



UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO

CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA

ESCOLA DE INFORMÁTICA APLICADA

GAPRO: Um Sistema para Recomendação de Prática Ágeis

Daniel Martiniano Ferreira

Victor Doria Cardoso da Rocha

Orientador

Gleison dos Santos Souza

RIO DE JANEIRO, RJ – BRASIL

JULHO DE 2016

GAPRO: Um Sistema para Recomendação de Prática Ágeis

Daniel Martiniano Ferreira
Victor Doria Cardoso da Rocha

Projeto de conclusão de curso apresentado à
Escola de Informática Aplicada da Universidade
Federal do Estado do Rio de Janeiro (UNIRIO),
como requisito parcial para obtenção do título de
Bacharel em Sistemas de Informação.

Aprovada por:

Prof. Gleison dos Santos Souza, D.Sc. (UNIRIO)

Leonardo Azevedo, D.Sc. (UNIRIO)

Cristina Teles Cerdeiral, D.Sc. (UNIRIO)

RIO DE JANEIRO, RJ – BRASIL.
JULHO DE 2016

Agradecimentos

Dedico este projeto a meu pais, Paulo Roberto e Edmara, por toda a força e luta sempre e por me mostrar o mundo da melhor forma possível. Obrigado por serem meu exemplo, obrigado por todo o carinho, atenção e dedicação e suporte, para minha educação e formação de vida

A minha irmã, Ana Lidia, obrigado pela confiança, conselhos e pelo apoio. Saiba que pode contar comigo para sempre.

A minha companheira Angélica, pelo carinho, amor, lealdade e dedicação. Obrigado por ser minha companheira da vida e meu amor. Obrigado pela paciência nos momentos de estudo e trabalho e pelo carinho e consideração sempre.

Ao nosso orientador Gleison Santos, que sempre nos motivou a melhorar e buscar mais conhecimento, além da dedicação e paciência necessárias e essenciais para a evolução deste trabalho. Suas correções, incentivos e discussões sempre permitiram novos avanços, mesmo nas dificuldades encontradas ao longo do caminho.

Agradeço a banca responsável pela avaliação deste trabalho, os professores Cristina e Leonardo, e com mesmo apreço e estimo, agradecemos a todos professores, equipe técnico administrativa e colaboradores do BSI, que nos acompanharam durante a graduação, sempre atentos a nossas tarefas e crescimento dentro da universidade.

Agradecemos a Rafaela Fonseca pelo tempo despendido, com dicas preciosas e pela consideração e apoio.

Gostaríamos de agradecer a esta universidade, seu corpo docente, direção e administração, esta conquista concretiza-se com a contribuição de cada uma delas.

Daniel Martiniano Ferreira

Agradecimentos

A Deus, pois sem Ele nada seria possível.

Aos meus pais, por todo o amor incondicional e apoio que sempre me deram. Obrigado por todos os conselhos e lições, e obrigado por estarem sempre presentes tanto nos momentos fáceis como nos desafiadores.

Agradeço às minhas irmãs e aos meus familiares, por toda a motivação e pela confiança depositada em mim.

Aos meus amigos, pelos pensamentos positivos e certeza do meu sucesso.

Ao nosso orientador Gleison Santos, por todo o incentivo, paciência e suporte. A sua colaboração nos motivou a adquirir novos conhecimentos e a sempre buscar fazer o melhor.

Agradecemos a Rafaela Fonseca pelas dicas e por compartilhar a sua experiência conosco.

A banca responsável pela avaliação deste trabalho, os Professores Cristina e Leonardo, e a todos os professores do BSI, pelo compartilhamento dos seus conhecimentos e colaboração para a minha formação profissional.

Aos colaboradores da UNIRIO.

Obrigado novamente a todos os mencionados. Todos contribuíram para que essa conquista e a conclusão dessa etapa e o início de uma nova fossem possíveis.

Victor Doria Cardoso da Rocha

RESUMO

Mudanças frequentes nas necessidades das organizações exigem que sistemas de informação sejam cada vez mais adaptáveis e dinâmicos. Projetos de software que não são adaptados rapidamente e de forma eficaz costumam ser afetados negativamente, a ponto de ter seu sucesso comprometido. Na busca para aumentar o percentual de sucesso de projetos de software, muitos gerentes de projeto têm feito uso de métodos e práticas ágeis em seus projetos. No entanto, a quantidade de métodos ágeis existentes e a inexperiência de gerentes de projetos no uso de métodos ágeis podem dificultar a escolha dos métodos e práticas mais adequados para um determinado projeto.

O objetivo desta monografia é criar um Sistema de Apoio a Decisão para auxiliar gerentes de projeto a escolher as práticas ágeis mais adequadas para um determinado projeto. Com base em fatores críticos de sucesso em projetos de software e problemas encontrados pelo gerente durante a execução de seus próprios projetos, o sistema GAPRO recomenda práticas ágeis que podem mitigar os problemas encontrados, contribuindo para o sucesso do projeto.

Uma avaliação foi realizada para determinar se o sistema efetivamente auxiliaria um gerente de projetos na escolha de práticas ágeis. Como resultado, houve evidências que gerentes de projetos iniciantes com conhecimentos básicos sobre metodologias ágeis podem ser auxiliados e tirar vantagem das recomendações do GAPRO.

Palavras-chave: Gerenciamento de projetos, Métodos Ágeis, Práticas Ágeis, Sistemas de apoio à decisão

ABSTRACT

Frequent changes in the needs of organizations require that information systems are adaptable and dynamic. Software projects that are not adapted quickly and effectively might be negatively affected and had their success compromised. While seeking to increase the success rate of software projects, many project managers make use of agile methods and practices. However, the quantity of agile methodologies and the inexperience of project managers in the use of agile methods can make it difficult to choose the most appropriate methods and practices for a particular project.

