UNIVERSIDADE DO VALE DO RIO DOS SINOS – UNISINOS

UNIDADE ACADÊMICA DE PESQUISA E PÓS-GRADUAÇÃO

ESPECIALIZAÇÃO EM QUALIDADE DE SOFTWARE

Fernanda Guimarães Costa

PROBLEMAS NA MANUTENÇÃO DE SISTEMAS LEGADOS: UM ESTUDO DE CASO

São Leopoldo

2018

UNIVERSIDADE DO VALE DO RIO DOS SINOS – UNISINOS

UNIDADE ACADÊMICA DE PESQUISA E PÓS-GRADUAÇÃO

ESPECIALIZAÇÃO EM QUALIDADE DE SOFTWARE

Fernanda Guimarães Costa

PROBLEMAS NA MANUTENÇÃO DE SISTEMAS LEGADOS: UM ESTUDO DE CASO

Trabalho de Conclusão de Curso apresentado como requisito parcial para a obtenção do título de Especialista em Qualidade de Software, pelo curso de Pós-Graduação Lato Sensu em Qualidade de Software da Universidade do Vale do Rio dos Sinos – UNISINOS.
Orientador: Profa. Me. Josiane Brietzke Porto

São Leopoldo

2018

Problemas na manutenção em sistemas legados: um estudo de caso

Fernanda Guimarães Costa

Universidade do Vale do Rio dos Sinos (Unisinos) - 93022-750 - São Leopoldo - RS - Brasil

costafergui@gmail.com

Resumo. *A manutenção e evolução de sistemas apresentam diversos desafios por envolver desenvolvimento em um ambiente legado. Principalmente quando esses sistemas legados apresentam falta de documentação, padronização, tecnologia antiga, de testes ou de uma equipe experiente. Por essas dificuldades que foi desenvolvido um estudo de caso no sistema legado de uma empresa do interior de Minas Gerais, com intuito de melhorar o produto desenvolvido e o processo de manutenção. O sistema estudado é essencial para empresa e para o funcionamento de outros sistemas, sendo necessário apresentar um processo de manutenção que comporte sua evolução. O objetivo do estudo foi propor melhorias no processo de desenvolvimento das equipes envolvidas na manutenção e na evolução do sistema. Os resultados mostram uma proposta de solução, adaptada de boas práticas da literatura, para 5 problemas prioritários no sistema legado, na percepção dos envolvidos.*

Abstract. *The maintenance and evolution of systems present several challenges because they involve development in a legacy environment. Especially when these legacy systems have no documentation, no standardization, have an old technology, no testing or an experienced team. Due to these difficulties, a case study was developed in the legacy system of a company from Minas Gerais, aiming to improve the product developed and the maintenance process.The system studied is essential for the company and for the operation of other systems, and it is necessary to present a maintenance process that supports its evolution. The objective of the study was to propose improvements in the process of development of the teams involved in the maintenance and evolution of the system. The results presented are a solution of problems, adapted from good practices of the literature, for 5 priority problems in the legacy system, the perception of those involved.*

1. Introdução

A partir do instante em que um sistema é implantado, ele entra em um estado constante de mudanças em resposta às alterações de requisitos e às necessidades do cliente [Sommerville 2011]. Nesse estado que se deve manter a preocupação em não perder a qualidade do produto e mantê-lo funcionando. Os sistemas legados podem ser considerados como sistemas críticos de um negócio, por isso muitas vezes devem ser mantidos [Sommerville, 2011]. Por tal motivo, as empresas seguem enfrentando desafios para mantê-los em funcionamento.

Os desafios estão relacionados à manutenção desses sistemas que incluem o desenvolvimento de novas funcionalidades e aprimoramento das antigas, muitas vezes sem

uma compreensão clara de suas regras e lógica. “A modificação de sistemas legados pode ser um processo demorado, repleto de campos minados de arquitetura e de código” [Coleman e McAnallen 2005, p. 1], além de ser influenciado por outro desafio crítico, a tecnologia defasada.

“Mesmo que tenha sido construıdo aplicando as melhores técnicas de projeto e codificação existentes, os sistemas vão se tornando obsoletos em vista das novas tecnologias que são disponibilizadas” [Piekarski e Quináia 2000, p. 33]. Devido as essas diversas dificuldades, segundo Chaves (2004) algumas empresas precisam definir estratégias para tentar manter os sistemas legados, com o objetivo de não serem surpreendidas ou sofrerem os efeitos que esses sistemas podem trazer.

De acordo com Chaves (2004, p. 12), “as organizações encontram muitos problemas no processo de manutenção de sistemas legados e isso causa aumento de custos”, além de complementar que às vezes os negócios dessas organizações estão diretamente ligados a esses sistemas, o que pode causar impactos a organização se não for realizado uma gerência do sistema.

Segundo Gangadharan et al. (2013), ao tentar controlar e a manter sistemas legados, as empresas investem uma parcela significativa e crescente de seu orçamento, por isso cada vez menos é investido em inovações tecnológicas. De acordo com Glass (2006 apud Gangadharan et al. 2013), as empresas podem gastar até 80% de seu orçamento em manutenção dos sistemas. Além de outros estudos mostrarem que o gasto com manutenção tendem a aumentar ao longos dos anos [Gangadharan et al. 2013].

Considerando o contexto acima, o artigo apresenta uma análise desses mesmos desafios também encontrados na empresa XYZ, a fim de melhorar a qualidade de um software desenvolvido. Essa empresa de médio porte mantém o Sistema Legado (SL), atualmente seu sistema de maior relevância. O SL está no mercado desde 2013, neste período já passou por diversas alterações e vários funcionários foram envolvidas em seu desenvolvimento.

Foi desenvolvido utilizando a versão 7 da plataforma *Java EE*, empregando como banco *PostGres 8.4* e sua extensão espacial *PostGIS 1.5*. O *Java* se torna um facilitador no desenvolvimento por ser uma linguagem muito popular, com isso é possível encontrar muitas soluções em fóruns na internet e o *PostGres* disponibiliza a extensão *PostGIS* que atende a necessidade de geoprocessamento do sistema SL.

O sistema analisado neste estudo é um sistema extenso e complexo de âmbito nacional, que apresenta diversos módulos integrados, além de ser customizado e disponibilizado para diferentes clientes. A equipe atual responsável pelo sistema SL se divide entre equipe de manutenção e desenvolvimento de novas funcionalidades. O sistema é modificado e aprimorado através de pedido dos cliente. Há um planejamento realizado pela Gerência necessário para manter o suporte ao sistema, no entanto, o ritmo de desenvolvimentos pode ser lento e exigem programadores específicos. Além disso, existe falta de documentações técnicas e de conhecimentos de regras de negócio por todos os envolvidos.

Considerando as dificuldades em manter sistemas legados e as apresentadas pela empresa XYZ, percebe-se a necessidade de identificação e melhorias de pontos no processo e nas ferramenta utilizadas. Portanto, este artigo tem por motivação identificar pro-

blemas enfrentados ao manter o sistema SL e, com base na literatura, identificar possíveis soluções a serem aplicadas para a melhora do produto entregue.

Com isso, pretende-se responder à seguinte questão: *Dos problemas destacados no processo do sistema legado da empresa XYZ, quais práticas podem ser seguidas para melhorar a qualidade do produto?*

O objetivo principal da pesquisa é realizar um levantamento das dificuldades da empresa XYZ em manter um sistema legado e uma análise de possíveis soluções, visando melhorar a qualidade do processo e do produto. Para atingir esse objetivo definiram-se os seguintes objetivos específicos:

1. Identificar os problemas recorrentes do sistema legado SL e da percepção dos envolvidos;
2. Propor sugestões e soluções, com base na literatura e nos principais problemas encontrados;
3. Analisar essas propostas em relação aos problemas destacados do sistema SL, com os envolvidos.

As técnicas e sugestões analisadas e propostas neste estudo podem apoiar a melhora da qualidade do processo da equipe envolvido com o desenvolvimento do sistema SL, ou seja, melhora no código novo e legado, novas práticas de manutenção, conhecimentos de novas metodologias, aumento da motivação da equipe e, consequentemente, o aumento na qualidade do produto entregue e da confiança dos clientes na empresa XYZ. Além de contribuir com o aumento da competência da empresa para manter futuros e necessários sistemas.

Em um contexto mais amplo, o artigo contribui ao destacar problemas recorrentes em sistemas legados e realizar uma síntese de estudos baseados nessa área, que a cada ano se deve aumentar a atenção para melhor controlar os sistemas já existentes e futuros novos.

O artigo está organizado em: a seção 2 exibe uma visão sobre o conceito de sistemas legados, manutenção de softwares e o trabalhos relacionados. Na seção 3 aborda-se a metodologia aplicada no presente trabalho. Em seguida, na seção 4 apresenta o estudo realizados realizado e seus resultados. Por fim, a seção 5 é responsável pela conclusão.

2. Referencial Teórico

2.1. Sistemas Legados

Na literatura é possível identificar uma diversidade quanto às características dos sistemas legados. Por exemplo, para Sommerville (2011, p. 519), sistema legado "é útil ou até essencial para uma organização, mas que foi desenvolvido com uso de tecnologias ou métodos obsoletos".

Em uma linha similar de pensamento, Warren (1999) afirma que sistema legado é um sistema antigo que ainda está em funcionamento, completando que foi desenvolvido com práticas e tecnologias defasadas, apresentam vida longa, além de ter passado por diversas mudanças.

Complementando, Dayani-Fard (1999, apud Pressman 2011, p. 36) enfatiza que sistemas legados foram "desenvolvidos décadas atrás e tem sido continuamente modificados para se adequar a mudança dos requisitos de negócio e a plataformas computacionais."

Para Coleman e McAnallen (2005), este tipo de sistemas já ultrapassaram os requisitos originais, duraram tempo suficiente para serem substancialmente modificado até que o atual não mais se assemelha ao que foi desenvolvido pela primeira vez.

Porém apesar das definições apresentarem algumas características diferentes(importância para empresa, tempo de vida, tecnologia usada, nível de modificação do original), é possível observar que o conceito de sistema legado, resumidamente, é ser um software que foi implantado e está em uso até hoje que apresenta dificuldades em sua manutenção. Dificuldades estas que podem ser resultados de tecnologia obsoleta, má documentação (técnica/negócio), diversas alterações de requisitos, nível de complexidade do sistema, longo tempo de uso pelo cliente.

O sistema analisado neste estudo demonstra todas as características evidenciadas pelos autores citados, desde sua importância a empresa até as mudanças constantes de requisitos e falta de testes automatizados.

2.2. Manutenção de software

"A manutenção de software requer alteração em uma estrutura ou sistema existente, ou seja, as modificações de software são introduzidas em uma arquitetura existente e deve permitir restrições impostas pela estrutura de design." [ISO/IEC-14764 1999, p. 7]

A Manutenção, segundo Pinto e Braga (2004, p. 49), é um "processo incremental e iterativo em que pequenas modificações são efetuadas no sistema" e essas podem estar relacionadas à correções de erros ou pequenas melhorias, mas nunca podem acarretar em grandes mudança de estrutura.

