Product Owner*. Nesta tabela, é exibida a descrição da funcionalidade que o sistema deverá ter e sua prioridade (feita pelo *PO*) na construção, sendo 1 a maior prioridade.

A funcionalidade de *id* 1 foi pensada para otimizar o tempo gasto do colaborador transferindo a responsabilidade de cadastrar a cotação para o próprio fornecedor. Uma vez que veículos parados elevam os custos da empresa, o requisisto 2 foi considerado para o colaborador poder analiser o tempo de entrega da peça e, consequentemente, diminuir custos. As funcionalidades de *id* 3, 4, 5 e 6 foram levantadas para manter a forma de trabalho que os colaboradores já estavam acostumados com o sistema antigo. Ademais, o requisito 5 foi considerado para

Tabela 3.1 – *Backlog* do produto

Id	Descrição	Prioridade
1	Permitir que o próprio fornecedor preencha a cotação sem necessidade de logar no sistema	1
2	Permitir que, junto com o valor da cotação, o sistema salve o prazo de entrega	2
3	Permitir que o colaborador que compra as peças visualize uma lista de solicitações de compra	3
4	Permitir que o colaborador tome posse da cotação e somente ele e o supervisor possam visualizar o andamento	7
5	Permitir que o supervisor troque a posse da cotação	8
6	Permitir que o colaborador preencha a cotação caso ele entre em contato com o fornecedor	4
7	Permitir que o colaborador visualize a situação das cotações que estão aguardando resposta do fornecedor	5
8	Permitir que o colaborador adicione mais uma peça na solicitação	10
9	Exibir as cotações de uma peça ordenada com um critério de otimização que leva em conta preço e tempo de entrega	9
10	Permitir que o colaborador cancele a solicitação de compra	11
11	Cadastrar valores limites para algumas peças e nunca deixar que o colaborador aprove uma compra com preço maior que este limite	12
12	Permitir que o software se integre com outros sistemas da empresa	6

Fonte: Do autor

outro colaborador poder continuar o procedimento no caso de falta ou afastamento da pessoa que tiver a posse da cotação no momento.

Uma vez que o colaborador iria precisar esperar as respostas dos fornecedores, o requisito de *id* 7 foi pensado para que ele possa acompanhar as situações sem ter que entrar individualmente nas telas de cada cotação. Já o 8 foi colocado para o caso do mesmo serviço de manutenção de um veículo exigir uma nova peça e não necessitar de abrir outra solicitação de compra.

Como outra forma de reduzir custos, foi abordado os requisitos de *id* 9 e 11. O 9 trata-se de montar uma equação levando os custos de manter um carro parado por dia e o preço da peça. Otimizando essa equação, teria-se a peça com menor custo, uma vez que dependendo do prazo compensaria mais pagar mais caro numa peça do que esperar mais por ela. Já o 11 seria o caso de forçar o colaborador a não comprar nenhuma peça com um preço maior que um pré-determinado. Entretanto, tal requisito exige bastante esforço pela necessidade de fazer uma tela para cadastrar os valores para todas as peças.

O requisito de *id* 10 foi considerado para o caso de ser mais viável a compra da peça direto do fornecedor de manutenção, neste caso a compra entraria no custo do serviço e a

solicitação de compra deve ser cancelada. E por último, o requisisto 12 é importante para aproveitar as informações já existentes no sistema de fornecedores com intuito de facilitar o software a conseguir os *e-mails* para solicitar as cotações. Ademais, a integração com o sistema de monitoramento é de suma importância para empresa poder acompanhar todo o processo de compra e não gerar atraso na entrega. Além disso, sem a integração com o APD seria necessário outro processo para pagamento dos fornecedores.

3.3 Planejamento da Sprint

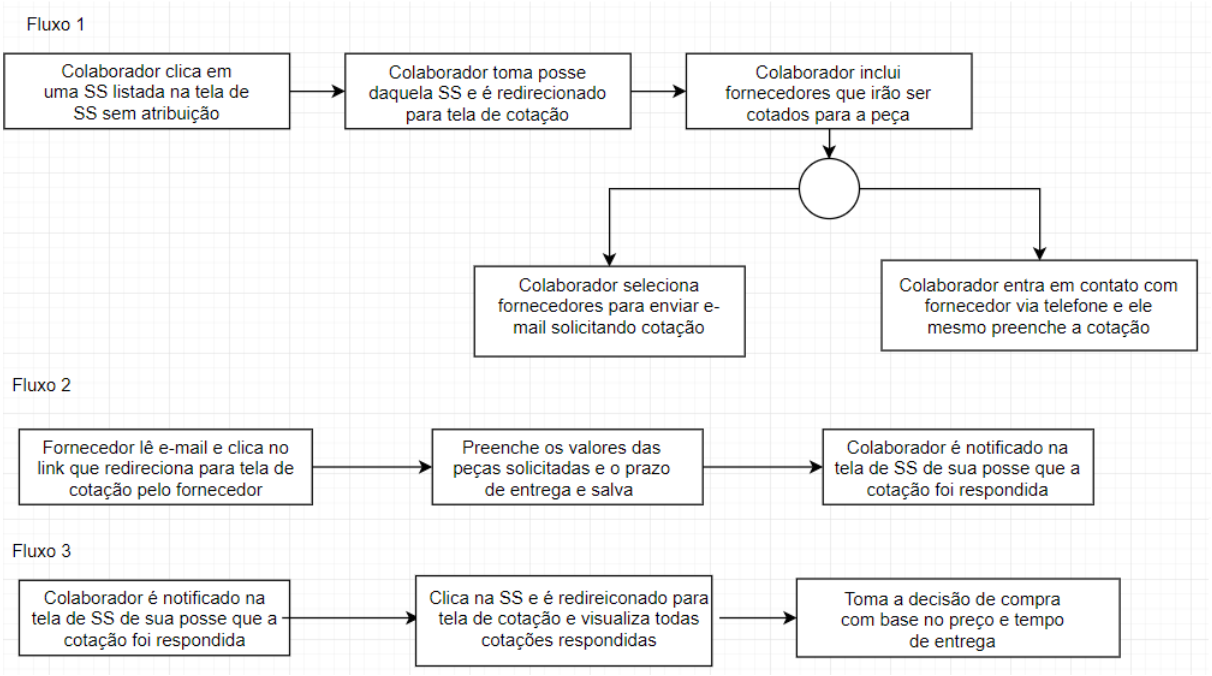
A partir do *backlog* do produto, foi possível construir o *backlog* da *Sprint*. Foram selecionados os requisitos prioritários de tal forma que num período curto de um mês já estivesse com o sistema funcionando em produção. Por esse motivo foram selecionados os requisitos de *id* 1 a 7 e 12 para serem detalhados tecnicamente e conseguir implementá-los durante a *Sprint*. Além disso, os requisitos restantes tiveram o planejamento adiado para a próxima iteração.