Our goal is to create a decision support system to help project managers to choose the most suitable agile practices for a particular project. Based on critical success factors in software projects and problems encountered by the manager during the execution of their own projects, GAPRO system recommends agile practices that can mitigate the problems encountered, contributing to the project's success.

We conducted an evaluation to determine whether the system effectively would help a project manager in choosing agile practices. As a result, we had evidence that novice project managers with basic knowledge of agile methodologies can be supported and take advantage of GAPRO recommendations.

Keywords: Projects Management, Knowledge Based Systems, Agile Methodologies

SUMÁRIO

CAPÍTULO 1 - INTRODUÇÃO	1
1.1 MOTIVAÇÃO	1
1.2 OBJETIVO.....	2
1.3 ESTRUTURA DO TRABALHO	2
CAPÍTULO 2 - GESTÃO DE PROJETOS	4
2.1 DEFINIÇÃO DE PROJETO	4
2.2 CICLO DE VIDA DE UM PROJETO	7
2.3 GERENCIAMENTO DE PROJETOS	8
2.4 FATORES CRÍTICOS DE SUCESSO EM PROJETOS DE SOFTWARE	10
2.5 CONSIDERAÇÕES FINAIS.....	11
CAPÍTULO 3 - MÉTODOS E PRÁTICAS ÁGEIS.....	14
3.1 DESENVOLVIMENTO ÁGIL DE SOFTWARE.....	14
3.2 DIFERENÇAS ENTRE MÉTODOS ÁGEIS E MÉTODOS TRADICIONAIS	15
3.3 SCRUM	17
3.4 EXTREME PROGRAMMING (XP).....	18
3.5 KANBAN.....	19
3.6 LEAN.....	20
3.7 CONSIDERAÇÕES FINAIS.....	21
CAPÍTULO 4 - SISTEMAS DE APOIO À DECISÃO.....	26
4.1 HISTÓRICO	26
4.2 CATEGORIAS DE UM SISTEMA DE APOIO À DECISÃO	28
4.3 CONCEITOS.....	29
4.4 CARACTERÍSTICAS DO SAD	30
4.5 COMPONENTES DO SAD	32
4.6 VANTAGENS E DESVANTAGENS	33
4.6.1 Vantagens	34
4.6.2 Desvantagens.....	35
4.7 CONSIDERAÇÕES FINAIS.....	36
CAPÍTULO 5 - DESENVOLVIMENTO DO SISTEMA	38
5.1 FUNCIONALIDADES DO GAPRO	38
5.2 ARQUITETURA DO SISTEMA	40

5.2.1 Padrão MVC	41
5.2.2 Ferramentas utilizadas.....	42
5.2.2.1 PHP	42
5.2.2.2 CodeIgniter.....	42
5.2.2.3 MySQL e phpMyAdmin	43
5.2.2.4 Lista de ferramentas	43
5.3 MODELAGEM DO SISTEMA	43
5.3.1 Diagrama de Entidade-Relacionamento.....	Erro! Indicador não definido.
5.3.2 Diagrama de Fluxo de Dados	45
5.3.3 Diagrama de Sequência.....	47
5.4 TELAS DO SISTEMA	52
5.4.1 Administrador	52
5.4.2 Gerente	58
5.5 FÓRMULA GERAL DE CLASSIFICAÇÃO DE PRÁTICAS ÁGEIS	67
5.5.1 Peso de Práticas Ágeis	67
5.5.2 Peso de fatores críticos.....	71
5.5.3 Peso de respostas do questionário	72
5.5.4 Fórmula Geral do Sistema.....	73
5.6 AVALIAÇÃO DE USO	73
5.7 CONSIDERAÇÕES FINAIS.....	77
CAPÍTULO 6 – CONCLUSÃO.....	77
6.1 CONSIDERAÇÕES FINAIS.....	77
6.2 LIMITAÇÕES	78
6.3 PRINCIPAIS CONTRIBUIÇÕES.....	78
6.4 TRABALHOS FUTUROS.....	79
REFERÊNCIAS.....	80
ANEXO A - RELACIONAMENTOS ENTRE ÁREAS DE CONHECIMENTO, PROBLEMAS DE PROJETOS DE SOFTWARE E FATORES CRÍTICOS DE SUCESSO AFETADOS	81
ANEXO B - PRÁTICAS ÁGEIS SELECIONADAS	84
ANEXO C - METODOLOGIAS ÁGEIS SELECIONADAS	87

ÍNDICE DE TABELAS

Tabela 1. Fatores Críticos de Sucesso para Projetos de Software.....	10
Tabela 2. Exemplo de Problema de Software.	13
Tabela 3. Diferenças entre Métodos Ágeis e Tradicionais de Software.	17
Tabela 4. Fatores Críticos de Sucesso e Práticas Ágeis	22
Tabela 5. Métodos e Práticas Ágeis	23
Tabela 6. Exemplo de Descrição de Prática Ágil.....	24
Tabela 7. Exemplo de Descrição Metodologia Ágil	25
Tabela 8. Lista de ferramentas e suas respectivas versões utilizadas na aplicação...	43
Tabela 9. Extratos com taxas de relevância referentes às práticas ágeis.....	68
Tabela 10. Primeira parte para montagem de uma Tabela de Frequência	69
Tabela 11. Tabela de Distribuição de Frequência	70
Tabela 12. Pesos de Práticas do sistema de apoio.....	71
Tabela 13. Pesos de questões do questionário.....	72
Tabela 14. Avaliação de uso do sistema GAPRO.....	74