De acordo com Sommerville (2011, p. 170), existem três tipos de manutenção de software:

1. Correção de defeitos. Erros de codificação são relativamente baratos para serem corrigidos; erros de projeto são mais caros, pois podem implicar reescrever vários componentes de programa. Erros de requisitos são os mais caros para se corrigir devido ao extenso reprojeto de sistema que pode ser necessário.
2. Adaptação ambiental. Esse tipo de manutenção é necessário quando algum aspecto do ambiente do sistema, como o hardware, a plataforma do sistema operacional ou outro software de apoio sofre uma mudança. O sistema de aplicação deve ser modificado para se adaptar a essas mudanças de ambiente.
3. Adição de funcionalidade. Esse tipo de manutenção é necessário quando os requisitos de sistema mudam em resposta às mudanças organizacionais ou de negócios. A escala de mudanças necessárias para o software é, frequentemente, muito maior do que para os outros tipos de manutenção.

Já segundo a ISO/IEC-14764 (1999), a manutenção é dividida em quatro tipos:

- Mudanças Corretiva: correção de erros relacionada ao não atendimento dos requisitos.
- Mudanças Preventiva: correção de falhas antes que aconteçam.

- Mudanças Adaptativa: mudanças para implementar novos requisitos de interface, requisitos ou hardware.
- Mudanças Perfectiva: melhoram o sistema em relação a desempenho ou manutenção.

A manutenção de sistemas é um termo muito amplo, pois pode ser apenas alguma alteração de interface do software ou a correção de uma falha impeditiva de uso. Ela está desde o desenvolvimento até após a implantação do software. Como é uma alteração em sistemas já existentes e consolidados, a manutenção deve ser um processo realizado com planejamento para que não acarretar em um mau funcionamento do sistema.

Utilizando a ISO/IEC-14764 (1999), o sistema legado apresentado realiza apenas a mudança corretiva e, quando solicitado, a mudança adaptativa. Isso por causa da dificuldade em organizar e unir a correção de defeitos do sistema com a sua evolução. É esperado que durante a melhoria do processo do sistema em questão, possa também ser incluído as duas outras mudanças (preventiva e perfectiva) para o aumento da qualidade do produto.

2.3. Práticas para melhorar a manutenção de sistemas legados

De acordo com o guia de implementação MPS.BR (2011, p. 210), "o desenvolvimento e a manutenção de software são caracterizados pela complexidade da interação entre pessoas, processos e tecnologias."

"Para fazer com que os sistemas legados de software sejam mais fáceis de serem mantidos, é preciso aplicar reengenharia nesses sistemas visando a melhoria de sua estrutura e inteligibilidade" [Sommerville 2011, p. 174]. Segundo o autor, a reengenharia pode abranger a documentação, a refatoração, a mudança de linguagem de programação ou modificações e atualizações da estrutura e dos dados, destacando que a funcionalidade do software não deve ser alterada com essas mudanças.

A utilização da reengenharia também é destacada pelos autores Piekarski e Quináia (2000) relacionado a dificuldade em manter um sistema legado e sugerem que o software deve ser reconstruıdo, partindo do já existente. Complementam que ao aplicando a reengenharia de software, o sistema pode ser redocumentado ou reestruturado, complementando que seu objetivo é desenvolver softwares flexıveis e fáceis de modificar. Além de reforçar que o campo da reengenharia está crescendo devido às necessidades de manutenção dos sistemas legados.

Ainda sobre a desatualização ou, até mesmo, a inexistência de documentação e a recuperação de informações do sistema, Chaves (2004) sugere como forma de recuperação aplicar o processo de engenharia reversa. Assim com a engenharia reversa é possível gerar uma visão do sistema que resulta no aumento do entendimento da estrutura e funcionalidades, aumentando a manutenabilidade do sistema.

Em seu estudo, Paduelli (2007, p. 40) diz "a manutenção de software aparece na norma como um dos processos dentro da categoria de processos fundamentais". Esta norma apresentada pelo autor é a ISO/IEC 12207 (ABNT, 2008), que ainda exemplifica com a Figura 1 que exibe quais são as atividades pertencentes a esse processo.

Figura 1. Atividades do processo de manutenção de software e sistema.

O autor ainda informa e explica cada atividade do processo, representado na Figura 1:

- Implantação do processo: essa atividade inclui tarefas para o desenvolvimento de planos e procedimentos para manutenção de software, criando procedimentos para receber, gravar e monitorar pedidos de manutenção, e estabelecer uma interface organizacional com o processo de gerenciamento de configuração. A implementação do processo deve começar cedo no ciclo de vida do software, conforme reforça Pigoski (1996) ao dizer que os planos de manutenção devem ser preparados em paralelo com os planos de desenvolvimento. Essa atividade inclui definir o escopo de manutenção e a identificação e análise de alternativas, bem como organizar e contratar a equipe de manutenção, relacionando recursos e responsabilidades.
- Análise do problema e da modificação: em um primeiro momento, essa atividade tem por objetivo analisar a requisição de manutenção para classificá-la, podendo então determinar o escopo em termos de tamanho, custos e tempo necessário, destacando ainda sua prioridade. As demais tarefas dessa atividade focam o desenvolvimento e a documentação de alternativas para mudança de implementação e aprovação das opções adotadas.
- Implantação da modificação: essa atividade engloba a identificação dos itens que precisam ser modificados e os processos de desenvolvimento que precisarão ser implementados. Outros requisitos da modificação incluem teste e validação de que as modificações estão corretamente implementadas e que os itens não modificados não foram afetados.
- Revisão / aceitação da modificação: as tarefas desta atividade são dedicadas à confirmação da integridade do software modificado e à conclusão dos negócios com o cliente, quando este concorda e aprova satisfatoriamente a conclusão da requisição de manutenção.

Muitos processos de apoio podem ser utilizados aqui, incluindo a garantia da qualidade de processo, o processo de verificação, o processo de validação e o processo de revisão conjunta.

- Migração: corresponde à atividade que ocorrerá quando o software for transferido de um ambiente de operação para outro. Será preciso o desenvolvimento de planos de migração e os usuários precisarão estar cientes dos requisitos, dos motivos do antigo ambiente não ser mais suportado e terem à disposição uma descrição do novo ambiente e sua data de disponibilidade. Outras tarefas desta atividade concentram-se em operações paralelas do novo e antigo ambiente, incluindo a revisão de pós-operação para certificar-se do impacto da mudança de ambiente.
- Descontinuação do software: essa atividade consiste em descontinuar o software por meio da formalização, junto ao cliente, de um plano de descontinuação. Um último comentário referente à norma refere-se ao fato de que ela não representa um modelo fixo ao qual uma organização que a adote se submete. Ao contrário disso, ela funciona como uma estrutura de apoio, devendo a organização que a adotar proceder com adaptações nas recomendações para a sua realidade.

Em geral, a identificação de mudanças e de evolução de software, segundo Sommerville (2011), são cíclicos e se mantém durante a vida do sistema, conforme a Figura 2 abaixo do mesmo autor.

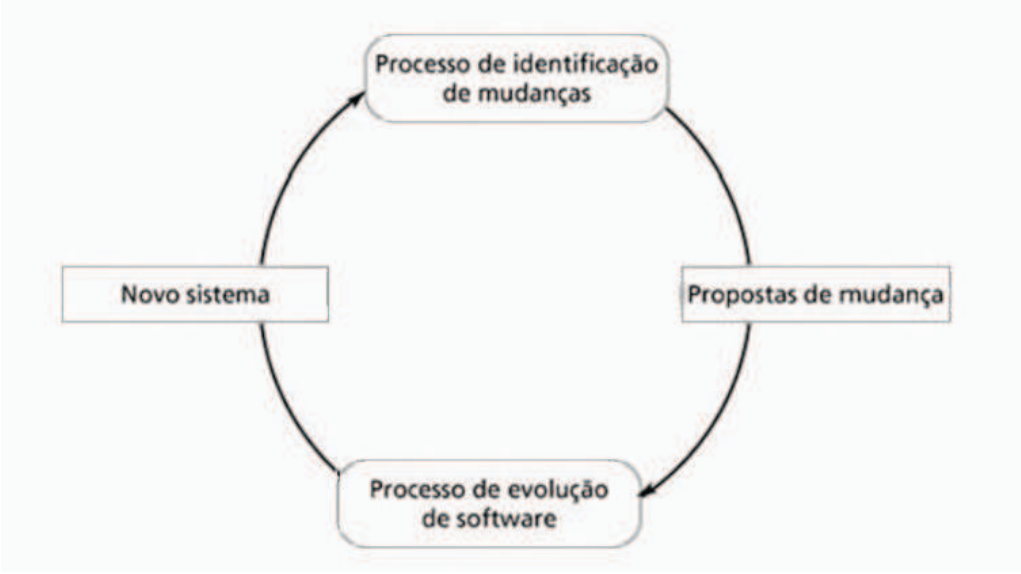

Figura 2. Processos de identificação de mudanças e de evolução.

É compromisso de todos os envolvidos, desde a gerência até a equipe de desenvolvimento, que o sistema deve ser mantido da melhor maneira possível. Ou seja, deve ocorrer as intervenções necessária, novos requisitos do cliente ou correções de defeitos, mas também o sistema deve ser melhorado sempre que possível para que essas intervenções sejam realizadas com maior facilidade e que não gerem impactos futuramente.

As melhorias que não são sugeridas pelo cliente podem gerar aumento no custo de manutenção do sistema, no entanto, deixar o sistema fácil de se manter acaba gerando benefícios para empresa principalmente para sistemas de longa vida. Benefícios estes que podem ser considerados como menor tempo de desenvolvimento de novas funcionalidades, facilidade em identificar e corrigir defeitos, diminuição da complexidade em desenvolver testes e mantê-los, além da atualização das documentações técnicas e de negócio.

2.4. Trabalhos relacionados

Para Pinto e Braga (2004, p. 66), os "sistemas legados existem e sempre existirão, à medida que as tecnologias vigentes tornam-se obsoletas ao longo de um tempo cada vez mais curto." Para eles descartar esses sistemas pode causar prejuízos financeiros e estruturais para as organizações. Complementam que a substituição do sistema legado por um novo sistema pode causar alguns transtornos que devem ser avaliados, ou seja, podem impactar no prazo previsto para a entrega ou na garantia de que terá a mesma funcionalidade fornecida pelo sistema anterior.

O objetivo do estudo de Pinto e Braga (2004) foi demonstrar métodos e tecnologias existentes para o aproveitamento em sistemas legados, fornecendo informações sobre a evolução desses tipos de sistemas, uma descrição de mecanismos de avaliação e por fim, abordagens e tecnologias para a integração da tecnologia ultrapassada a tecnologia atualizada. Ao final, concluem que não existe uma forma ou método geral que defina qual abordagem ou tecnologia utilizar, que cada caso deve ser estudado e se deve aproveitar o melhor de cada tecnologia disponível para adaptar os sistema.

Chaves (2004, p. 2) em seu estudo afirma que "as organizações precisam definir estratégias para manter seus sistemas legados, com o objetivo de não serem surpreendidas e sofrerem os efeitos que tais sistemas podem trazer."O objetivo deste estudo foi apresentar alguns aspectos críticos da manutenção e os riscos de gerenciar um sistema legado, além de apresentar recomendações para lidar com esse sistemas.