Levando em consideração os requisitos de *id* 1 a 7 e 12, foi levantada a construção de 4 telas e 3 fluxos de interação. Com relação as telas, uma exibe as solicitações de compra (SS) sem posse de um colaborador, uma outra a cotação onde há os detalhes da SS e as peças solicitadas para cotação. Além disso, tal tela permite, por meio de um botão, selecionar vários fornecedores e enviar *e-mail* para eles realizarem a cotação. A terceira tela é a que o fornecedor acessa por meio do *link* recebido por *e-mail*, e nele há as informações da peça e os campos para ele preencher a cotação e salvar. A quarta e última tela desenvolvida na Sprint é a de acompanhamento das cotações que têm posse de um colaborador. Nela o colaborador avalia se alguma cotação já tem resposta de fornecedor ou não e permite redirecionar para a tela de cotação onde podem ser vistos os detalhes das cotações enviadas e poder tomar a decisão de compra.

Na Figura 3.1 pode-se ver a descrição dos 3 fluxos possíveis de interação entre as 4 telas desenvolvidas. O fluxo 1 ilustra o momento que o colaborador entra no sistema, vê todas as solicitações de compra, assume a responsabilidade de uma cotação e, a partir daí, ou ele mesmo preenche a cotação após uma comunicação direta com o fornecedor, ou envia a este um *e-mail* com o *link* para que o próprio fornecedor a preencha. O fluxo 2 descreve a interação em que o fornecedor recebe o *e-mail* e vai para a tela de cotação do fornecedor para que ele mesmo impute no sistema sua cotação. E o fluxo 3 descreve o momento em que o colaborador apura se

há respostas das cotações que são de sua responsabilidade e volta para a tela de cotação tomando a decisão de compra.

Figura 3.1 – Os 3 possíveis fluxos de interação entre as 4 telas desenvolvidas.



Fonte: Do autor

Como o requisito de maior prioridade foi de o fornecedor preencher as informações sem logar no sistema, a seguinte estratégia foi utilizada: A tela que o fornecedor acessa ficaria aberta na internet, porém, as informações que dizem qual é a cotação que está sendo preenchida ficariam na URL e criptografadas, para evitar fraude de cotação. Logo, ao enviar o *e-mail*, o sistema criptografa as informações da cotação e coloca as informações na URL. Então, quando o fornecedor acessa o *link*, o sistema pega a informação da URL, descriptografa e reconhece qual peça o fornecedor deve preencher a cotação. Assim, somente com acesso ao *link* recebido no *e-mail*, é possível preencher a cotação na rede aberta.

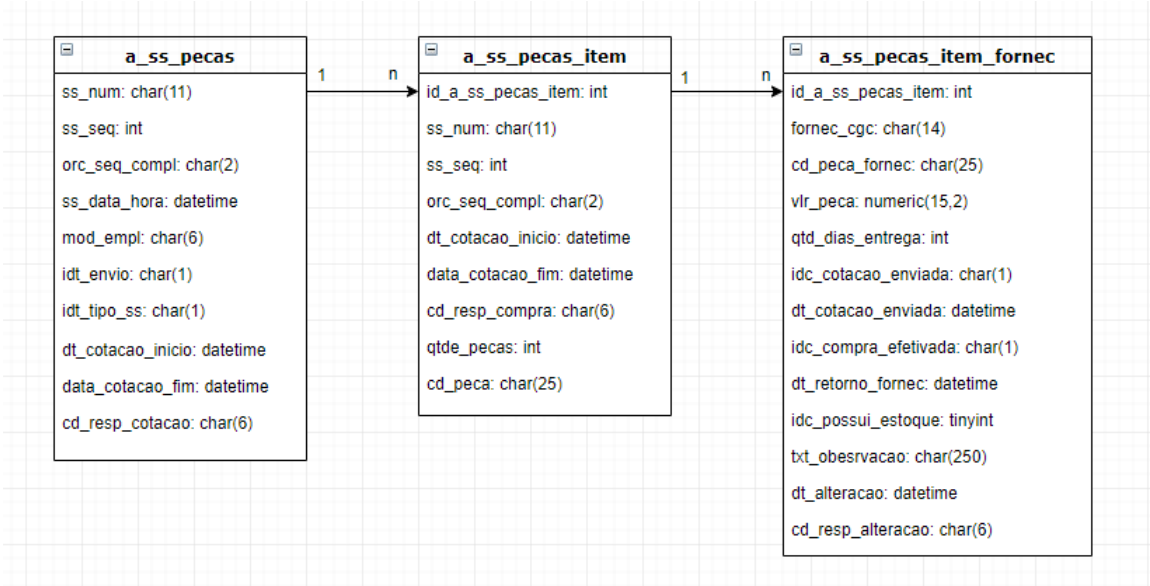
3.4 Implementação

Nesta seção, serão descritos os detalhes da implementação da solução proposta no planejamento da *Sprint*.

3.4.1 Alteração do banco de dados

Para fazer o controle das cotações feitas pelo sistema, foram criadas duas tabelas e uma foi editada no banco de dados SQL Server. A tabela editada foi a *a_ss_pecas* tendo a adição de três colunas, sendo elas *dt_cotacao_inicio*, *dt_cotacao_fim* e *cd_resp_cotacao*. Os campos foram adicionados para ter-se o controle de quando a cotação foi iniciada, finalizada e quem foi o colaborador responsável. Na Figura 3.2 são ilustradas, além da *a_ss_pecas*, as outras duas tabelas: *a_ss_pecas_item* e a *a_ss_pecas_item_fornec*.

Figura 3.2 – As três tabelas utilizadas para fazer o controle das cotações.