ÍNDICE DE FIGURAS

Figura 1. Projetos presentes em diversas áreas de atuação (Berkun, 2008).....	5
Figura 2. Estrutura genérica de ciclo de vida e utilização de recursos (PMI, 2013)...	8
Figura 3. Modelo Lógico	44
Figura 4. Diagrama de fluxo de dados	46
Figura 5. Diagrama de sequência com funções do Administrador no sistema	48
Figura 6. Diagrama de sequência de gerenciamento de perfis.....	49
Figura 7. Diagrama de sequência de respostas ao questionário	50
Figura 8. Diagrama de sequência de geração de relatório	51
Figura 9. Tela de login	52
Figura 10. Tela de início do Administrador.....	53
Figura 11. Lista de Fatores Críticos	53
Figura 12. Tela de adição de item	54
Figura 13. Adição de Prática Ágil.....	55
Figura 14. Adição de Metodologia Ágil	56
Figura 15. Grupos de perguntas do questionário	56
Figura 16. Adição de problemas	57
Figura 17. Tela de início do Gerente.....	58
Figura 18. Tela de perfil do Gerente	59
Figura 19. Tela de adição de perfil	59
Figura 20. Definição de prioridades de fatores críticos	60
Figura 21. Tela de alerta do sistema	61
Figura 22. Tela de instruções do questionário	61
Figura 23. Lista de questões em um grupo do questionário.....	62
Figura 24. Lista de perfis de um Gerente na tela de relatórios	63
Figura 25. Aba do relatório com práticas ágeis que tratam cada questão	65
Figura 26. Aba do relatório com as práticas ágeis mais recomendadas.....	66
Figura 27. Aba do relatório com uma lista de metodologias ágeis recomendadas ...	66

TABELA DE ABREVIATURAS E SIGLAS

PMI	<i>Project Management Institute</i>
SAD	<i>Sistemas de apoio à decisão</i>
MVC	<i>Model View Controller</i>
PHP	<i>Hypertext Preprocessor</i>
HTML	<i>HyperText Markup Language</i>
CSS	<i>Cascading Style Sheets</i>
URL	<i>Uniform Resource Locator</i>
CRUD	<i>Create, Read, Update e Delete</i>
XP	<i>Extreme Programming</i>
TI	<i>Tecnologia da Informação</i>

CAPÍTULO 1 - INTRODUÇÃO

1.1 Motivação

Uma pesquisa anual organizada pelo *Project Management Institute* (2012) mostrou que cerca de 60% das organizações têm cumprido metas consideradas como fatores críticos para o sucesso de um projeto. Pela pesquisa, 40% das organizações consideram que fracassaram no cumprimento de suas metas, e que 61% dos projetos de TI no mundo também podem ser classificados como fracassos. Nos anos seguintes houve uma melhora desses percentuais, mas as metas ainda estão longe do ideal segundo os autores.

Projetos cada vez mais dinâmicos e mudanças constantes nos requisitos dos sistemas ainda em fase de desenvolvimento levam muitos gerentes de projetos a tentarem adotar uma abordagem mais voltada para práticas focadas em comunicação com as partes interessadas e desenvolvimento incremental de pequenos entregáveis, ao invés de grandes entregas e prazos extensos. Esse fator sozinho já sugere um afastamento das metodologias clássicas de desenvolvimento de sistemas de TI e gerenciamento de projetos como um todo e uma aproximação dos ideais presentes no desenvolvimento ágil.

Ao mesmo tempo, em projetos maiores ainda são adotadas práticas propostas por metodologias consideradas mais clássicas, como grande controle de tempo, recurso e escopo. No entanto, estas práticas podem ser adaptadas e mescladas, buscando a otimização do sucesso do projeto. Esta variedade de tipos de projetos exige uma grande experiência do gerente de projetos, que deve considerar se cada prática proposta por uma metodologia será realmente benéfica para aquele projeto, ou ainda se a determinada prática pode trazer algum risco adicional para o projeto, antes de utilizá-la.

A crescente tendência em mesclar e adaptar práticas de diferentes metodologias remete a uma postura pós-agilista (HIBBS *et al.*, 2009 *apud* LEAL, 2014), revelando uma visão com foco em adaptar metodologias a projetos, e não o inverso.

1.2 Objetivo

O objetivo deste trabalho é desenvolver um sistema de Apoio a Decisão, chamado de GAPRO, que auxilie gerentes de projetos de software a escolher as práticas ágeis propostas por metodologias ágeis de desenvolvimento de software e gerenciamento de projetos.

Com experiências e problemas relatados pelo próprio gerente de projetos durante a utilização do sistema, será possível definir o perfil de sua organização, que guiará o usuário durante sua interação no sistema.

Este perfil, é alinhado com fatos estatísticos apresentados nesta monografia, decorrentes de resultados de questionários e estudos apresentados por pesquisadores e com participação de profissionais da área de projetos.

Estes fatos auxiliaram no desenvolvimento de um algoritmo de relevância das metodologias e práticas ágeis atuais. Ao final do levantamento das informações necessárias, o sistema deverá apresentar ao usuário um conjunto de práticas adaptáveis e não conflitantes entre si que contribuirão para a mitigação de problemas e riscos dos projetos, tendo como consequência o aumento do percentual de sucesso de seus projetos.

1.3 Estrutura do Trabalho

Esta seção apresentou a introdução do trabalho. O restante do texto está estruturado nos seguintes capítulos:

- Capítulo 2 - Gestão de Projetos: serão apresentados conceitos relacionados a gestão de projetos e fatores críticos de sucesso em projetos para situar o leitor na proposta do trabalho. Além disso, serão apresentados problemas e riscos presentes em projetos de software e os fatores críticos de sucesso afetados por estes problemas.
- Capítulo 3 - Métodos e Práticas Ágeis: serão apresentados conceitos relacionados ao surgimento do Manifesto Ágil e às premissas e princípios ágeis. Serão discutidas as diferenças entre Métodos Tradicionais e Ágeis de desenvolvimento de software e apresentados os Métodos e Práticas Ágeis abordados nessa monografia. Por fim será apresentado o mapeamento

realizado entre os fatores críticos de sucesso em projetos de software, práticas e metodologias ágeis.