Ainda considerando sistemas legados, Chaves (2004) apresenta algumas características que também foram identificadas no sistema SL e serão apresentadas no decorrer deste artigo. Seguem as características destacadas pelo autor:

- Falta de documentação;
- Tecnologias ultrapassadas;
- Sistemas grandes e complexos;
- Elevados números de erros e falhas.

Com relação às práticas de métodos ágeis em sistema legado, Coleman e McAnallen (2005, p. 7), em seu estudo, concluiu que "alguns dos métodos ágeis funcionam bem para sistemas legados e alguns não", no entanto que uma mudança para um modelo misto tem suas vantagens notáveis sobre os modelo únicos e tradicionais. No caso da empresa do artigo, ao qual foi aplicado o método *Extreme Programming* (XP), descobriu-se que as entregas contínuas e a colaboração junto ao cliente geram benefícios, tanto na velocidade de entrega quanto na melhoria das necessidades do usuário, resultando em softwares de maior qualidade com menos erros. Das práticas do XP, a que menos trouxe resultados foi a padrões de código, pois a padronização apresenta dificuldades quando já há código legado e seria necessário reprojetar o código.

Por fim, o estudo de Barrozo et al. (2012) tem o propósito em alcançar a qualidade através do melhoria no código. O autores afirmam que para um software alcançar qualidade não deve apenas estar funcionando e, sim, estar "bem estruturado, compreensível e de fácil manutenibilidade". Segundo os autores, a refatoração "é a alteração de um código fonte, visando melhorar o entendimento e a manutenibilidade sem alterar suas funções externas", ainda completam que mesmo com os benefícios, a refatoração pode apresentar riscos como: "atraso do projeto, introdução de falhas no sistema, tornar o código ilegível e não modificável" [Barrozo et al. 2012, p. 2].

A refatoração também foi sugerida por Chiele (2017) em seu artigo que após identificar esse problema, desenvolveu uma heurística utilizada na refatoração de código legado, destacando a importância em identificar que ponto do sistema deveria ser aplicado a refatoração e teste unitários. No artigo, autor conclui que a aplicação dessas técnicas trouxe melhoras na legibilidade, manutenibilidade e testabilidade do código em questão, destacando que a refatoração podem minimizar os problemas do código a médio e longo prazo diminuindo o esforço para compreensão e manutenções, além de reforçar a necessidade da melhoria ser contínua.

É perceptível a importância dos sistemas legados, pois nesses estudos apresentados é possível notar a busca por melhorias nesses softwares, tanto em relação ao código e estrutura técnica como em negócio e processo de desenvolvimento. Neste artigo também é conhecido a importância da sistema analisado a empresa e, com isso, está sendo considerado seus problemas e a busca por melhorias através dos resultados desses estudos.

Com as contribuições de resultados obtidos e estudos realizados a partir de problemas destacados em sistemas legados, o artigo dá ênfase aos problemas que são recorrentes do sistema analisado e estão presentes nos estudos acima. Além de utilizar as conclusões obtidas como fundamento para a proposta de soluções para melhorias no processo e produto.

O Quadro 1 apresenta uma comparação dos trabalhos exibidos nesta seção.

Quadro 1: Trabalhos relacionados

Autores	**Problemas destacados**	**Solução Proposta**	**Resultados**
Pinto e Braga (2004)	Tecnologia obsoletas e necessidade de modernização do sistema.	Apontar métodos e tecnologias existentes para o aproveitamento em sistemas legados	O objetivo deste estudo não foi destacar uma tecnologia ou abordagem como a correta, e sim fornecer subsídios para a análise da melhor abordagem e tecnologia indicada para cada caso.

Chaves (2004)	Dificuldades no processo de manutenção.	Apresentar possíveis caminhos e recomendações para lidar com sistemas legados.	A decisões relacionadas a sistema legado deve ser bem embasada. Estratégias devem ser definidas e riscos avaliados para execução da reengenharia de software.
Coleman e McAnallen (2005)	Desenvolvimento do sistema lento e alto custo de manutenção.	Introdução da metodologia ágil Extreme Programming (XP).	Alguns métodos funcionam para sistemas legados e outros não, ter uma metodologia híbrida pode ser considerada um boa escolha.
Barrozo e Reis (2012)	Baixa qualidade do software e dificuldade de manutenção.	Refatoração de código	Um código quando refatorado pode se tornar mais legível e eficiente, facilitando a sua manutenção.
Chiele (2017)	Baixa qualidade de código, dificultando a legibilidade e a manutenibilidade	Melhorar a qualidade do software através da aplicação de práticas de refatoração e testes automatizados.	Foi observado umamelhora na legibilidade, manutenibilidade e testabilidade do código. Outra contribuição foi a descoberta de heurística que ajuda no processo de identificação de pontos no,código legado que é necessário a refatoração.

Os estudos do Quadro 1 apresentam certa preocupação na qualidade de códigos e softwares legados e buscam soluções focando em alguns dos problemas que esse ambiente traz. Alguns estudos focam em partes técnicas e estruturais do sistema (arquitetura, código, técnologia usada) e outros focam na parte de negócio e dos profissionais envolvidos, além de propor e aplicar soluções a esses casos (Metodologias de desenvolvimento, métodos já existentes). No entanto, é de entendimento da autora que melhorar a quali-

dade de um sistema legado está correlacionado com a melhoria do processo de desenvolvimento e com a melhoria no código e tecnologias desenvolvidas, dado este motivo que o estuda apresentado não se preocupa em destacar qual área gera um débito ao sistema, e sim identificar todos os problemas a fim de diminuí-los ou até eliminá-los do sistema SL, objeto de análise do presente estudo.

3. Metodologia

Nesta seção é abordada a metodologia utilizada para a realização do presente estudo.

3.1. Delineamento da Pesquisa

A natureza do artigo é do tipo aplicada, pois segundo Gerhardt e Silveira (2009, p. 35), "objetiva gerar conhecimentos para aplicação de práticas, dirigidos à solução de problemas específicos. Envolvendo verdade e interesses locais".

Em relação ao tipo de pesquisa, quanto aos objetivos, o artigo classifica-se como exploratório. "As pesquisas exploratórias têm como principal finalidade desenvolver, esclarecer e modificar conceitos e ideias" [Gil 2008, p. 27].

Quanto à abordagem do estudo, foi utilizada a pesquisa qualitativa. Segundo Gerhardt e Silveira (2009), na pesquisa qualitativa a preocupação é no aprofundamento da compreensão da realidade de um grupo social ou uma organização, ou seja, não tem a preocupação com dados numéricos ou realidade gerais.

Quanto ao tipo de pesquisa, relacionada aos procedimentos, o método de pesquisa escolhido foi o estudo de caso. "O estudo de caso é caracterizado pelo estudo profundo e exaustivo de um ou de poucos objetos, de maneira a permitir o seu conhecimento amplo e detalhado" [Gil 2008, p. 57].

Ainda a esse respeito, Gil (2008) continua que o estudo de caso está sendo aplicado com freqüência por servir com diferentes propósitos, por exemplo, explorar situações da vida real; descrever uma situação a partir de um contexto ou explicar fenômenos em situações muito complexas.

3.2. Unidade de Análise

O estudo desenvolvido foi relacionado a empresa, que por confidencialidade será nomeada como XYZ. Essa empresa está no mercado há cerca de 11 anos e em seu primeiro projeto de relevância foi desenvolvido o sistema analisado neste artigo.

Localizada no sul de Minas Gerais, a empresa XYZ desenvolve inúmeras ferramentas relacionadas a geoprocessamento, desenvolvimento sustentável e gestão ambiental. A empresa desenvolve esses serviços para instituições públicas e privadas, com alcance em âmbito nacional. Atualmente, conta com mais de 120 colaboradores, que incluem analistas de geoprocessamento, gerentes de projetos, gerente de operações, analistas de qualidade de processo, analistas de requisitos, desenvolvedores, analistas de banco de dados, analistas de testes, entre outros.

Para o estudo foi levado em consideração o sistema legado da empresa XYZ e os colaboradores envolvidos direta ou indiretamente em seu desenvolvimento. O propósito de acolher as opiniões dos envolvidos se fez para identificar dentre os problemas encontrados na literatura os que mais se adequam ao sistema analisado. Assim, podendo obter

sugestões de soluções para esses problemas específicos e contribuir para a melhoria no processo do sistema SL.

O sistema legado SL, que será tratado assim para facilitar sua identificação, foi escolhido como fonte de estudo neste artigo por apresentar diversas características de sistemas legados e ter extrema relevância a empresa XYZ. A manutenção desse sistema não só afeta outros produtos da empresa como sistemas externos que necessitam dos dados gerado por ele.

Começou a ser desenvolvido em 2013 e no mesmo ano já tinha sido implantado. Atualmente o sistema SL é compostos por vários módulos, tendo como o principal módulo uma ferramenta de geoprocessamento, e ainda se mantém evoluindo. Por ter uma grande abrangência e importância nacionalmente, o sistema apresenta mais de um cliente, sendo adaptado e customizado de acordo com a necessidades de cada um.

De maneira geral, é um sistema web composto por no mínimo 5 módulos: módulo de cadastramento, módulo que recebe os cadastros, módulo de análise dos cadastros, módulo de relatórios e módulo de acesso externo a dados. A partir da necessidade de um cliente, esse sistema pode ter novos módulos ou alterar os módulos existentes.

Ressaltando que cada módulo apresenta uma complexidade de acordo com suas funcionalidades e são integrados para o funcionamento completo do fluxo do sistema. Como destacado, SL é um sistema extenso e complexo que já apresenta uma vida longa. Durante seu desenvolvimento houveram algumas falhas no gerenciamento que até hoje impactam a sua manutenção e evolução. Nesse cenário é que esse estudo se realiza para contribuir na melhoria desse complexo sistema.

3.3. Coleta de Dados

Para a identificação e apuração dos problemas encontrados no sistema SL foi desenvolvido um questionário que apresentava algumas características encontradas em sistemas legados, na literatura revisada. O questionário foi utilizado como instrumento de coleta de dados e serviu para destacar alguns problemas específicos do sistema SL, na percepção dos principais envolvidos.

O questionário foi elaborado pela autora e seu conteúdo foi composto através da observação da própria, que está inserida na equipe de desenvolvimento do sistema e conhece alguns de seus problemas. Além de conciliar com as características destacadas no artigo de Chaves (2004).

O questionário foi aplicado a 21 membros da empresa que colaboram ou colaboraram com o desenvolvimento do sistema SL. Este questionário foi dividido em dois pontos. O primeiro ponto solicitava dados gerais dos colaboradores, a fim de identificar o perfil profissional e seu envolvimento com o sistema. Já o segundo ponto estava focado no sistema em si, nele era apresentado as características selecionadas pela a autora e foi utilizada a escala de Likert como escolha do nível de concordância do colaborador com a existência daquele problema no sistema legado SL. A escala de *Linkert* foi escolhida para ser utilizada nesta pesquisa por ser comum em questionários e apresentar a opção de informar qual o grau de concordância ou discordância o envolvido tem com relação aquela afirmação. Por fim, era disponibilizado um campo aberto e não obrigatório, para que o colaborador tivesse a opção de destacar algum problema não listado.