Fonte: Do autor

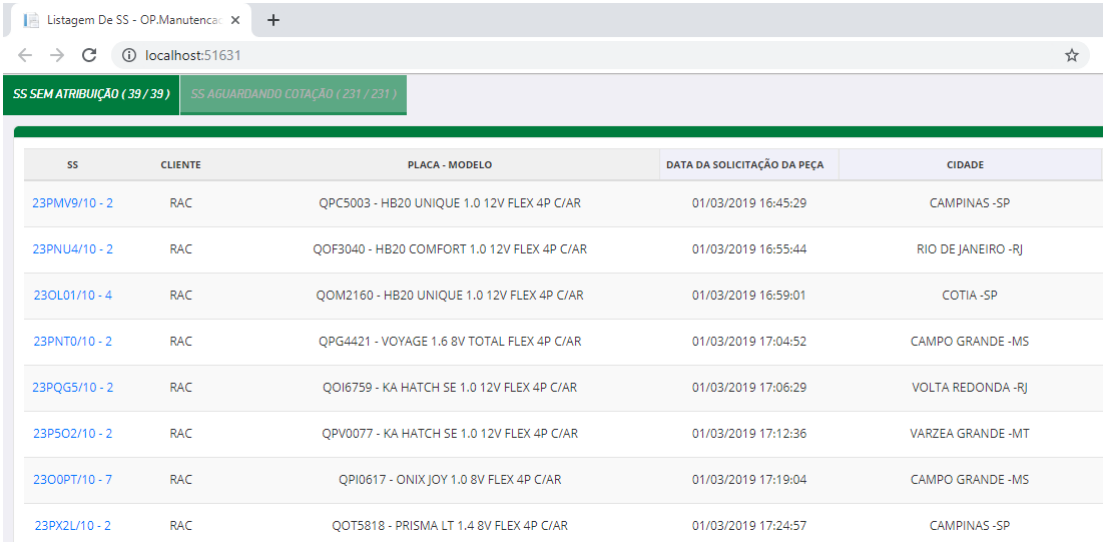
Quando uma manutenção de carro precisa de peças é gerado um registro na tabela *a_ss_pecas* com a chave *ss_num*, *ss_seq* e *orc_seq_compl*. Como uma manutenção pode gerar a necessidade de *n* peças, com isso pode haver *n* registros na *a_ss_pecas_item* que se derivaram de um registro na *a_ss_pecas*. E a chave desta segunda tabela é um número inteiro gerado automaticamente e gravado no campo *id_a_ss_pecas_item*.

Cada peça registrada na *a_ss_pecas_item* pode ter cotação de *n* fornecedores e este registro é gravado na tabela *a_ss_pecas_item_fornec*. A chave desta tabela é a combinação do campo *id_a_ss_pecas_item* e o *fornec_cgc* que é o CNPJ do fornecedor. Além disso, há os campos como *vlr_peca* que armazena o valor da cotação e o *qtd_dias_entrega* que guarda o prazo estimado para entrega da peça.

3.4.2 Tela SS Sem Atribução

Esta tela foi construída para exibir as solicitações de compra que nenhum colaborador tomou posse. Para isso, é feita uma consulta no banco de dados de todas as SS's cujo campo *cd_resp_cotacao* é nulo. Além disso, para cada registro são exibidas as informações da chave da SS, placa e modelo do veículo em manutenção, data que foi solicitada a compra e cidade que o veículo se encontra. A Figura 3.3 ilustra o resultado da implementação e como as informações estão dispostas.

Figura 3.3 – Tela de solicitações de compra sem atribução.



Fonte: Do autor

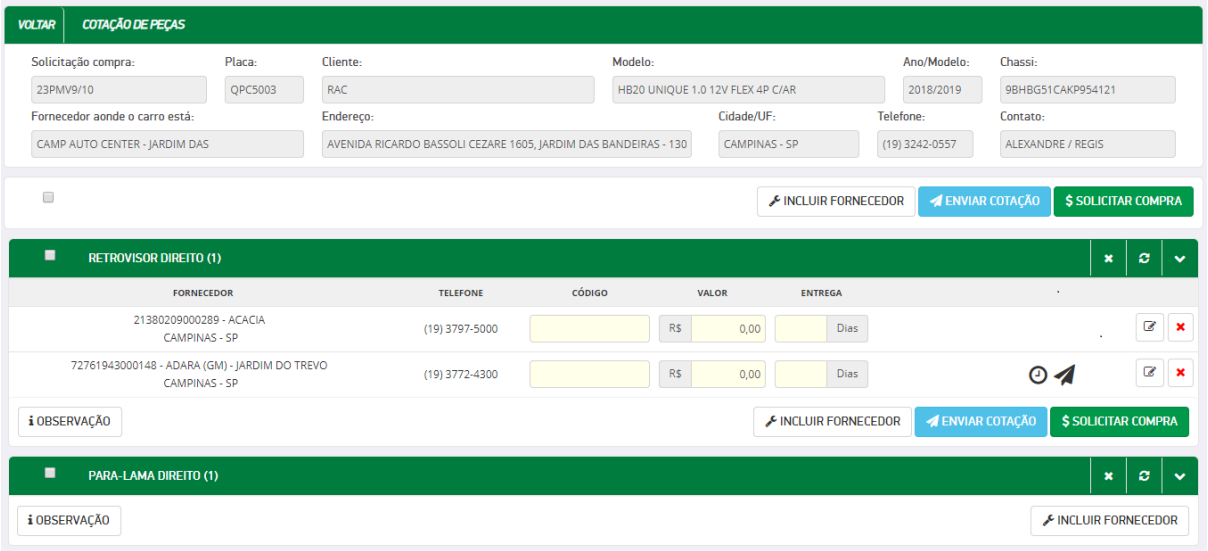
Ao clicar na chave da SS, automaticamente o campo *cd_resp_cotacao* é marcado com a matrícula do usuário logado para indicar que ele tomou posse. Em seguida, ele é redirecionado para a tela de cotação conforme foi especificado no fluxo 1 na Figura 3.1.

3.4.3 Tela de Cotação de Peças

Esta tela foi construída para o colaborador ter informações do carro que está em manutenção e poder realizar e acompanhar a cotação de peças para este veículo. O cabeçalho da tela contém as informações principais do carro, como placa, modelo, ano, chassi, fornecedor que o carro se encontra e o endereço deste. O campo cliente indica se o carro é da plataforma de aluguel de carros para pessoa física ou frota para pessoa jurídica ou de vendas no setor da seminovos. A maneira como estas informações se dispõem podem ser vistas na Figura 3.4.

No corpo da página, são exibidas as peças que aquela manutenção exige compra, onde cada peça mostrada é um registro na *a_ss_pecas_item* correspondente àquela solicitação de

Figura 3.4 – Tela de Cotações de Peças.



Fonte: Do autor

compra. Os botões na parte superior fazem ações com as peças cujo *checkbox* está marcado e o *checkbox* superior marcam ou desmarcam todas as peças da cotação. Já os botões na região da peça fazem ações com aquela peça exclusiva.