- Capítulo 4 - Sistemas de apoio à decisão: neste capítulo será feita a apresentação dos conceitos envolvidos e das características que definem o que é um Sistema de Apoio a Decisão.
- Capítulo 5 – Desenvolvimento do Sistema: Após a apresentação dos conceitos e práticas sobre o tema, será apresentada a estrutura do sistema GAPRO, que apoia a decisão da escolha de práticas ágeis adequadas a um projeto. A partir da interação com o usuário, que fornece o perfil da organização e equipe e relata as dificuldades enfrentadas em projetos de desenvolvimento, o GAPRO apresenta opções de práticas ágeis que podem mitigar tais dificuldades se utilizadas. Por fim será apresentada uma avaliação de uso da ferramenta.
- Capítulo 6 - Conclusão: apresenta as conclusões finais, incluindo as principais contribuições e sugestões para trabalhos futuros.

CAPÍTULO 2 - GESTÃO DE PROJETOS

Nesse capítulo, serão apresentadas definições de gestão de projetos. Será discutida a importância da utilização de projetos e suas características principais. Além disso, será apresentado o ciclo de vida de um projeto. Após as definições de projeto, o capítulo trata da área gerenciamento de projetos, que é responsável por definir uma estratégia para atender às necessidades de um projeto e processos de software para trazer o leitor para mais próximo do tema, que trata de desenvolvimento de software. O capítulo finaliza apresentando definições de fatores críticos de sucesso em projetos de software.

2.1 Definição de Projeto

A tarefa central dos projetos é combinar os trabalhos de diferentes pessoas em um todo singular e coerente que será útil para as pessoas ou clientes. Seja um projeto desenvolvido utilizando HTML, C++ ou cimento e aço, há sempre um conjunto principal de conceitos irrefutáveis que a maioria dos projetos deve compartilhar (Berkun, 2008). Este conceito é representado pela Figura 1, que mostra a similaridade de processos, mesmo em áreas profissionais diversificadas.

Um projeto só traz vantagens quando aplicado, independente do seu objetivo ou área de atuação. Metas, prazos, objetivos e custos inerente a um projeto irá sempre aumentar a chance de sucesso de um plano de ação.

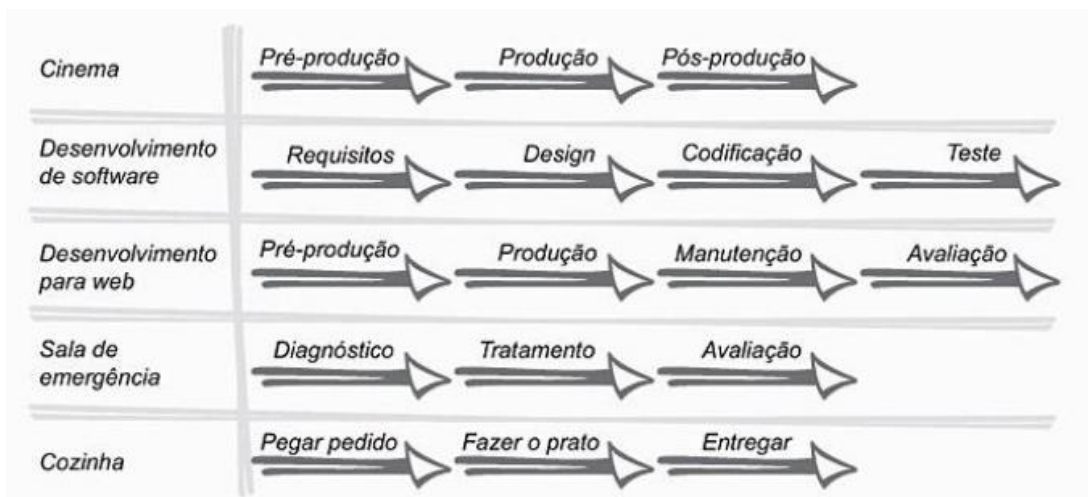


Figura 1. Projetos presentes em diversas áreas de atuação (Berkun, 2008)

Uma das definições utilizadas para projeto é apresentada pelo PMI – *Project Management Institute* (PMI, 2013), que diz que um projeto pode ser definido como um esforço temporário empreendido para criar um produto, serviço ou resultado exclusivo e que é utilizado como meio para atingir uma ou mais metas específicas dentro de um plano estratégico de uma organização. Várias outras definições são consideradas corretas e algumas são apresentadas a seguir.

Turman (1983) apresenta um conceito genérico e foca sua definição de modo mais direto e objetivo. O autor diz que um projeto é uma organização de pessoas dedicadas visando atingir um propósito e objetivo específico.

Seguindo a mesma linha de definição, Vargas (2003) define que projeto é um empreendimento não repetitivo caracterizado por uma sequência clara e lógica de eventos, com início, meio e fim, que se destina a atender um objetivo claro e definido, conduzido por pessoas dentro de parâmetros pré-definidos de tempo, custo, recursos envolvidos e qualidade.

Importante notar que a gestão de projetos pode ser aplicada a inúmeras necessidades de uma organização. Frequentemente, atividades de projeto e atividades rotineiras têm as mesmas necessidades, como relatórios, reuniões, análises, mas nem sempre estes esforços representam um projeto.

Uma das diferenças está nos objetivos das ações realizadas, sendo que os projetos possuem metas claras e definidas. Diferente do trabalho rotineiro, projetos são realizados em período definido de tempo e não indefinidamente, como os trabalhos

rotineiros (Vargas, 2003). Além disso, Vargas (2003) também destaca que todos os projetos são esforços, mas nem todos os esforços são projetos.