Para a divulgação e coleta dos dados foi utilizado a ferramenta *Google Forms* e foi repassado aos colaboradores através da ferramenta de comunicação interna da empresa. O questionário, que encontra se no Apêndice A, foi disponibilizado por um período de 10 dias, do dia 01 de abril de 2018 à 10 de abril de 2018, obtendo 21 respostas consideradas válidas para fins de análise.

Além do questionário, foi utilizado a observação participante que "consiste na participação real do conhecimento na vida da comunidade, do grupo ou de uma situação determinada" [Gil 2008, p. 103]. Ou seja, o observador assume um papel de membro. Ainda para Gil (2008), há duas formas do observado participar do grupo, de forma natural ou artificial. Para este estudo, a autora estava de forma natural observando a situação, pois é um membro (analista de testes) da equipe envolvida no desenvolvimento do sistema estudado.

3.4. Análise de Dados

Como já mencionado foi utilizado o questionário, como técnica de coleta de dados. Para as questões sócio-demográficas e referentes à identificação dos problemas adotou-se estatística básica e descritiva, segundo (Gil, 2008, p. 161) "estes procedimentos possibilitam: caracterizar o que é típico no grupo; indicar a variabilidade dos indivíduos no grupo, e verificar como os indivíduos se distribuem em relação a determinadas variáveis."

Através das respostas obtidas para essas questões foram selecionadas as características que obtiveram mais votação de 4 (Concordo Parcialmente) ou 5 (Concordo totalmente) de 2/3 dos participantes (14). Essas foram consideradas as mais recorrentes e características do sistema SL, sendo que serviram de base para a proposição de melhorias nesse estudo.

Já, para a última questão, que solicitava dados textuais, não foram obtidas respostas consideradas como um problema neste estudo, em razão que apenas um participante respondeu e em seu conteúdo apresentou um questionamento relevante sobre a manutenção Preventiva e Perfectiva. Entretanto, esse ponto já seria discutido em um próprio tópico relacionados aos problemas do sistema SL (tópico 4.2.4).

3.5. Etapas da pesquisa

A partir do objetivo geral e de todas as informações já apresentadas, as etapas que foram realizadas para a construção da pesquisa foram:

1. Realizar um estudo bibliográfico sobre Sistemas Legados, problemas na manutenção e soluções aplicadas;
2. Com auxílio da literatura e da observação da autora na empresa XYZ, mapear alguns problemas recurrentes no sistema legado SL;
3. Desenvolver um questionário com os problemas selecionados e disponibilizar ao funcionários da empresa XYZ que estão/estiveram envolvidos com o desenvolvimento do sistema;
4. Selecionar os problemas com maior destaque no questionário;
5. Através da literatura, propor soluções e boas práticas para melhorar a manutenção do sistema SL.

4. Estudo de Caso

O estudo de caso foi realizado com o objetivo de verificar os problemas de manutenção de software existentes no sistema legado de uma empresa de médio porte.

4.1. Resultados Obtidos

Esta seção apresenta os resultados obtidos através do questionário disponibilizado ao colaboradores. Foram obtidas 21 respostas dos participantes e seguem abaixo.

4.1.1. Dados dos participantes

Inicialmente, são apresentados os dados gerais dos participantes, a fim de caracterizá-los. O intervalo de idade que mais se destacou entre os participantes foi de 25 a 35 anos, com 14 respostas obtidas. A Figura 3 abaixo apresenta o tempo em que o participante encontra-se na empresa.

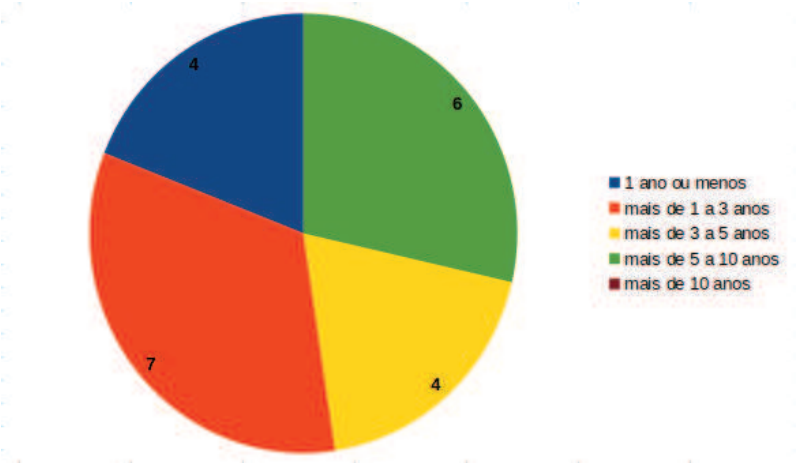

Figura 3. Tempo de trabalho dos participantes na empresa.

É possível verificar através dessa informação acima, que não há uma faixa de tempo que se evidencie, e sim, que todas os participantes estão a menos de 10 anos na empresa. Podendo concluir que há colaboradores que apresentam certa maturidade nos projetos da empresa e também novos, que ainda estão em fase de aprendizado.

Sobre o papel exercido pelos participantes na empresa houve um maior número de desenvolvedores (14) que responderam. Na Figura 4, percebe-se que apenas 1 participante está envolvido com o projeto a mais de 5 anos, que reforça os dados da Figura 3, no qual é possível constatar que há colaboradores não muito experientes envolvidos com um sistema grande e complexo como o sistema legado SL.

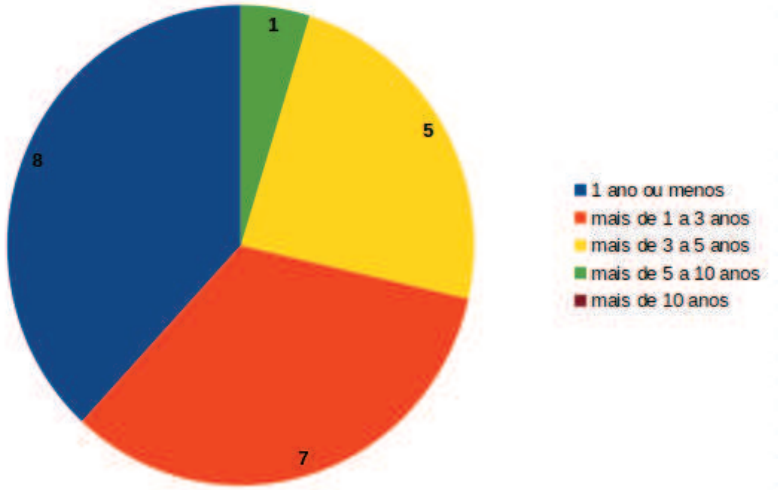

Figura 4. Tempo envolvido com o sistema SL.

Por fim, é apresentado a Figura 5, que confirma ainda mais os dados já mencionados acima. Ou seja, a maior parte dos colaboradores participantes (16) não estavam envolvidos no desenvolvimento e concepção inicial dos módulo e do sistema SL.

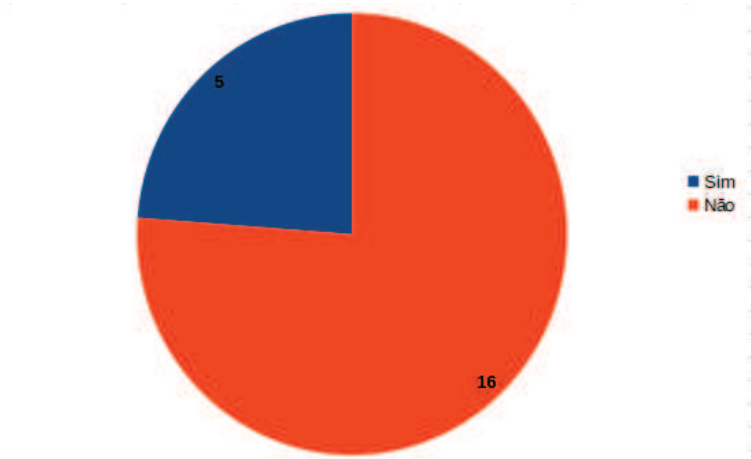

Figura 5. Participante envolvidos na concepção do sistema SL.

Como já descrito em seções acima, um dos problemas de sistemas legados está relacionado com a falta de experiência e conhecimento técnico e de negócio do envolvidos no desenvolvimento. Através desses dados citados é possível perceber que esse é um dos problemas que dificultam a manutenção e evolução do sistema legado SL. Este fato, também é destacado por Pressman (2011), que afirma que o aumento da dificuldade de manutenção está relacionada às modificações feitas nos sistemas por colaboradores que já não estão mais nas empresas, dificultando no entendimento do código e na documentação desenvolvida.

4.1.2. Problemas destacados do sistema SL

Nesta seção são apresentadas as respostas dos participantes em relação aos os problemas apresentados pela autora e qual o nível de concordância da existência desses problemas no sistema estudado.

Abaixo seguem as Figuras 6 e 7 que representam os gráficos com os resultados.

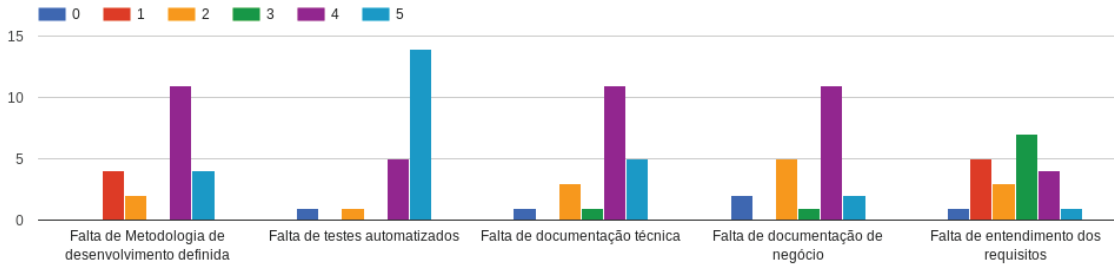

Figura 6. Gráfico com a escolha dos participantes (Parte 1)

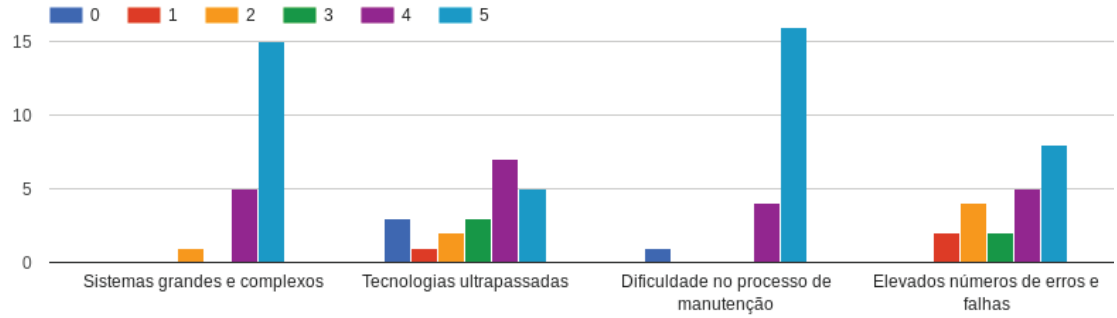

Figura 7. Gráfico com a escolha dos participantes (Parte 2)

Como mencionado na seção 3.4 (Análise de Dados), foram selecionados para fim de análise apenas os problemas que se destacaram em 4 ou 5 em grau de concordância entre dois terços (2/3) dos participantes. Então as características que serão analisadas e terão soluções propostas são:

- Falta de Metodologia de desenvolvimento definida;
- Falta de testes automatizados;
- Falta de documentação técnica;
- Sistemas grandes e complexos;
- Dificuldade no processo de manutenção.