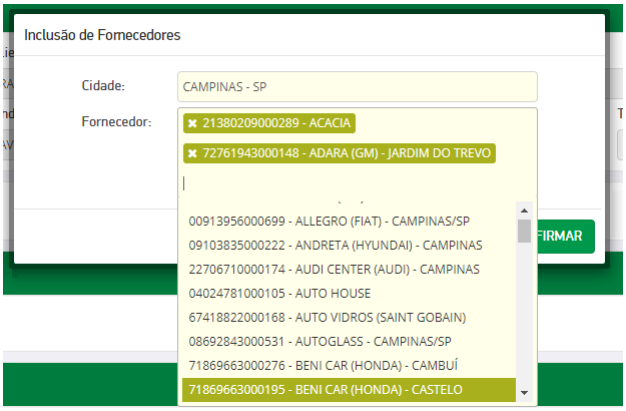
O colaborador tem a opção de preencher ele mesmo a cotação ou clicar no botão que envia o *link* para os próprios fornecedores preencherem. Uma vez preenchido, apenas o colaborador tem a permissão de editar ou excluir a cotação realizada. Uma vez enviada a cotação há ícones que indicam a situação dela. O ícone do relógio exibe há quanto tempo foi enviada a cotação quando se passa o mouse sobre ele e o avião de papel sinaliza que foi enviado *e-mail* para aquele fornecedor.

O botão incluir fornecedor abre o modal conforme ilustra a Figura 3.5. Nele há o *input*, que deve ser colocada a cidade de consulta de fornecedor, por padrão vem a cidade que o veículo se encontra em manutenção. O campo de fornecedor é do tipo *select* que permite seleção de mais de um fornecedor. A tela permite pesquisar por nome e selecionar mais de uma opção por vez. Além disso, a consulta é feita na tabela de fornecedores que fica no banco de dados Sybase.

Ao selecionar todos os fornecedores que deseja incluir na cotação e clicar em confirmar, é feito a inserção na tabela *a_ss_pecas_item_fornec* com a chave sendo a combinação do *id* do registro na *a_ss_pecas_item* correspondente e o CNPJ do fornecedor incluído. Assim, é listado abaixo da peça os fornecedores onde serão feitas as cotações conforme exibido na Figura 3.4.

O botão enviar cotação abre um modal de confirmação com as opções de enviar *e-mail* a todos os fornecedores ou apenas aos que não foram enviados ainda. Ao confirmar a opção, o

Figura 3.5 – Modal de inclusão de fornecedores.



Fonte: Do autor

e-mail do fornecedor é consultado na base de dados do Sybase e o *link* criptografado é criado e enviado.

A cotação feita por um fornecedor foi formulada para ser feita por solicitação de compra (SS), sendo assim, se forem cotadas mais de uma peça para uma mesma SS, ele responderá as cotações de todas as peças numa tela somente. Dessa forma, as informações que precisariam ser criptografadas seriam a chave da SS e o CNPJ do fornecedor. No exemplo da Figura 3.4 as informações seriam $ss\_num = 23PMV9$, $ss\_seq = 10$, $seq\_orc\_compl = 2$ e $cnpj = 72761943000148$. Para não ficar visível ao fornecedor o que significa cada campo, na montagem da URL os parâmetros da *querystring* *ss_num* é utilizado *a*, *ss_seq* é *b*, *seq_orc_compl* é *c* e *cnpj* é *d*. Logo o final da URL com as informações criptografadas fica:

*<.../CotacaoPecaFornecedor/IniciarCotacao?**a**=ss_num_criptografado**&b**=ss_seq_criptografado**&c**=seq_orc_compl_criptografado**&d**=cnpj_criptografado>*

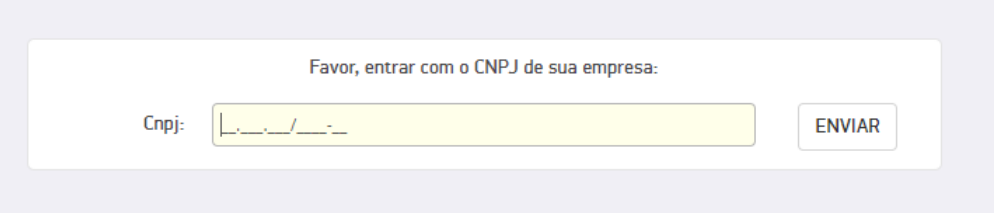
O tipo de criptografia utilizado foi o RSA (Acrônimo dos sobrenomes de Ron Rivest, Adi Shamir e Leonard Adleman, criadores do algoritmo) e para gerar os dados colocados na URL foi utilizada uma chave pública. Na seção seguinte será mostrado como é feito o acesso à tela do fornecedor. Para tornar possível montá-la, é necessário fazer uso da chave privada para recuperar as informações da cotação e exibir para o fornecedor a cotação que ele deve preencher.

O botão solicitar compra valida se há alguma cotação selecionada, se não, exibe-se uma mensagem na tela dizendo que é preciso selecionar uma cotação. Caso tenha-se uma cotação selecionada, um *e-mail* é enviado ao fornecedor confirmando a intenção de compra e são criados registros no banco que indicam um orçamento de compra de peça aprovado e outro registro numa tabela que posteriormente é encaminhada para o sistema de pagamento.

3.4.4 Tela de Cotação de Peças do Fornecedor

A tela de cotação do fornecedor foi construída para que ele possa acessá-la na rede aberta fora da empresa e ele poder fazer sua cotação sem a necessidade de credenciamento. Conforme descrito na seção anterior, as informações da cotação que serão preenchidas está na URL. Para garantir uma segurança mínima de que o *e-mail* com o *link* chegou para a pessoa certa, é exigido, antes de acessar a tela de cotação, que o usuário entre com o CPNJ da empresa que ele trabalha conforme ilustra a Figura 3.6. Ao clicar em entrar, o CNPJ da URL é descriptografado com a chave privada e comparado ao preenchido pelo usuário. Se coincidirem, então é feito o redirecionamento para a tela de cotação do fornecedor.

Figura 3.6 – Tela de solicitações de compra sem atribuição.