Dessa forma, é importante se conseguir identificar um projeto. De acordo com o PMPSG (2014), um projeto tem características bem definidas:

- Projetos são únicos;
- Projetos são temporários por natureza e possuem uma data de começo e um fim definidos;
- Projetos são considerados completos quando os objetivos do projeto foram atingidos;
- Um projeto é considerado um sucesso quando os requerimentos iniciais atingem ou excedem as expectativas das partes interessadas.

Vargas (2003) possui definição similar e diz que as principais características dos projetos são a temporariedade, a individualidade do produto, a complexidade e a incerteza. Para complementar os conceitos apresentados sobre o que define um esforço como um projeto, Vargas (2003) apresenta uma lista com maior detalhamento das características comuns de um projeto:

- **Empreendimento não repetitivo** – não faz parte da rotina da empresa, é algo novo para os envolvidos que o irão realizar.
- **Sequência clara e lógica de eventos** – caracterizado por atividades encadeadas logicamente, permitindo que durante a execução o acompanhamento e o controle sejam precisos.
- **Início, meio e fim** – respeita um ciclo de vida bem definido, possui uma característica temporal.
- **Objetivo claro e definido** – um projeto tem metas e resultados bem estabelecidos a serem atingidos em sua finalização.
- **Conduzido por pessoas** – sem a ação de pessoas, um projeto não existe, mesmo que disponha de equipamentos modernos de controle e gestão.
- **Projetos utilizam recursos** – um projeto utiliza recursos especificamente alocados a determinados trabalhos.

- **Parâmetros predefinidos** – um projeto necessita ter estabelecido, durante o plano de projeto, prazos, custos, pessoal, material e equipamentos envolvidos, bem como a qualidade desejada para o projeto.

2.2 Ciclo de vida de um projeto

As fases de um ciclo de vida de projeto são geralmente limitadas pelo tempo. Cada ciclo possui um início e término, sendo que as entregas e atividades específicas de cada ciclo variam muito de acordo com o projeto, organização e tecnologias adotadas. Uma definição mais abrangente, apresentada por Dennis *et al.* (2014), classifica um ciclo de vida de um projeto como “o processo de determinação do modo como um sistema de informações pode fornecer suporte para as necessidades da empresa, projetar o sistema, construí-lo e entregá-lo aos usuários”.

O PMI (2013) expõe o ciclo de vida de um projeto como “a série de fases pelas quais um projeto passa, do início ao término”. Uma definição bastante genérica, mas que ilustra de maneira prática o ciclo de vida.

Um projeto pode ser dividido em um número qualquer de fases, que representam um conjunto de atividades relacionadas de maneira lógica, as quais culminam na conclusão de uma ou mais entregas (PMI, 2013).

Ainda de acordo com as definições do PMBoK, independente dos objetivos de um projeto, todos podem ser mapeados por uma estrutura genérica de ciclo de vida. Essa estrutura, representada na Figura 2, é apresentada sequencialmente por (1) Início do projeto; (2) Organização e preparação; (3) Execução do trabalho do projeto e (4) Encerramento do projeto.

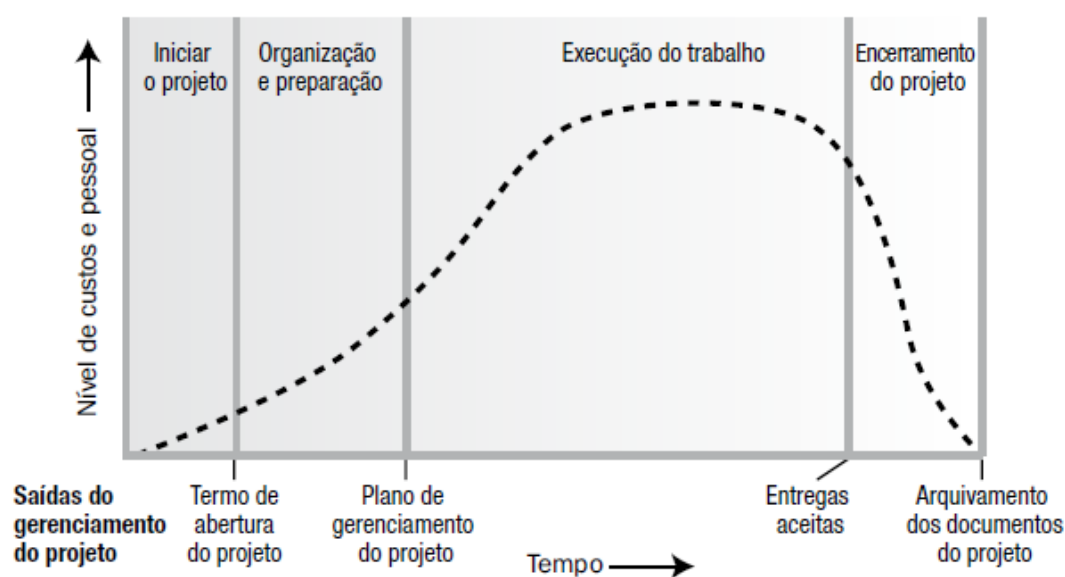


Figura 2. Estrutura genérica de ciclo de vida e utilização de recursos (PMI, 2013)

O gráfico da Figura 2 define bem os momentos que existem durante o ciclo de vida de um projeto de forma satisfatória, englobando projetos de diferentes tamanhos e objetivos. Estas particularidades são mencionadas a seguir (PMI, 2013):

- Para a grande maioria dos projetos, os níveis de custo e de pessoal são baixos no início, atingem um valor máximo enquanto o projeto é executado e caem rapidamente conforme o projeto é finalizado.
- Riscos e incertezas são maiores no início do projeto e diminuem ao longo da vida do projeto à medida que as decisões são tomadas e as entregas são aceitas.
- A possibilidade de alterações das características finais do produto do projeto diminui à medida que o projeto progride, porque o impacto nos custos é grande.