Os itens apresentados acima reforçam a necessidade da empresa XYZ compreender a importância de investir em um processo de manutenção e evolução de seus sistemas, além de um planejamento inicial durante o desenvolvimento das primeiras versões de um novo produto, não só dirigido às partes técnicas como também as áreas contratuais e de negócio. Ou seja, a dificuldade no processo de manutenção apresentada pelo último item,

que recebeu 20 votos como 4 ou 5 dos 21 resultados (Figura 7), é efeito dos problemas dos itens anteriores ligado à falta de gravidade dada pela empresa nos primeiros momentos em que o sistema SL entrou em produção.

Ponto reforçado por Chaves (2004, p. 2) ao afirmar que "as organizações precisam definir estratégias para manter seus sistemas legados, com o objetivo de não serem surpreendidas e sofrerem os efeitos que tais sistemas podem trazer". Sommerville (2011) completa ao dizer que com a constante mudança, a estrutura do sistema tende a se degradar e deve ser investido tempo e dinheiro em melhorias para que o software não se torne cada vez mais difícil e onerosos.

Por fim, houve apenas uma resposta no campo em aberto disponibilizado para a opinião dos participantes. A resposta destacou outro problema do sistema legado SL que seria a falta de tempo para desenvolvimento de melhorias no sistema (manutenção Preventiva e Perfectiva). No entanto, por esse problema não ter sido apresentado na relação inicial da coleta de dados e não se enquadrar nos critérios de seleção apresentados, não será proposto uma solução associada a ele. Pela relevância desse problema destacado foi discutida um pouco mais sobre ele no tópico 'Sistemas grandes e complexos' (tópico 4.2.4).

4.2. Análise dos problemas selecionados e propostas de solução

4.2.1. Falta de Metodologia de desenvolvimento definida

Situação atual da empresa: Na empresa XYZ, durante disponibilização do sistema SL aos clientes resultou em uma grande carga de uso, consequentemente foram encontrados falhas e necessidades de novos requisitos. Neste momento, por falta de maturidade da empresa em relação a proporção do sistema, não houve um planejamento e nenhum responsável pelo gerenciamento das correções e novas atividades. Nesta época era utilizado uma metodologia com alguns conceitos do SCRUM, ou seja, era planejado o escopo de sprints de 15 dias com atividades estimadas e reuniões diárias, além de reunião de planejamento, revisão e retrospectiva.

No entanto, com as diversas interferências e mudanças no escopo, já fechado, houve diversas quebras de sprints e a não finalização das atividades combinadas, resultando em falhas nas entregas aos clientes e funcionalidades do sistema sem documentação ou planejamento.

Uma alternativa apresentada para melhorar o processo da equipe foi o método KANBAN, que não apresenta muita estruturação e se torna mais maleável pelo momento que o projeto passava. No entanto, o modo como estava sendo utilizado o KANBAN não apresentava marcos de entrega aos clientes, que ocasionou no surgimento de datas muito próximas e acabava por prejudicar o desenvolvimento e a equipe envolvida.

Atualmente com dois Gerentes de Projetos e um planejamento mais focado, foi dividido a equipe anterior em três, uma voltada a manutenção e as outras duas ao desenvolvimento de novas funcionalidade. As equipes de evolução voltaram a usar um processo baseado no SCRUM, no entanto a equipe de manutenção não apresenta nenhuma metodologia e recebe demanda de dois gerentes, o que acarreta em falta de priorização de problemas.

A equipe de manutenção é composta por 3 níveis de suporte: atendimento direto aos clientes, correções simples (sistema fora, reprocessamento de dados, falha resultante de um caso específico) e correções que precisam de implementação. Atualmente, cada nível é composto por apenas um colaborador e, além disso, esses membros da equipe de manutenção precisam ser pessoas mais experientes e com conhecimento do sistema.

Com isso seria necessário uma metodologia que atendesse a equipe de manutenção para que melhorasse o processo e a qualidade de desenvolvimento do sistema SL. Para a equipe de evolução do sistema, o SCRUM está atendendo as necessidades atuais.

Proposta: Como a empresa XYZ já utiliza alguns conceitos provenientes das metodologias ágeis, nesse estudo será apresentado os resultados positivos ao aplicar a metodologia *Extreme Programming* (XP), que pode ser implantada nas equipes envolvida com a manutenção e evolução do sistema SL, visto que umas das maiores dificuldades dessa implantação é a mudança cultural que ocorre quando há alteração de uma metodologia tradicional para um ágil, no caso, isso não irá ocorrer dado que os envolvidos já estão familiarizados com essa cultura. Mesmo podendo aplicar o XP para as 3 equipes, este estudo irá propor, inicialmente, implantar para a equipe de manutenção que apresenta maior dificuldade em seu gerenciamento

Primeiramente, *Extreme Programming* (XP) é "um estilo de desenvolvimento de software focado em excelentes técnicas de programação, comunicação clara e trabalho em equipe" [Beck 2012, p. 2]. Segundo (Sommerville, 2011, p. 45), algumas empresas que implantaram o XP não chegam a utilizar todas as práticas, que são:

Quadro 2: Práticas de Extreme Programming

Princípio ou prática	**Descrição**
Planejamento incremental	Os requisitos são gravados em cartões de história e as histórias que serão incluídas em um release são determinadas pelo tempo disponível e sua relativa prioridade. Os desenvolvedores dividem essas histórias em 'Tarefas'.
Pequenos releases	Em primeiro lugar, desenvolve-se um conjunto mínimo de funcionalidades útil, que fornece o valor do negócio. Releases do sistema são freqüentes e gradualmente adicionam funcionalidade ao primeiro release.
Projeto simples	Cada projeto é realizado para atender às necessidades atuais, e nada mais.
Desenvolvimento test-first	Um framework de testes iniciais automatizados é usado para esc rever os testes para uma nova funcionalidade antes que a funcionalidade em si seja implementada.
Refatoraçâo	Todos os desenvolvedores devem refatorar o código continuamente assim que encontrarem melhorias de código. Isso mantém o código simples e manutenível.
Programação em pares	Os desenvolvedores trabalham em pares, verificando o trabalho dos outros e prestando apoio para um bom trabalho sempre.

Propriedade coletiva	Os pares de desenvolvedores trabalham em todas as áreas do sistema, de modo que não se desenvolvam ilhas de expertise. Todos os conhecimentos e todos os desenvolvedores assumem responsabilidade por todo o código. Qualquer um pode mudar qualquer coisa.
Integração contínua	Assim que o trabalho em uma tarefa é concluído, ele é integrado ao sistema como um todo. Após essa integração, todos os testes de unidade do sistema devem passar.
Ritmo sustentável	Grandes quantidades de horas-extra não são consideradas aceitáveis, pois o resultado final, muitas vezes, é a redução da qualidade do código e da produtividade a médio prazo.
Cliente no local	Um representante do usuário final do sistema (o cliente) deve estar disponível todo o tempo à equipe de XP. Em um processo de Extreme Programming, o cliente é um membro da equipe de desenvolvimento e é responsável por levar a ela os requisitos de sistema para implementação.

No estudo de Coleman e McAnallen (2005), o XP foi apresentado como uma sugestão de metodologia para sistemas legados. O estudo implantou e analisou o XP em um sistema legado, além de apresentar autores que também implantaram o XP nas empresas que trabalhavam e tiveram bons resultados. No entanto, destacam a dificuldade de implantar as 12 práticas do XP e que na maioria dos estudos de sua implantação apresentados por eles houve aplicação parcial de suas práticas.

Diante disso, eles implantaram algumas práticas que a empresa ainda não executava. Após a implantação, perceberam que, no caso do sistema escolhido, descobriu-se que a abordagem ágil e de desenvolvimento test-first reduziu o tempo de entrega e gerou um código de maior qualidade com menos erros, consequentemente a empresa lucrou com essas mudanças no código. Porém, a programação em pares e o projeto simples não obtiveram bons resultados, além também da refatoração que ficou como trabalhos futuros.

Apesar de algumas práticas não serem aplicáveis no ambiente implantado, os autores chegaram à conclusão que é possível se beneficiar da metodologia XP, mesmo sem aplicá-la totalmente.

Já no estudo de Santos e Córdova (2017), é proposto uma união entre o Kanban e o XP. Mas, antes de implantar o XP, foram realizados algumas análises do contexto da empresa. Assim, não foram aplicadas as práticas de colaboração com o cliente e projeto simples. As demais surtiram efeitos positivos tanto na produtividade da equipe quanto no relacionamento entre os membro, “ficou evidente, que a equipe se tornou mais colaborativa e disposta a aderir a outras práticas a fim de acelerar as implementações e agregar mais qualidade aos produtos de software” .

No caso do sistema SL, a proposta seria aplicar as 12 práticas junto ao método Kanban, sem a obrigatoriedade de prosseguir com o uso de todas. Para isto, seria necessário o investimento em mais um ou dois colaboradores para agregar a equipe de manutenção. Sobre as práticas desenvolvimento test-first, refatoração e padronização do código, a equipe de manutenção estaria unida com a equipe de evolução, a fim de obter

um resultado que beneficie as três equipes e o sistema.

4.2.2. Falta de testes automatizados

Situação atual da empresa: O sistema SL, atualmentes, apresenta 0% (zero porcento) de cobertura de teste automatizado. Anteriormente, foram desenvolvidos testes unitários para algumas funcionalidades, no entanto, devido ao processo instável de desenvolvimento, como já exemplificado no tópico acima (4.2.1.1), novos testes não foram desenvolvidos e os anteriores pararam de ser utilizados. Com isso, possivelmente, os testes já desenvolvidos estão desatualizados, sendo necessário iniciar o desenvolvimento novamente.

Testes automatizados de sistema nunca foram desenvolvidos para o sistema SL, por este motivo, também deve ser iniciado desde o início. A falta desta automatização impacta diretamente na produtividades dos dois tester responsáveis pela qualidade desse sistema complexo, pois, após a análise dos impactos de uma alteração, se percebe como essa alteração repercute por todo o sistema, aumentando a carga dos testes de regressão. Algumas vezes, nem sendo previsto todos os possíveis impactos reais e causando mais demanda a manutenção.

É ciente que não é possível começar a automatizar tudo, principalmente pela complexidade e tamanho do código e, ainda mais sendo um sistema que está sendo desenvolvido a algum tempo e por diversas pessoas. Com isso, é necessário alguma estratégia para, pelo menos, ser iniciado o desenvolvimento de testes unitários e de sistema para algum módulo ou funcionalidade do sistema SL.