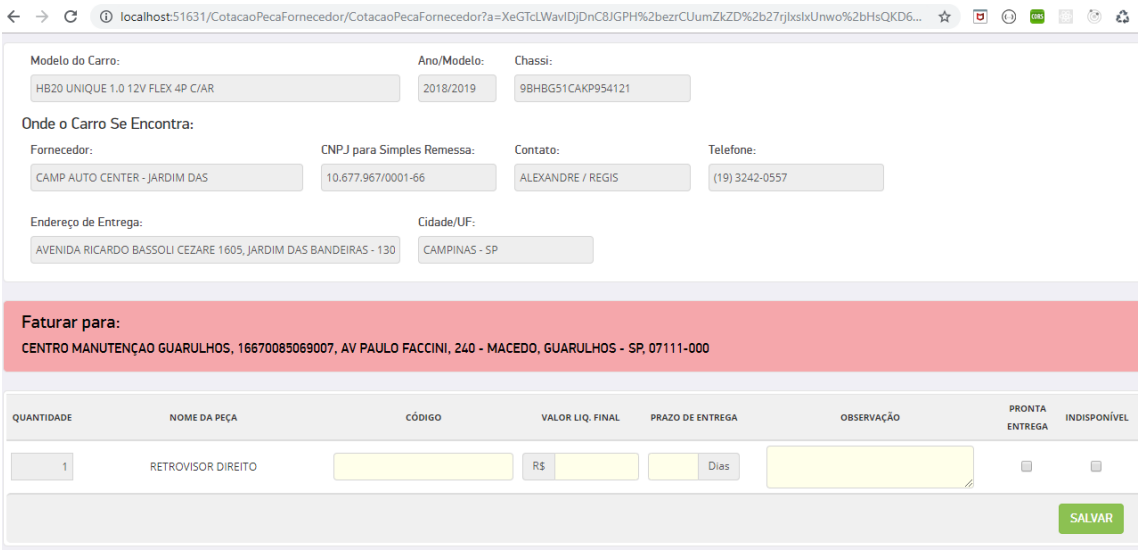
Fonte: Do autor

A tela de cotação do fornecedor pode ser vista na Figura 3.7. Assim como a tela de cotação acessada pelo colaborador, ela traz informações do veículo e do local onde ele se encontra, pois caso a peça seja comprada é pra este local que o fornecedor deve submeter a peça e considerar para cálculo do preço do frete. Além disso, traz a informação para quem deve ser faturada a compra.

Com base no *ss_num*, *ss_seq* e *seq_orc_compl*, obtidos da URL e descriptografados com a chave privada, é possível obter a solicitação de compra e, com isso, todas as peças (linhas na tabela *a_ss_pecas_item*) dessa SS cujo registro é existente na *a_ss_pecas_item_fornec* com o CNPJ da empresa que está fazendo a cotação. Tais peças são mostradas na parte inferior da página conforme é exibido na Figura 3.7.

Na região que exibe as peças é possível ver a quantidade de peças, o nome da peça a ser cotada, o código interno que ele utiliza para registrar a peça, valor da peça, prazo de entrega, alguma observação limitada a 250 caracteres e se é pronta entrega ou a peça está indisponível. O código interno é para facilitar na comunicação entre ele e o colaborador responsável pela compra caso haja necessidade. E, ao preencher todas as informações necessárias e clicar em salvar, o sistema pega os valores da chave na tabela *a_ss_pecas_item_fornec* e atualiza os campos correspondentes com as informações preenchidas.

Figura 3.7 – Tela de cotação de peças do fornecedor.



Fonte: Do autor

3.4.5 Tela SS Aguardando Cotação

Esta tela foi construída para exibir as solicitações de compra que o colaborador tomou posse e as que estão em andamento. Para isso, é feita uma consulta no banco de dados de todas as SS's cujo campo *cd_resp_cotacao* está com sua matrícula e o campo *dt_cotacao_fim* é nulo (isto é, cotação ainda não terminou). No caso em que o usuário *logado* seja o supervisor, não é feita a discriminação por número de matrícula e SS's com posse de todas as matrículas são trazidas. O resultado da implementação desta tela pode ser visto na Figura 3.8. Além das mesmas informações por SS mostradas na tela de SS's sem atribuição, são exibidos a quantidade de peças faltantes, data da posse, nome do responsável e um botão de editar. Estes dois últimos só aparecem para os supervisores.

A coluna de quantidade faltante mostra a relação de quantas peças faltam ser compradas com o total de peças solicitadas. A data de posse, como o próprio nome diz, exibe a data que o colaborador tomou posse da cotação, isto é, quando ele clicou nela na tela de SS's sem atribuição.

O botão de editar abre um modal com a informação do colaborador responsável e um *select*. O *select* que lista todos os colaboradores do compra de peça e permite o supervisor selecionar outro colaborador para tomar a posse da SS. A posse somente é alterada ao clicar no botão de confirmação do modal.

Além dessas informações, há ícones que auxiliam o colaborador no acompanhamento da cotação. O ícone do "avião de papel" diz que para aquela cotação há *e-mails* enviados para

Figura 3.8 – Tela de solicitações de compra que estão em andamento.

SS	CLIENTE	PLACA - MODELO	QUANTIDADE FALTANTES	DATA DA POSSE	DATA SOLICITAÇÃO DE PEÇAS	CIDADE	RESPONSÁVEL
23O88R/10 - 2	RAC	QPH6766 - ONIX LT 1.0 8V FLEX 4P C/AR	19 / 22	26/02/2019 17:36:48	26/02/2019 16:37:01	RIO DE JANEIRO -RJ	MARLON SOUZA VIEIRA
23NXEX/42 - 2	Carro Localiza Fleet	PWS1623 - FOCUS HATCH SE PLUS 2.0 16V DIRECT FLEX 4P C/AR - AUTOMÁTICO	2 / 3	26/02/2019 17:52:30	26/02/2019 17:02:24	RIO DE JANEIRO -RJ	MARLON SOUZA VIEIRA
23LZOL/10 - 4	RAC	QNJ8627 - KA+ SEDAN SE 1.5 16V FLEX 4P C/AR	2 / 5	28/02/2019 09:32:38	27/02/2019 10:33:47	RIO DE JANEIRO -RJ	MARLON SOUZA VIEIRA
23NB2W/42 - 2	Carro Localiza Fleet	QNA3760 - ONIX JOY 1.0 8V FLEX 4P C/AR	0 / 1	27/02/2019 10:50:16	27/02/2019 10:41:22	SAO PAULO -SP	GLENDSON BELCHIOR RODRIGUES SILVA
23PMWE/10 - 2	RAC	QPB2921 - GOL 1.6 8V FLEX 4P C/ AR	1 / 1	01/03/2019 16:45:36	01/03/2019 16:44:40	SAO PAULO -SP	ANDRE RESENDE NASCIMENTO
23PMV9/10 - 2	RAC	QPC5003 - HB20 UNIQUE 1.0 12V FLEX 4P C/AR	2 / 2	28/04/2019 20:10:31	01/03/2019 16:45:29	CAMPINAS -SP	LUCAS ARIGONI GUIMARAES

Fonte: Do autor

fornecedores. Já o "i", quando o *mouse* é colocado em cima, exibe informações como há quanto tempo a cotação foi respondida ou o *e-mail* foi enviado. E o ícone de “carta com uma seta” informa que há cotações respondidas por fornecedores.

Já a coluna “SS” exibe a chave da solicitação de compra de peça. Nela, há ação de clique idêntica à da tela de *SS Sem Atribuição* que é o redirecionamento para a tela de cotação. Com isto, o colaborador pode ver mais detalhes das cotações respondidas e tomar a decisão de compra.