Uma outra análise que apresenta conclusão semelhante é a de Vargas (2003). O autor diz que “o nível de esforço destinado ao projeto inicia-se praticamente em zero e vai crescendo até atingir um valor máximo e, logo após esse ponto, reduz-se bruscamente até atingir o valor zero, representante do término do projeto”.

2.3 Gerenciamento de projetos

A dependência de tecnologia nas organizações atualmente exige um gerenciamento de projetos rígido, para que se possa dominar métodos, processos e

qualidade necessários para o sucesso de um projeto. Muitas vezes um projeto é baseado na experiência dos gerentes de projetos e em poucos dados de projetos anteriores.

Definir uma estratégia para atender às necessidades de um cliente é sempre um desafio, por isso a atividade de gerenciamento de projetos é fundamental para o sucesso de um desenvolvimento de sistemas.

Dentre benefícios do gerenciamento de projetos, pode-se destacar, a antecipação de situações de risco, otimização de rotinas, tomada de decisões mais corretas, maior controle da gerência, melhor comunicação da equipe, otimização na alocação de recursos, conhecimento de estimativas e prazos. Estes e mais benefícios estão presentes sempre que o gerenciamento de projetos é aplicado, e não é necessária uma grande operação para tornar isto possível, como afirma Vargas (2003): “A principal vantagem do gerenciamento de projetos é que ele não é restrito a projetos gigantescos, de alta complexidade e custo. Ele pode ser aplicado em empreendimentos de qualquer complexidade, orçamento e tamanhos, em qualquer linha de negócios”.

O PMBoK (PMI, 2013) define o gerenciamento de projetos como o uso do conhecimento, das habilidades, ferramentas e técnicas com a finalidade de suprir as necessidades e expectativas do empreendedor com relação a um projeto. A mesma visão é compartilhada na definição de Hiram (2011), que diz que o gerenciamento de projetos deve cuidar do dimensionamento do projeto, estabelecendo uma estratégia para atingir os resultados desejados dentro de restrições de custo e prazo.

Reeve, diretor do PMI em 1999, complementa a definição do PMBoK, um gerenciamento de projetos rigoroso proporciona um foco para que a comunicação, a coordenação e o controle se tornem eficientes, um plano para atingir o sucesso, com ênfase em tempo e custo. O gerenciamento de projetos também proporciona a estrutura para métodos, processo, monitoração e controle de mudanças (REEVE, 1999 *apud* VARGAS, 2003).

Um plano de projeto é o primeiro documento que o gerenciamento de projetos deveria produzir, este documento deve ser usado para acompanhar e controlar o andamento do projeto. Conforme o projeto é realizado, replanejamentos podem ser necessários caso algum desvio significativo seja detectado. O responsável pelo plano de projeto é o gerente de projetos que de acordo com o PMI (2013), é: “Um profissional no campo de gerência de projetos que tem a responsabilidade de planejar e controlar a

execução de projetos em diversas áreas de atuação, como a construção civil, arquitetura e desenvolvimento de software, entre outras”.

Destaca-se que um gerente de projetos precisa tratar de temas como o escopo do projeto, premissas, custos, riscos e restrições durante o desenvolvimento de um sistema, e é o plano de projeto que auxilia na divisão de tarefas e de recursos utilizados durante o tempo de vida de um projeto.

2.4 Fatores críticos de sucesso em projetos de software

Vezzoni (2011) apresenta fatores críticos de sucesso definidos por Leidecker e Bruno (1984) como as características, variáveis ou condições que quando geridas de modo correto causam um impacto relevante em relação aos concorrentes da organização. Segundo Forster e Rockart (1989), fatores críticos de sucesso auxiliam na gestão de prioridades do sistema de informação. Segundo Dvir *et al.* (1998), cada projeto possui um conjunto distinto de fatores críticos de sucesso.

Nasir e Sahibuddin (2011), citados por Pimenta (2016), realizaram uma revisão bibliográfica com 43 artigos publicados de 1990 a 2010. Como resultado os autores identificaram 26 fatores críticos de sucesso para projetos de software. A Tabela 1 lista estes fatores ordenados pela frequência de citação na bibliografia encontrada pelos autores.

Tabela 1. Fatores Críticos de Sucesso para Projetos de Software.

Fator Crítico de Sucesso	Frequência de citação
Requisitos e especificações claras	26
Objetivos e metas claros	24
Prazos realistas	23
Habilidades eficazes de gestão de projeto / metodologias (Gerente do projeto)	23
Apoio da alta administração	22
Usuário / envolvimento do cliente	20
Comunicação eficaz e feedback	20
Orçamento realista	19
Pessoal qualificado e suficiente	18
Requisito congelado	17
A familiaridade com a tecnologia / metodologia de desenvolvimento	15
Planejamento adequado	15
Processos adequados de desenvolvimento / metodologias (processo)	14
Relatório de progresso atualizado	12
Monitoramento e controle efetivo	12
Recursos adequados	11
Boa liderança	11
Gerenciamento de riscos	10
Complexidade, tamanho do projeto, duração, número de organizações envolvidas	10
Mudança efetiva e gerenciamento de configuração	10
Ferramentas de apoio e boa infraestrutura	9
Equipe comprometida e motivada	9

Fator Crítico de Sucesso	Frequência de citação
Bom gerenciamento da qualidade	9
Designação clara de papéis e responsabilidades	7
Bom desempenho de fornecedores / empreiteiros / consultores	4
Fornecimento de treinamento ao usuário final	2

Fonte: NASIR e SAHIBUDDIN (2011) *apud* PIMENTA (2016)

Os fatores listados na Tabela 1 foram utilizados como base para a elaboração de uma nova lista para o escopo deste projeto, detalhada na próxima Seção deste capítulo.