Proposta: Segundo Sartorelli (2007, p. 104), "a diversidade dos sistemas legados tornam possíveis diversas soluções para a criação e automação dos testes". Os 5 passos propostos por ele apresentam uma forma de organizar as fases da automação. O autor ainda afirma que esses 5 passos são aplicados para o início da implantação da automatização dos testes e para sua continuidade, seriam aplicados apenas 3 passos. Segue os passos abaixo:

1. Identificar pontos com a necessidade de automação e que são críticos ao sistema.
2. Planejar e desenvolver os casos de testes a serem utilizados e executados.
3. Definir as ferramentas para automação.
4. Implantar a automatização e incluir no processos de desenvolvimento.
5. Acompanhamento periódica dos testes e análise dos resultados para verificar a efetividade dos testes desenvolvidos.

Na segunda fase são executados os passos 2, 3 (se necessário) e passo 4. Essa fase é uma continuação da primeira e se trata de um ciclo que se repete até quando a automação se faz necessária. Neste estudo também, o autor explica com mais detalhes os conceitos de cada passo. Por fim, o ator destaca que essa proposta necessita de uma comprovação prática e que esses passos seriam apenas uma proposta de melhoria no processo de desenvolvimento.

No artigo de Chiele (2017) é proposto e realizado a refatoração e a automatização de testes em um sistema legado e, para isso, inicia realizando um estudo para identificar os

pontos do sistema que seriam aplicadas essas melhorias. Foi escolhido um destes pontos e foram aplicadas 5 técnicas de refatoração, "a medida que as refatorações reduziam o acoplamento, aumentavam a coesão e deixavam as funções mais simples, também foram desenvolvidos testes unitários, visando melhorar a cobertura de testes do sistema". Por fim, o autor realizou uma proposta de heurística para refatoração de código legado que será tratado no tópico 4.2.4.

Segundo Chiele (2017, p. 37), "foi possível observar uma melhora significativa na legibilidade, manutenibilidade e testabilidade do código, comprovando que estas técnicas de fato podem minimizar a dívida técnica do código no médio e longo prazo."

Com as informações apresentadas acima, os passos propostos para a implantação de testes automatizados nos módulos do sistema SL, independente de testes unitários ou de sistemas, serão implantados moderadamente para não causar um aumento de carga de trabalho pelas equipes envolvidas e nem um grande impacto nos prazos acordados com os clientes.

Esses passos serão diferentes entre a equipe de evolução e a de manutenção, visto que, a equipe de manutenção tem um ambiente mais dinâmico. Para a equipe de evolução serão propostos os seguintes passos:

1. Identificar os módulos mais crítico do sistema.
2. Nesses módulos, identificar quais pontos apresentam a necessidade de automação. Podendo apresentar pontos diferente de acordo com cada objetivo de teste realizado (unitário ou de sistemas).
3. Com os pontos divididos entre os responsáveis pelo desenvolvimento dos testes, ou seja, teste unitário serão realizados pelo desenvolvedores e o de sistemas pelos testers, será escolhido as ferramentas utilizada na automatização dos testes.
4. Realizar um nivelamento entre os colaboradores sobres as técnicas e ferramentas a serem usadas.
5. Antes do desenvolvimento da automação, deve ser acordado um cronograma entre a equipe, a gerência e o cliente para que todos fiquem cientes da melhoria do sistema e que nenhuma parte saia prejudicada.
6. Implantar a automatização e incluir no processo de cada equipe, ou seja, durante a reunião de planejamento deve ser estimadas as novas funcionalidades ou alterações das existentes com o desenvolvimento/atualização dos testes automatizados. Também deve ser inserido um ponto destacado pelo passo 2 ou, pelo menos, parte dele no planejamento da Sprint como atividade, para começar a aumentar a cobertura de testes.
7. Sempre antes de disponibilizar novas funcionalidades para o teste de aceitação do cliente, deve ser rodados os testes existentes no ambiente para verificar se não houve nenhum impacto durante o desenvolvimento.
8. Por fim, acompanhar e analisar os resultados obtidos pela equipe para que esses dados sejam expostas a Gerência.

Para a equipe de manutenção são propostos menos passos por se tratar de um ambiente que apresente mais urgência nas entregas e não sendo necessário o desenvolvimento de testes para o código legado, apenas a manutenção dos já existente. Os passos são:

1. Será escolhido a ferramenta utilizada na automatização dos testes junto às equipes de evolução, para que as tecnologias trabalhadas sejam as mesmas.
2. Também junto às equipes de evolução, será realizado um nivelamento do conhecimento da tecnologia utilizada.
3. Deixar as partes interessadas (equipe, gerência e cliente) cientes da nova prática adotada para o sistema.
4. Durante o desenvolvimento das correções, deve manter atualizado os testes existentes ou desenvolver novos para os casos que a correção leva a nova implementação.

Percebe-se que a implantação da automação dos testes do sistema SL pode demorar para gerar resultados, principalmente por ser inserida aos poucos nos processos de desenvolvimento das equipes. O importante é deixar todos os envolvidos com conhecimento dos possíveis benefícios que o aumento da cobertura de testes pode causar para o sistema, principalmente um sistema grande e complexo como o SL.

4.2.3. Falta de documentação técnica

Situação atual da empresa: Anteriormente, o sistema SL, por ser um projeto muito amplo e complexo, foi dividido em diversas equipes, o que ocasionou em falta ou inexistência da documentação técnica e padronização das informações. Agora, que se encontra com duas equipes responsáveis, os envolvidos apresentam dificuldades em entender o sistema de maneira geral e, assim, poder desenvolver novas funcionalidades ou alterá-las da melhor maneira possível, sem impactar as existentes.

Outro problema está na manutenção e atualização das documentações, pois não há muito tempo, nem conhecimento dos envolvidos para a evolução dessas informações. Através desse problema conhecido, há impactos em novos membros que não tem muito conhecimento do sistema e apresentam dificuldades em executar tarefas novas e simples no sistema, sobrecarregando ainda mais os experientes.

Um estratégia adotada pelas equipes está sendo através do desenvolvimento de documentos a partir das novas funcionalidades, deixando a Gerência ciente do tempo investido na criação desses novos documentos e de sua padronização. Mas ainda é necessário investir tempo em recuperar as informações técnicas perdidas durante as trocas de equipe e de membros até a estabilização atual.

Proposta: Há muita dificuldade em recuperar informações de sistemas legados, isto ocorre devido a falta de documentação ou, para as existentes, por estar desatualizada. Para recuperar essas informações aplica-se um processo de engenharia reversa [Pressman 1995 apud Chaves 2004]. "A engenharia reversa deve produzir, preferencialmente de forma automática, documentos que ajudem a aumentar o conhecimento geral de sistemas de software, facilitando o reuso, manutenção, teste e controle de qualidade de software" [Braga 1998, p. 17].

Na pesquisa de Junior et al. (2005, p. 7), são apresentada dois tipos de engenharia reversa:

- No primeiro caso, o código-fonte já está disponível, mas os aspectos mais globais, talvez documentação escassa ou não válida, têm que ser descobertos.
- No segundo caso o código-fonte do software não está disponível, e todos os esforços para descobrir uma possível fonte do código para o software são considerados como engenharia reversa.

Para o sistema legado SL, o tipo de engenharia reversa que poderá ser utilizada é o primeiro caso, pois os envolvidos apresentam fácil acesso ao código fonte dos módulos do sistema. Os autores, também, disponibilizam técnicas de engenharia reversa para o caso aplicável no sistema estudado. É apresentado dois passos e informações sobre eles, segue abaixo de forma resumida:

- Extração das Informações: primeiramente deve ser coletados as informações do sistemas. Elas podem ser extraídas do código-fonte, da execução dele, de dados (banco de dados), da documentação (se houver), ou outras fontes. Possui três fases: análise, projeto e implementação.
- Tratamento dos Fatos: abstrair informações de mais alto nível de abstração, eliminando os detalhes, mas o problema é decidir o que é importante ou não.

Caldana (2003), em seu estudo, elabora um proposta baseada no método FUSION, o FUSION/RE, que "tem como objetivo a recuperação do projeto de sistemas legados".

O método FUSION/RE apresenta as fases, informadas pelo autor:

- Revitalizar a Arquitetura do Sistema: recuperar informações do sistema a fim que ele seja entendido, pode recuperar tanto de documentação, se houver, como de código fonte.
- Recuperar o Modelo de Análise da Solução Atual: desenvolver uma modelagem apenas das características atuais do sistema
- Abstrair o Modelo de Análise do Sistema: abstrai-se os conceitos que realmente fazem parte do sistema.
- Mapear o Modelo de Análise do Sistema para o Modelo de Análise do Sistema Atual: importante parte para futuros desenvolvimentos ou manutenção.

Ainda em seu estudo, Caldana (2003) aplica o método em um sistema comercial e analisa os resultados e documentações geradas. Para o autor, ao término foi gerado uma documentação completa e detalhada sobre o sistema analisado. No entanto, "um critério não compreendido pelo método FUSION/RE é a validação entre a documentação gerada e o sistema atual", para o autor, essa validação é importante para verificar se todas as funcionalidades foram analisadas e recuperadas. Ao final, é proposto um novo passo ao método que apresenta o objetivo de validar a documentação gerada.

No sistema SL, seria interessante o método proposto por Caldana (2003), pois é um método voltado a sistemas legados. Como, onde e quando aplicá-lo, ficaria como trabalhos futuros, já que, primeiramente, deveria ser planejado como realizar a engenharia reversa nos módulos e sistema SL, junto a equipe de desenvolvimento e a Gerência. Portanto, para este tópico foi exposto apenas a proposta da utilização do método FUSION/RE com a nova fase de avaliação dos documentos gerados, proposto por Caldana (2003), a fim de melhorar a documentação técnica do sistema SL.

4.2.4. Sistemas grandes e complexos

Situação atual da empresa: O sistema SL é composto por módulos de tamanhos e complexidades diferentes, a princípio, a fim de diminuir a complexidade, foi dividido entre equipes diferentes que apresentavam outros projetos e não estavam integradas entre si. A quebra do sistema entre equipes, que não apresentavam foco 100% no sistema SL, resultou em alguns problemas já apresentados, como falta de padronização e documentação, mas também no conhecimento aprofundado do fluxo total entre seus módulos. Uma vez que os módulos não devem ser considerados separadamente, dado que eles se integram e a ação em um impacta, quase sempre, diretamente ou indiretamente nos outros. Consequentemente, aumenta a complexidade de manutenção e cada alteração realizada em um dos módulos é obscura com relação aos impactos no restante do sistema.

Por essa obscuridade e, inicialmente, falta de planejamento e relevância dada a manutenção, o sistema foi se tornando ainda maior (código duplicado, falta de comentários, falta de planejamento da arquitetura, falta de padronização de desenvolvimento, falta de padrão das informações de banco de dados).

Utilizando a ISO/IEC-14764 (1999) para exemplificar os tipos de manutenção utilizadas na empresa XYZ, pode dizer que a equipe de manutenção está realizando, exclusivamente, a manutenção corretiva e, muita das vezes, nem há tempo para ser realizados a implementação da correção do problema e, sim, somente a correção do caso encontrado pelo cliente/usuário. Ou seja, é identificado que o caso encontrado pelo usuário/cliente ocorre devido a uma falha de desenvolvimento, no entanto, a manutenção não tem tempo para corrigir essa falha, realizando a correção apenas do caso ou da parte da falha que resulta neste caso.