O fluxo proposto e explicado na Seção 3.3 por meio das telas descritas nesta seção permitiu uma nova forma de executar o processo de compra de peça. Os resultados obtidos devido a utilização deste novo fluxo de processo serão discutidos no capítulo subsequente onde foram analisados os impactos na produtividade e, consequentemente, o retorno financeiro do investimento.

4 RESULTADOS E DISCUSSÃO

Neste capítulo serão mostrados e discutidos os resultados obtidos após a utilização em produção do software desenvolvido. Primeiramente, será apresentada a análise da produtividade, posteriormente, os ganhos econômicos e, finalmente, os *feedbacks* coletados dos colaboradores e fornecedores que utilizaram o sistema.

O desenvolvimento do software deu-se início em 2 de agosto de 2017 e o término em 25 do mesmo mês, tendo uma *Sprint* de aproximadamente três semanas, e foi colocado em produção no dia 29. No mês de setembro o novo sistema foi utilizado em paralelo com o antigo para mitigação dos riscos no caso de falha do sistema. Já em novembro, o sistema antigo foi completamente substituído pelo desenvolvido por este trabalho.

4.1 Análise da produtividade

Para avaliar o valor que o software agregou ao negócio, foram levados em consideração os seguintes elementos antes e depois da implantação em produção: a quantidade de peças compradas por colaborador, o tempo médio de compra por colaborador e o custo médio da peça. Nas Tabelas 4.1, 4.2 e 4.3 são exibidos os dados de solicitações de compra, quantidade de peças e o total gasto nas compras. O período entre maio e julho são de antes da utilização do software desenvolvido neste trabalho, já os dados de novembro em diante são posteriores à implantação em produção. Os meses entre estes intervalos não foram analisados por se tratar do período de transição da utilização do método antigo de compra para o novo.

É possível notar que as compras de peça aumentaram gradativamente ao longo dos meses e, consequentemente, o dinheiro gasto com elas. Uma contribuição para este salto no número de maio de 2017 à janeiro de 2018 é o aumento de colaboradores no setor, uma vez que no primeiro momento encontrava-se com 4 colaboradores e 7 na última análise. Entretanto, ao comparar junho de 2017 com novembro do mesmo ano (antes e depois da implantação), percebe-se que o número de solicitações de compras atendidas logo após a implantação teve aumento de $137,4\%$.

Para que o número de colaboradores não influencie na avaliação, foi construída a Tabela 4.4. Os valores de solicitações de compras atendidos em média por colaborador foram calculados com média ponderada, tendo como peso a carga horária de trabalho e o número de dias trabalhados no mês. Já o tempo médio gasto para finalizar uma solicitação de compra foi feita a soma de todas as diferenças entre início da posse até a data de compra dividido pelo número total de solicitações. Este último cálculo foi realizado para os meses posteriores à no-

Tabela 4.1 – Dados de solicitações de compras atendidas, quantidade de peças compradas e custo de compra do meses de maio e junho de 2017.

	2017			2017		
	maio			junho		
	SS's Atendidas	Qtd Peças	Valor Total	SS's Atendidas	Qtd Peças	Valor Total
Colaborador 1				196	960	R$388.788,00
Colaborador 2	157	542	R$223.586,00			
Colaborador 3	242	829	R$318.323,00	309	1226	R$550.933,00
Colaborador 4	178	1025	R$372.063,00	186	736	R$358.643,00
Colaborador 5	8	15	R$12.412,00	102	407	R$180.047,00
Soma	585	2411	R$926.405,00	793	3329	R$1.478.412,00

Fonte: Do autor

Tabela 4.2 – Dados de solicitações de compras atendidas, quantidade de peças compradas e custo de compra do meses de julho e novembro de 2017.

	2017			2017		
	julho			novembro		
	SS's Atendidas	Qtd Peças	Valor Total	SS's Atendidas	Qtd Peças	Valor Total
Colaborador 1	313	1211	R$406.980,00	681	2466	R$860.669,00
Colaborador 2	103	722	R$265.622,00	371	1588	R$547.474,00
Colaborador 3	300	1112	R$431.052,00	529	2047	R$699.243,00
Colaborador 4	190	1052	R$387.196,00	397	1348	R$471.781,00
Colaborador 5	208	995	R$448.126,00	534	1648	R$687.313,00
Colaborador 6	3	3	R$611,00	140	425	R$143.023,00
Soma	1117	5094	R$1.939.589,00	2652	9522	R$3.409.504,00

Fonte: Do autor

Tabela 4.3 – Dados de solicitações de compras atendidas, quantidade de peças compradas e custo de compra do meses de dezembro de 2017 e janeiro de 2018.

	2017			2018		
	dezembro			janeiro		
	SS's Atendidas	Qtd Peças	Valor Total	SS's Atendidas	Qtd Peças	Valor Total
Colaborador 1	519	2069	R$812.153,00	536	1816	R$536.703,00
Colaborador 2	389	1616	R$559.539,00	513	2065	R$715.534,00
Colaborador 3	441	1443	R$532.641,00	388	1197	R$456.648,00
Colaborador 4	412	1272	R$450.040,00	620	2063	R$713.391,00
Colaborador 5	478	1754	R$671.566,00	625	2011	R$798.234,00
Colaborador 6	207	595	R$229.562,00	184	512	R$195.610,00
Colaborador 7	213	639	R$229.562,00	472	1755	R$539.252,00
Soma	2659	9388	R$3.485.284,00	3338	11419	R$3.955.376,00

Fonte: Do autor

vembro, pois antes da implantação não havia controle via sistema destes valores e, por isso, foi considerado o tempo de 60 horas passados pelos supervisores da área que mantinham o controle na época.

Tabela 4.4 – Valores médios por colaborador calculados para solicitações atendidas e tempo para finalizar uma solicitação de compra.

	2017					2018
	maio	junho	julho	novembro	dezembro	janeiro
Atendimento médio por colaborador	192	198	223	469	426	499
Tempo médio por colaborador	60h	60h	60h	22h	20h	19,7h

Fonte: Do autor

Comparando os valores antes e depois da implantação, na Tabela 4.4 é notável a diferença na produtividade dos colaboradores. Antes eram atendidos em média 204 solicitações de compra por colaborador em 1 mês gastando um tempo médio de 60 horas por solicitação. Depois, foram atendidas 464 SS's em média, levando um tempo médio de 20,5 horas por solicitação. Portanto, a média de SS atendida teve um acréscimo de 127% e o tempo médio diminuiu em cerca de 66% evidenciando o aumento da produtividade.

Tabela 4.5 – Resposta das cotações.