2.5 Considerações Finais

Este capítulo apresenta o que é projeto e suas características, passando pelo ciclo de vida de um projeto, a necessidade do gerenciamento de projetos, e fatores críticos que possibilitam maiores chances de sucesso em um projeto. O uso de práticas gerenciais auxilia o gerente no momento do planejamento de um projeto, pois ele pode prever quando e por quanto tempo recursos deverão ser alocados e pode identificar em que ponto um projeto se encontra no momento, permitindo maior capacidade de gerência, redução de riscos, controle de prazos e melhora na tomada de decisão.

No entanto, apenas o uso de práticas gerenciais não é suficiente. Como apresentado na introdução deste trabalho, uma grande porcentagem de sistemas não tem sucesso e muitos não chegam a ser finalizados, sendo necessário adotar medidas para minimizar os riscos durante um projeto. Para criar um software ou sistema é necessário implantar processos, métodos e ferramentas para tornar o desenvolvimento mais previsível independente de quem produza o software (HIRAMA, 2011).

Um modelo de processo de software fornece uma orientação para coordenar e controlar sistematicamente as tarefas que precisam ser executadas visando garantir o produto final e os objetivos do projeto (Tsui e Karam, 2013). Segundo Hirama (2011, *apud* SOMMERVILLE, 2011), “um processo é um conjunto de atividades que leva à produção de um produto de software, um processo define quais atividades devem ser realizadas, e um método para realizar estas atividades”. De maneira mais crítica, Tsui e Karanm (2013) definem processos de software como um “conjunto de tarefas, a sequência e o fluxo destas tarefas, as entradas e saídas, e as pré-condições e pós-condições para cada uma das tarefas envolvidas na produção de um software”.

Cada vez mais está sendo difundido nas organizações o uso de processos de software baseado em práticas ágeis de desenvolvimento, assunto do próximo capítulo.

Adicionalmente, de acordo com a revisão realizada, definir fatores críticos para o sucesso de um determinado projeto pode contribuir para o aumento no percentual de sucesso dos projetos. Com base na lista de fatores críticos identificados por Nasir e Sahibuddin (2011), foi elaborada uma nova lista para ser utilizada no escopo desse projeto:

- Requisitos e especificações claras;
- Objetivos e metas claros;
- Prazos realistas;
- Apoio da alta administração;
- Usuário / envolvimento do cliente;
- Comunicação eficaz e feedback;
- Orçamento realista;
- Pessoal qualificado e suficiente;
- A familiaridade com a tecnologia de desenvolvimento;
- A familiaridade com a metodologia de desenvolvimento;
- Planejamento adequado;
- Relatório de progresso atualizado;
- Monitoramento e controle efetivo do projeto;
- Gerenciamento de riscos;
- Controle efetivo de mudanças e gerenciamento de configuração;
- Equipe comprometida e motivada;
- Bom gerenciamento da qualidade;
- Designação clara de papéis e responsabilidades;
- Ferramentas de apoio.

Os fatores críticos a seguir, contidos na lista elaborada pelos autores Nasir e Sahibuddin (2011), não fizeram parte do escopo deste projeto por não haver nenhuma prática ágil dentre as listadas na Seção 3.7 que fosse relacionada a estes fatores.

- Habilidades eficazes de gestão de projeto / metodologias (Gerente do projeto)
- Requisito congelado
- Processos adequados de desenvolvimento / metodologias (processo)
- Recursos adequados
- Boa liderança
- Complexidade, tamanho do projeto, duração, número de organizações envolvidas
- Boa infraestrutura
- Bom desempenho de fornecedores / empreiteiros / consultores
- Fornecimento de treinamento ao usuário final

Após a elaboração da lista de fatores críticos que seriam considerados no escopo do projeto, foram levantados possíveis problemas em projetos de software que poderiam dificultar o cumprimento destes fatores. Os problemas foram escritos a partir dos fatores críticos considerados. A Tabela 2 contém um exemplo de problema:

Tabela 2. Exemplo de Problema de Software.

Área	Problema	Descrição do problema	Fatores críticos de sucesso que foram afetados
Pessoas	Falta de comunicação entre a equipe	Nos meus projetos foi verificado que a equipe não se comunica o suficiente. Os membros estão frequentemente desinformados sobre as atividades dos seus companheiros de equipe, podendo resultar em problemas como perda de tempo de projeto e retrabalho.	1. Comunicação Eficaz e feedback

O Anexo A contém a lista completa dos problemas levantados, a descrição de cada um e os fatores críticos de sucesso que são afetados por cada um dos problemas.

CAPÍTULO 3 - MÉTODOS E PRÁTICAS ÁGEIS

Neste capítulo serão apresentados o surgimento do Manifesto Ágil e as premissas e princípios ágeis. Serão discutidas as diferenças entre Métodos Tradicionais e Ágeis de desenvolvimento de software e apresentados os Métodos e Práticas Ágeis abordados nessa monografia.

3.1 Desenvolvimento Ágil de Software

Durante a evolução dos processos de Engenharia de Software, as baixas taxas de sucesso em projetos que faziam uso de métodos tradicionais de desenvolvimento de software motivaram especialistas na busca de soluções mais eficientes para desenvolver sistemas complexos (BASSI, 2008). Em 2001, um grupo de dezessete especialistas se reuniu em Utah, nos Estados Unidos, para definir um novo método de desenvolvimento de software que pudesse substituir os métodos tradicionais (AGILE MANIFESTO, 2013).

Após dois dias de debate, o grupo concluiu que desenvolver *softwares* é algo complexo demais para ser definido por um único processo. No entanto, o grupo chegou ao consenso de que alguns princípios eram determinantes para a obtenção de resultados positivos. Os resultados deste encontro foram a identificação de 12 princípios ágeis e a publicação do Manifesto Ágil (AGILE MANIFESTO, 2013), baseado em quatro premissas fundamentais:

- Indivíduos e iterações são mais importantes do que processos e ferramentas;
- Software funcionando é mais importante do que documentação completa;
- Colaboração com o cliente é mais importante do que negociação de contratos;
- Adaptação a mudanças é mais importante do que seguir o plano inicial.