Como mencionado por um participante do questionário ao responder a última pergunta, não há tempo dedicado na realização de manutenções perfectivas e preventivas. Aliás, é necessário indicar que o sistema SL é complexo e extenso desde sua concepção, pois apresenta diversas regras e é disponível para diferentes tipos de usuário.

Após a apresentação desses fatores, não é possível reverter o que já está feito, mas algumas decisões de manutenção e evolução a fim de manter ou até diminuir sua complexidade podem ser tomadas.

Proposta: No artigo de Chiele (2017), como mencionado no tópico 4.2.2, foi proposto e realizado a refatoração e a automatização de testes em um sistema legado. Após identificar os pontos do sistema que seriam aplicadas essas melhorias, foram aplicadas 5 técnicas de refatoração (extrair método/função, mover método/função, decompor condicional, condensar hierarquias, renomear nomes de métodos/funções/módulos/variáveis).

"Refatoração é uma alteração feita na estrutura interna do software para torná-lo mais fácil de ser entendido e menos custoso de ser modificado sem alterar seu comportamento observável." [Fowler 2004, p. 52]. Barrozo et al. (2012) alerta que a refatoração não pode ser aplicada em todo código, pois é necessário ser analisado cada parte separadamente e para verificar a possibilidade e necessidade de refatorá-lo, uma vez que "aplicar a refatoração de forma equivocada poderá trazer problemas ao software".

Ainda no estudo, Chiele (2017) propõe uma heurística de refatoração, devido o

problema encontrado na linguagem de programação utilizada para fornecer métricas de software para guiar na escolha dos pontos de refatoração e testes. As heurísticas são apresentadas resumidamente através da Figura 8 abaixo, disponibilizada pelo autor.

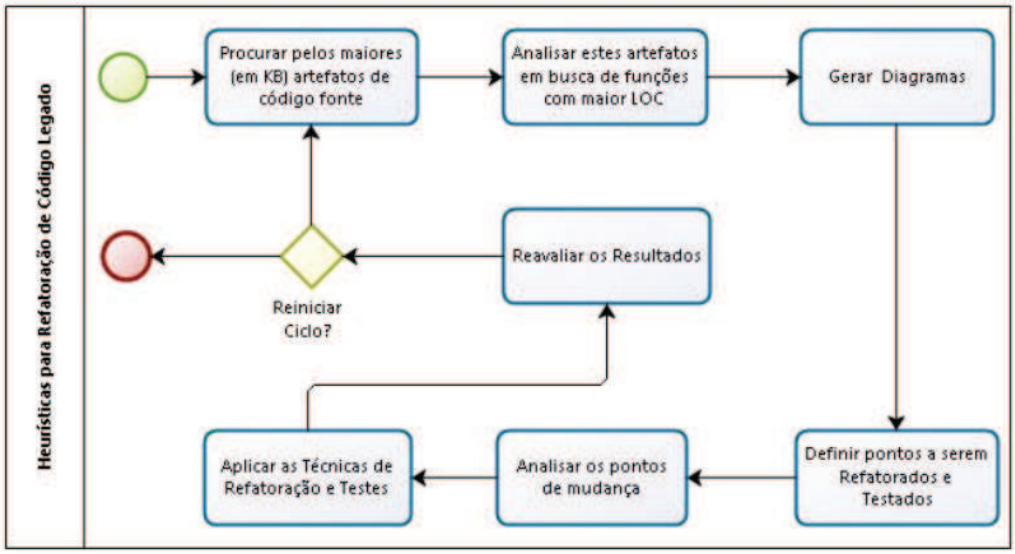

Figura 8. Resumo das heurísticas utilizadas no processo de refatoração

Por fim, Chiele (2017) afirma que mesmo a refatoração sendo realizada em apenas um módulo do sistema estudado, "um projeto de refatoração bem definido e planejado, pode levar este sistema a ter uma vida útil por muito mais tempo", também podendo diminuir o esforço de compreensão e manutenção do sistema.

Baseado no que Barrozo et al. (2012) alertam e Chiele (2017) afirma a cima, será proposto a refatoração em módulos do sistema SL e de forma gradativa. Podendo ser unido aos pontos selecionados e que precisam ser testados automaticamente, proposto no tópico 4.2.2, para verificar se esses pontos escolhidos também precisam ser refatorados. Deste modo, as alterações e melhorias serão realizadas e as equipes continuariam desenvolvendo atividades e entregas aos clientes.

Sobre as técnicas para identificar e as para refatorar, assim como as ferramentas de testes automatizados e de engenharia reversa, será necessário reuniões com os envolvidos, a fim de verificar com desenvolvedores mais experientes e com mais conhecimento do sistema como prosseguir com essas melhorias, a fim de analisar a estrutura/arquitetura do sistema com as soluções propostas.

4.2.5. Dificuldade no processo de manutenção

Situação atual da empresa: O sistema estudado apresenta 5 módulos e customizações para diversos clientes, que resulta na manutenção e evolução de, no mínimo 20 módulos, todos já implantados e em uso pelos usuários finais. As demandas para as correções de falhas e desenvolvimento de novas funcionalidades chegam a todo momento, no entanto elas apresentam grau de urgência diferentes para cada equipe.

O ambiente da equipe de manutenção se torna mais conturbado por trabalhar com

demandas que o próprio usuário encontrou e que pode ter trago alguma insatisfação por não conseguir prosseguir com o que pretendia. Atualmente, na manutenção, é utilizada uma ferramenta de gerenciamento de erros com a qual o cliente relata o erro reportado. Nesta ferramenta que é realizado a distribuição entre os níveis de suporte.

No entanto, não é realizado nenhum fluxo de trabalho e nem de priorização entre as demandas de cada cliente, tornando o método de manutenção informal para o grau de complexidade do sistema. Além, de alguns defeitos não serem reportados na ferramenta mas para o Gerente de Projeto ou outro profissional não envolvido diretamente com a equipe. Isso, se torna um problema pois são passadas informações incompletas, tornando o ambiente de manutenção ainda mais conturbado.

Um processo de manutenção mais elaborado pode resultar na redução dos custos de manter o sistema SL e aumentar a produção dos membros da equipe, melhorando a satisfação do cliente.

Proposta: Segundo Polo et al. [2003, apud Paduelli 2007], aplicar técnicas, ferramentas, modelos de processo à manutenção de software semelhante ao desenvolvimento pode não ser uma boa decisão, já que esses momentos apresentam características diferentes. Paduelli (2007) ainda continua citando Polo et al. (2003) ao apresentar algumas razões que justificam essa diferença.

- O esforço despendido em tarefas de desenvolvimento e manutenção não é igual,constituindo uma prática comum nas organizações concentrar a maior parte do esforço humano disponível em codificação e tarefas de teste durante o desenvolvimento.
- Outro fator de diferença está ligado ao fato de que a manutenção tem mais similaridade com um serviço do que com desenvolvimento, o que acaba implicando na necessidade de não aplicar as mesmas estratégias de desenvolvimento em manutenção.

A ISO/IEC-14764 (1999) fornece etapas como exemplos de passos necessários para executar e implementar as atividades de manutenção, que estão presentes dentro da categoria de processos fundamentais da ISO/IEC-12207 (2008). As atividades que compõem o processo de manutenção são:

- Implementação de Processos;
- Análise de Problema e Modificação;
- Implementação de Modificação;
- Revisão de Manutenção/Aceitação;
- Migração;
- Aposentadoria.

Com as informações apresentadas, com o conhecimento do ambiente de manutenção da empresa XYZ e baseado no fluxo de evolução disponibilizado por Sommerville (2011, p. 167), este estudo irá propor um processo de manutenção para o sistema SL. É de conhecimento da autora, que Sommeville considera o processo de evolução de sistema como o processo de desenvolvimento e manutenção, no entanto, para o ambiente que se encontra a manutenção e desenvolvimento do sistema legado SL, o processo destes

dois ambientes serão tratados de forma diferentes e neste estudo será proposto apenas um processo para a manutenção.

Na Figura 9, a autora exemplifica o fluxo do processo proposto.

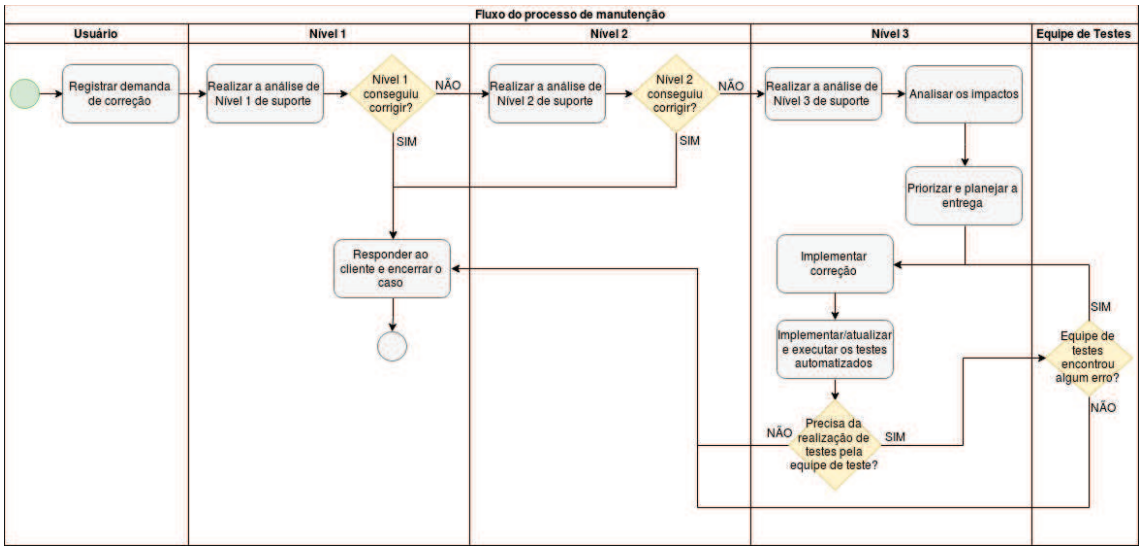

Figura 9. Proposta de processo da manutenção.

O fluxo de comunicação entre Nível 1 e 2 do Suporte continuará o mesmo, no entanto, todo o processo do nível 3 seria incrementado a fim de atender as melhorias propostas neste trabalho. Antes do nível 3 começar a desenvolver as correções, deve se fazer uma análise do impacto dessas alterações. A partir dessa análise, deve haver uma priorização de acordo com a necessidade e urgência do cliente e é só após esse momento que é iniciado a implementação da correção, consequentemente, deve ser realizado o teste automatizado. Por fim, verifica se há necessidade de interação com os analistas de testes e, caso a correção não cause mais nenhum impacto, esta é implantada em todos os ambientes do sistema, para mantê-los semelhantes. Assim o caso é corrigido e encerrado.