	2017		2018
	novembro	dezembro	janeiro
Cotações enviadas	34787	40591	59512
Respondidas pelo fornecedor	24024	24835	35922
Respondidas pelo colaborador	16922	16524	21537
Relação de respondidas pelo fornecedor com total respondidas	58,7%	60,0%	62,5%

Fonte: Do autor

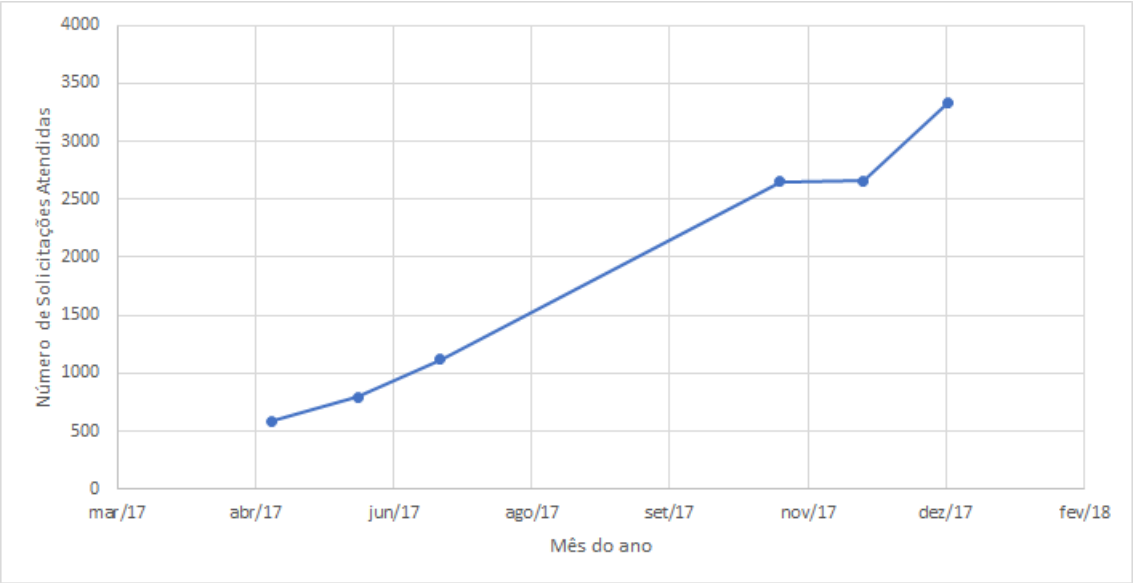
Na Tabela 4.5 foi feita a relação de cotações enviadas e respondidas, nela é possível perceber que nem todas cotações são respondidas. Também, é perceptível o aumento gradual da relação de respondidas pelo fornecedor com o total de cotações respondidas mostrando a adaptação do fornecedor com a nova maneira de fazer cotações. Além disso, tal número é bem próximo do decréscimo do tempo que o colaborador gastava para completar a compra (66%) mostrando que parte desta melhora veio da transferência da tarefa de preenchimento da cotação do colaborador para o fornecedor.

4.2 Análise econômica

Para realizar análise econômica foi necessário comparar o preço do software com o retorno deste para área. A hora de cada desenvolvedor custa em média $R\$125,00$, como no projeto havia 4 desenvolvedores trabalhando 8 horas por dia, isto gera um custo de $R\$4.000,00/dia$. Logo, como a duração do desenvolvimento foi 18 dias, a primeira versão do software teve o valor de $R\$72.000,00$.

Como antes da implantação do software não havia controle do tempo de entrega das peças, não foi possível comparar o tempo de entrega antes e depois. Entretanto, a Tabela 4.4 mostra que o tempo de compra reduziu de 2 dias e meio para 1 dia, isto é, agora o carro fica em média 1 dia e meio a menos na manutenção. Levando em consideração que a diária média de um aluguel é de $R\$100,00$ e que em novembro e dezembro de 2018 e janeiro de 2019 foram atendidos, conforme as Tabelas 4.2 e 4.3, 2652, 2659 e 3338 carros respectivamente, isto gera uma receita potencial de $R\$1.297.350,00$. Portanto, se 5% destes veículos fossem alugados assim que saíssem da manutenção, a primeira versão seria paga nestes 3 meses.

Figura 4.1 – Aumento das solicitações de compra de peça no tempo.



Fonte: Do autor

O gráfico da Figura 4.1 ilustra o aumento das solicitações de compra no período da implantação do software. Tal acréscimo é devido ao aumento da demanda de compra e crescimento da empresa que teve expansão da frota de veículos para locação. Os cálculos feitos na seção anterior mostram que o software proporcionou o atendimento médio de 464 solicitações por colaborador, contra os 204 de antes. Se N é o número de solicitações atendidas por mês e S

o salário médio por colaborador, é possível construir a Equação 4.1 que demonstra a economia de dinheiro em novas contratações no mês em função de N e S:

$$E(N,S) = \left(\frac{1}{204} - \frac{1}{464}\right) \times N \times S \tag{4.1}$$

Assumindo S o valor médio de $R\$3.000,00$, como valor pago no salário bruto, e N os valores do meses de novembro, dezembro de 2017 e janeiro de 2018 é possível calcular com a Equação 4.1 a economia de $R\$71.270,91$. Portanto, considerando a economia em contratação, a primeira versão do software seria paga nos primeiros 3 meses e a partir disso geraria lucro para a empresa.

4.3 *Feedbacks* coletados

O software foi bem recebido pelos colaboradores e *feedbacks* foram colhidos. Um deles foi a necessidade de ser acrescentado a possibilidade de filtrar solicitações de compra na tela de *SS Aguardando Cotação*. Tal requisito foi necessário porque cada colaborador trabalha atendendo solicitações de compra de uma determinada região, e como inicialmente o software não tinha esta funcionalidade, era necessário usar a ferramenta de busca do navegador para encontrar as solicitações da região em que trabalhava. Tal requisito foi levado para *backlog* do produto para ser detalhado e desenvolvido nas próximas *Sprints*.

Figura 4.2 – Tela de cotação do fornecedor quando aberta em uma tela de *tablet*.

Fonte: Do autor

Por parte dos fornecedores, o sistema também foi bem recebido e apresentou apenas dois problemas. O primeiro foi o fato do *link* para a tela de cotação recebido via *e-mail* ficar ocultado no *HTML* em um "clique aqui". Nos caso de fornecedores que utilizam *e-mails* que não interpretam *HTML*, que por sua vez, o *link* ficava solto e ocupava boa parte do *e-mail*, porque

os valores quando criptografados viravam textos muito grandes, gerando o desconforto de ter que copiar e colar em outra aba do navegador. O segundo foi a tela de cotação do fornecedor não ter sido construída para ser aberta em celular ou *tablet*, e quando isso ocorria os campos de preenchimento ficavam espremidos e não era possível enxergar o valor preenchido conforme ilustra a Figura 4.2. Ambas reclamações foram avaliadas, porém, por serem atípicas, não foram colocadas no *backlog* do produto como prioridade.

5 CONSIDERAÇÕES FINAIS

Este trabalho propôs o desenvolvimento de um software para compra de peças para substituição do processo já existente numa empresa de aluguel de carros a partir do estudo do problema e das restrições do processo. A metodologia ágil aplicada mostrou-se bastante eficaz em dividir o software em vários pacotes entregáveis e priorizar os que mais agregariam valor. Com isso, foi possível entregar a primeira versão no prazo de 1 mês.

A análise realizada no capítulo anterior ilustrou que este software aumentou significativamente a produtividade dos colaboradores que trabalham no setor de compras de peça. Logo, um dos objetivos que era mostrar a influência na produtividade foi alcançado. Além disso, embora não tenha sido feita a avaliação dos valores pagos nas peças antes e depois da implantação do sistema, considerando que 5% dos carros fossem alugados imediatamente após a saída da manutenção, a diminuição do tempo de compra de peça faria com que o software fosse pago nos primeiros 3 meses. Ademais, admitindo apenas a não necessidade de contratar mais funcionários para aumentar o número de peças compradas, o investimento também seria pago nos primeiros 3 meses e posteriormente geraria lucro.

Para trabalhos futuros será considerada, conforme discutido na seção 4.3, a retroalimentação do *backlog* do produto com o novo requisito de acrescentar filtros na tela de *SS's Aguardando Cotação* para ser implementado nas próximas *Sprints*, além dos requisitos de *ids* 8 à 11 da Tabela 3.1. Ademais, foi discutida a possibilidade de, no momento da cotação de uma peça, pesquisar os registros de compra dela nos últimos 3 meses para o mesmo modelo e fabricante do carro da solicitação e trazer a tela o menor valor pago. Assim, o colaborador teria um critério de comparação para não comprar uma peça com valor acima do mercado.

REFERÊNCIAS

BECK, K. Manifesto for agile software development. 2001. Disponível em: <http://www.agilemanifesto.org>.

CASTELLS, M. **A sociedade em rede**. 29. ed. [S.l.]: Paz Terra, 2009.

COHN, M. **Succeeding with Agile: Software Development Using Scrum**. 1. ed. [S.l.]: Addison-Wesley Professional, 2009.

CORTÉS, M. I. **Fundamentos de Engenharia de Software**. 1. ed. [S.l.: s.n.], 2013.

DEVMEDIA. **Um comparativo entre XP e Scrum**. 2012. Disponível em: <https://www.devmedia.com.br/um-comparativo-entre-xp-e-scrum/25752>. Acesso em: 06 may. 2019.

GOMES, A.; WILLI, R.; REHEM, S. **Métodos Ágeis para Desenvolvimento de Software**. 1. ed. [S.l.]: Bookman, 2014. 3-15 p.

INTELIPOST. Cotação de frete automatizada: reduzindo (drasticamente) seus custos logísticos! nov. 2017. Disponível em: <https://www.intelipost.com.br/blog/cotacao-de-frete-automatizada-reduzindo-drasticamente-seus-custos-logisticos/>.

JAMES, M. **About Scrum**. 2014. Disponível em: <http://scrumreferencecard.com/scrum-reference-card/>. Acesso em: 25 abr. 2019.

LEI, H. A statistical analysis of the effects of scrum and kanban on software development projects. **Robotics and Computer-Integrated Manufacturing**, dez. 2015.

MARINO, L. H. F. de C. Gestão da qualidade e gestão do conhecimento: fatores-chave para produtividade e competitividade empresarial. **Simpósio de Engenharia de Produção**, I, n. 1, p. 1–9, nov. 2006. Disponível em: <http://www.simpep.feb.unesp.br/anais/anais_13/artigos/598.pdf>.

MARTIN, R. **Princípios, padrões e práticas ágeis em C**. 1. ed. [S.l.]: Bookman, 2011.

MEDEIROS, M. P. **Conceitos e Práticas sobre eXtreme Programming**. 2006. Disponível em: <https://www.devmedia.com.br/extreme-programming-conceitos-e-praticas/1498>. Acesso em: 05 may. 2019.

PATANAKUL, P.; HENRY, J.; LEACH, J. A. **Agile Project Execution**. [s.n.], 2015. 311-322 p. Disponível em: <https://onlinelibrary.wiley.com/doi/abs/10.1002/9781119174820.ch11>.

PATANAKUL, P.; RUFO-MCCARRON, R. Transitioning to agile software development: Lessons learned from a government-contracted program. **Robotics and Computer-Integrated Manufacturing**, nov. 2018.

PENA, L. P. M. A tecnologia da informação como ferramenta para inovações de gestão da localiza na indústria de aluguel de carros: estudo de caso. **Universidade Federal da Bahia**, 2010.

PORTER, M. E. **Vantagem Competitiva**. 29. ed. [S.l.]: Campus / Elsevier, 1999.

SCHWABER, K. **Agile Project Management with Scrum**. 1. ed. [S.l.]: Microsoft Press, 2004. 3-15 p.

SCHWABER, K.; SUTHERLAND, J. **Software in 30 Days**. 1. ed. [S.l.]: John Wiley Sons; Edição, 2012.

SCRUM ORG. **Whats is Scrum?** 2019. Disponível em: <https://www.scrum.org/resources/what-is-scrum>. Acesso em: 22 abr. 2019.

SOMMERVILLE, I. **Engenharia De Software**. 9. ed. [S.l.]: Pearson Universidades, 2011.

TICKETLOG. **Solução completa para a gestão de manutenção dos seus veículos**. 2019. Disponível em: <https://www.ticketlog.com.br/gestao-de-manutencao>. Acesso em: 13 may. 2019.

TOMÁS, M. R. S. Métodos ágeis: características, pontos fortes e fracos e possibilidades de aplicação. **IET Working Papers Series**, p. 1–19, jul. 2009. Disponível em: <https://run.unl.pt/bitstream/10362/2003/1/WPSeries_09_2009Tomas.pdf>.

ULTRACAR. **Ultracarweb: Gestão Em Suas Mãos**. 2019. Disponível em: <https://ultracarweb.com/>. Acesso em: 13 may. 2019.

VAART, T. van der; DONK, D. P. van. A critical review of survey-based research in supply chain integration. **International Journal of Production Economics**, jan. 2008.

VALKENHOEF, G. V. et al. Quantitative release planning in extreme programming. **Elsevier**, maio 2011.