4.3. Avaliação das melhorias propostas

Após as propostas desenvolvidas pela autora, foram apresentadas aos participantes do questionário os problemas selecionados e suas soluções. A equipe se apresentou bastante receptiva as melhorias propostas e motivadas a implantá-las. Esta apresentação gerou algumas informações válidas e que serão apresentada a seguir:

- A proposta de aplicar as práticas da metodologia ágil XP foi muito bem aceita e gerou várias discussões e planejamentos. No entanto, a parte da proposta sobre a contratação de novos colaboradores para a equipe de manutenção deveria ser analisada com a Coordenadoria a fim de iniciar os processo de contratação de novos membros, isso demoraria um tempo mais para ocorrer. Ficou o questionamento se seria possível a aplicação do XP antes da contratação de novos membros, após uma conversa, ficou acordado que seria implantado todas as práticas do XP a primeiro momentos, antes dessas novas contratações. Se o XP resultasse em melhorias, o envolvidos ficariam interessados em aplicá-lo também nas equipes de evolução.

- Sobre a parte da documentação técnica, a proposta de engenharia reversa não foi bem aceita a primeiro momento, devido a prática atual adotada pelas equipes, a qual ao se deparar com a necessidade de uma informação técnica, após obtê-la, esta é documentada no padrão atual do sistema. Devido a este motivo e as necessidades prioritárias de melhoria no sistema, a prática proposta poderá ser aplicada futuramente.
- Sobre a automatização dos testes do sistema, os envolvidos acharam de extrema importância o ponto ao qual é dito que deve ser analisado quais pontos são críticos e devem ser automatizados. No entanto, ressaltaram que para escolher esses pontos é necessário disponibilizar um desenvolvedor experiente, o que resulta em uma queda de entrega aos clientes.
- A mesma conclusão obtida no aumento de cobertura dos testes automatizados foi a relacionada a refatoração do código a fim de diminuir a complexidade o sistema.
- Por fim, foi muito bem aceito o fluxo do processo de manutenção proposto. Todos se mostraram bastante positivos em implantá-lo na equipe de manutenção. A única ressalva foi sobre a dificuldade de priorização da resolução das atividades, pois o sistema apresenta diversos clientes e é necessário um planejamento mais amplo de entrega.

Os envolvidos se mostraram bastante motivados em terem soluções aos problemas recorrentes no sistema SL, além de poderem discutir esses problemas. Com as propostas analisadas, será desenvolvido uma apresentação para a coordenadoria para que todos estejam cientes das novas práticas adotadas pela equipe que desenvolve e mantém o sistema SL.

5. Considerações Finais

A identificação das estratégias mais apropriadas para a evolução e manutenção de sistemas legados ainda é um desafio para muitas empresas. Como identificado na literatura utilizada e no sistema estudado, sistemas legados apresentam vários problemas que impactam diferentes etapas de sua manutenção, ou seja, desde a parte do conhecimento do sistema, através de documentos, até a parte de código, arquitetura e testes.

Essa dificuldade em manter e evoluir os sistemas deriva da complexidade e da falta de experiência das empresas no planejamento desses sistemas de vida longa e essenciais a elas. É perceptível que essas empresas não estão cientes da importância da manutenção de seus sistemas, o que acarreta em sistemas com grande custo para serem mantidos. Devido a essas características, o estudo propôs identificar os problemas mais recorrentes do sistema legado SL da empresa XYZ, através da observação da autora e de um questionário disponível aos envolvidos. Diante da identificação desses problemas, o estudo ainda apresenta o objetivo de propor possíveis melhorias através da literatura apresentada.

Este estudo teve um desafio em encontrar conteúdos voltados a melhoria na manutenção de sistemas legados, a literatura atual está muito voltada a processos de maneira geral, mas é de conhecimento que o processo de um sistema legado apresenta suas particularidades. Dos estudos encontrados, que auxiliaram neste trabalho, os conceitos e focos eram em algumas áreas específicas e principalmente em refatoração de código, foi encontrada pouca conteúdo relacionada a uma melhoria de processo e boas práticas de manutenção para sistemas que já são complexos e legados.

Para realizar este estudo de caso foram priorizados os 5 principais problemas do sistema SL, por meio de um questionário aplicado para 21 colaboradores da empresa XYZ envolvidos com o desenvolvimento e manutenção desse sistema. Diante disso, foram apresentados a situação atual da empresa em relação ao problema e soluções encontradas na literatura.

Além de destacar problemas identificados pelos envolvidos na manutenção do sistema SL e identificar possíveis resoluções, outra contribuição importante, está relacionada a realização de uma síntese com os estudos baseados nessa área, que a cada ano ganha maior relevância. Lembrando que após a entrega de um determinado sistema não resulta em sua conclusão e, sim, em um longo caminho de evolução e manutenção para atender as expectativas dos clientes.

A partir do estudo de caso realizado, seria importante sugerir e aplicar essas propostas de melhorias no processo de desenvolvimento e manutenção do sistema SL e, durante a implantação, coletar e analisar os resultados obtidos para, por fim, validar a relevância e consistência dessas sugestões. Também poderia ser realizado uma pesquisa sobre ferramentas que melhor se adequam às propostas e ao sistema.

Além de desenvolver e aplicar métricas para detectar possíveis problemas de desempenho e de inadequações na aplicação dessas melhorias, pois medições são importante para um processo de software já que fornecem dados que permitem conhecer melhor o desempenho do processo de melhorias. [Rocha et al. 2012]

Referências

Barrozo, G. C., Vinhas, H. M., e Reis, J. C. (2012). Refatoração: Aperfeiçoando um código existente. Unifenas.

Beck, K. (2012). Extreme programming explained: Embrace change.

Braga, R. T. V. (1998). Padrões de software a partir da engenharia reversa de sistemas legados.

Caldana, C. G. (2003). Infusion: Uma experiência de engenharia reversa orientada a objetos para sistemas legados.

Chaves, L. L. (2004). Sistemas legados e a aplicação de processos de reengenharia de software. 1ª International Conference on Information Systems and Technology Management - CONTECSI.

Chiele, C. (2017). Estudo sobre práticas ágeis de refatoração e testes automatizados no desenvolvimento de software para melhoria da qualidade de sistemas legados. Universidade do Vale do Rio dos Sinos (Unisinos), São Leopoldo, RS.

Coleman, G. e McAnallen, M. (2005). Tailoring extreme programming for legacy systems: lessons learned. EuroSPI Conference, Budapest, Hungary.

Fowler, M. (2004). *Refatoração: Aperfeiçoando o projeto de código existente*. Porto Alegre: Bookman.

Gangadharan, G. R., Kuiper, E. J., Janssen, M., e Luttighuis, P. O. (2013). It innovation squeeze: Propositions and a methodology for deciding to continue or decommission legacy systems. International Federation for Information Processing.

Gerhardt, T. E. e Silveira, D. T. (2009). *Métodos de Pesquisa*. Editora UFRGS, 1ª edição edition.

Gil, A. C. (2008). *Métodos e Técnicas de Pesquisa Social*. Editora Atlas S.A., 6ª edição edition.

ISO/IEC-12207 (2008). *Systems and software engineering – Software life cycle processes*. International Organization for Standardization.

ISO/IEC-14764 (1999). *Information technology – Software Maintenance*. International Organization for Standardization.

Junior, A. J. S. d. C., Souza, D. A., Moutinho, D. d. S., e Lohnefink, F. P. (2005). Engenharia reversa. UFF – Universidade Federal Fluminense.

MPS.BR (2011). *Guia de Implementação – Parte 9: Implementação do MR - MPS em organizações do tipo Fábrica de Software*. Melhoria de Processo do Software Brasileiro.

Paduelli, M. M. (2007). Manutenção de software: problemas típicos e diretrizes para uma disciplina específica.

Piekarski, A. E. e Quináia, M. A. (2000). Reengenharia de software: o que, por quê e como. Departamento de Informática - UNICENTRO. Revista Ciências Exatas e Naturais, Guarapuava, PR.

Pinto, H. e Braga, J. (2004). Sistemas legados e as novas tecnologias: técnicas de integração e estudo de caso.

Pressman, R. (2011). *Engenharia de Software*. AMGH.

Rocha, A. R. C., Souza, G. d. S., e Barcellos, M. P. (2012). *Medição de Software e Controle Estátistico de Processos*.

Santos, D. B. e Córdova, P. R. (2017). CombinaÇÃo de mÉtodos Ágeis no processo de desenvolvimento de software: Um estudo de caso.

Sartorelli, R. C. (2007). Proposta de processo para automação de testes em sistemas legados.

Sommerville, I. (2011). *Engenharia de Software*. Pearson Addison - Wesley, 9ª edition.

Warren, I. (1999). The renaissance of legacy systems: Method support for software-system evolution.

APÊNDICE A - QUESTIONÁRIO

Este questionário irá contribuir para uma pesquisa relacionada à melhoria de processo do sistema legado e do produto desenvolvido pela empresa. Este questionário é anônimo, todos os dados obtidos serão utilizados apenas para fins de estudo, nenhuma informação pessoal será exposta.

Informações gerais

Favor marcar com um **X** somente em uma única resposta que melhor se apresente para você.

1. Faixa de idade:
( ) Até 25 anos ( ) De 25 a 35 anos ( ) De 35 a 45 anos ( ) De 45 a 60 anos ( )Acima de 60 anos

2. Tempo em que você está na empresa:
( ) 1 ano ou menos ( ) mais de 1 a 3 anos ( ) mais de 3 a 5 anos ( ) mais de 5 a 10 anos
( ) mais de 10 anos

3. Seu papel na empresa:
( ) Gerente de Projetos ( ) Analista de Requisitos ( ) Desenvolvedor ( ) Analista de Testes
( ) Analista de Banco de Dados

4. Tempo em que trabalhou/trabalha nos módulos ou no sistema como um todo:
( ) 1 ano ou menos ( ) mais de 1 a 3 anos ( ) mais de 3 a 5 anos ( ) mais de 5 a 10 anos
( ) mais de 10 anos

5. Esteve desde o início da concepção do sistema ou de seus módulos?
( ) Sim ( ) Não

Informações do sistema

As informações exibidas são características comuns encontradas em sistemas legados, favor responder considerando sua percepção ou opinião quanto a presença dessas características no sistema legado da empresa. A opção '5' seria a confirmação da existência dessa característica no sistema em questão.

0- Sem condições de opinar 1 - Discordo totalmente 2 - Discordo parcialmente 3 - Indiferente 4 - Concordo Parcialmente 5 - Concordo totalmente							
01	Falta de Metodologia de desenvolvimento definida	0	1	2	3	4	5
02	Falta de testes automatizados	0	1	2	3	4	5

03	Falta de documentação técnica	0	1	2	3	4	5
04	Falta de documentação de negócio	0	1	2	3	4	5
05	Falta de entendimento dos requisitos	0	1	2	3	4	5
06	Sistemas grandes e complexos	0	1	2	3	4	5
07	Tecnologias ultrapassadas	0	1	2	3	4	5
08	Dificuldade no processo de manutenção	0	1	2	3	4	5
09	Elevados números de erros e falhas	0	1	2	3	4	5

Caso ache que o sistema apresenta algum outro problema, o campo abaixo está aberto a opiniões.

_____
_____
_____
_____
_____
_____
_____
_____
_____