Baseados nas premissas acima, os 12 princípios do desenvolvimento ágil, propostos pelo Manifesto Ágil são (AGILE MANIFESTO, 2013):

- A maior prioridade é satisfazer o cliente através da entrega contínua e adiantada de software com valor agregado;

- Mudanças nos requisitos são consideradas bem-vindas, mesmo ocorrendo em fases avançadas do desenvolvimento. Os processos ágeis devem aproveitá-las para oferecer vantagens competitivas para o cliente;
- Entregar frequentemente software funcionando, de poucas semanas a poucos meses, com preferência à menor escala de tempo;
- As equipes da área de negócio e de desenvolvimento devem trabalhar em conjunto durante todo o projeto;
- A construção dos projetos deve ocorrer em torno de indivíduos motivados. É preciso oferecer um ambiente e o suporte necessários e confiar neles para fazer o trabalho;
- O método mais eficiente e eficaz de trocar informações com uma equipe de desenvolvimento é através de conversa face a face;
- Software funcionando é a medida primária de progresso;
- Os processos ágeis promovem desenvolvimento sustentável. Os patrocinadores, desenvolvedores e usuários devem ser capazes de manter um ritmo constante indefinidamente;
- A contínua atenção à excelência técnica e bom design aumenta a agilidade;
- Simplicidade: a arte de maximizar a quantidade de trabalho que não precisou ser realizado;
- As melhores arquiteturas, requisitos e designs emergem de times auto organizáveis;
- Em intervalos regulares, o time deve refletir sobre como ficar mais efetivo e então, se ajustam e otimizam seu comportamento de acordo com esta reflexão.

3.2 Diferenças entre métodos ágeis e métodos tradicionais

Métodos tradicionais de desenvolvimento de software surgiram em um cenário de construção do software onde o custo para correção de erros ou qualquer outra alteração era muito elevado, pois o acesso aos computadores e processamento de dados era limitado e não havia ferramentas para auxiliar na construção de um software. Para mitigar riscos e problemas encontrados durante o desenvolvimento, era preciso

planejar e documentar muito bem o software antes do início do desenvolvimento. A partir desta ideia surgiram os processos de gerenciamento de software, baseados em documentação detalhada, planejamento extenso e etapas bem delimitadas (PRESSMAN, 2002).

Os métodos tradicionais possuem vantagens como a experiência obtida com projetos passados, a padronização do processo e a capacidade de seu refinamento e mensuração. A partir de dados de fatos já ocorridos, obtém-se a melhoria da capacidade do processo através da padronização, medição e controle do projeto. Mas muitos dos problemas e empecilhos vistos em gerenciamento de projetos de software estão relacionados ao nível de burocratização dos métodos tradicionais, dificultando mudanças de requisitos no projeto por possuir documentação extensa e aumentando o custo do projeto (NETO, 2009).

Métodos ágeis de gerenciamento de projetos apareceram como uma alternativa para se adaptar ao contexto do mercado atual, onde são exigidos resultados rápidos para atender às necessidades dos clientes, sob condições de constantes mudanças e incertezas (NETO, 2009). De acordo com Cockburn (2001), métodos ágeis possuem mais capacidade de responder de forma rápida e menos dispendiosa às mudanças de requisitos e se adaptam aos novos fatores que surgem durante o desenvolvimento do projeto.

A Tabela 3 faz uma comparação entre métodos ágeis e métodos tradicionais.

Tabela 3. Diferenças entre Métodos Ágeis e Tradicionais de Software. Fonte: PRIKLADNICKI (2014)

Característica	Tradicional	Ágil
Pressupostos fundamentais	Sistemas totalmente especificáveis, previsíveis; desenvolvidos a partir de um planejamento extensivo e meticuloso	Software adaptativo e de alta qualidade, pode ser desenvolvido por equipes pequenas utilizando os princípios da melhoria contínua do projeto e testes orientados a rápida resposta a mudanças
Controle	Orientado a Processos	Orientado a Pessoas
Estilo de gerenciamento	Comandar e controlar	Liderar e colaborar
Gestão do conhecimento	Explícito	Tácito
Atribuição de papéis	Individual - favorece a especialização	Times auto-organizáveis - favorece a troca de papéis
Comunicação	Formal	Informal
Ciclo do projeto	Guiado por tarefas ou atividades	Guiado por funcionalidades do produto
Modelo de desenvolvimento	Modelo de ciclo de vida (Cascata, Espiral, ou alguma variação)	Modelo Iterativo e incremental de entregas
Forma/estrutura organizacional desejada	Mecânica (burocrática com muita formalização)	Orgânica (flexível e com incentivos a participação de cooperação social)

Para determinar quais métodos ágeis seriam considerados, foi utilizada como base uma pesquisa realizada por Tainá Leal (2014), onde os métodos Scrum, Extreme Programming (XP), Kanban e Lean foram citados como os métodos mais adotados em projetos correntes pelos colaboradores da pesquisa. Estes mesmos métodos também foram citados como os métodos em que os colaboradores possuíam maior nível de experiência. Para esta monografia foram selecionadas práticas ágeis utilizadas por estes métodos, que serão descritos a seguir.

3.3 Scrum

Scrum é definido pelo site Desenvolvimento Ágil (2013) como "uma metodologia ágil para gestão e planejamento de projetos de software". Ken Schwaber e Jeff Sutherland o definem como "um framework para desenvolvimento e manutenção de produtos complexos" (SCHWABER *et al.*, 2014).

O Scrum consiste nos times do Scrum associados a papéis, eventos, artefatos e regras. Cada um destes itens serve a um propósito específico e é essencial para o uso e sucesso do Scrum. As regras do Scrum integram os eventos, papéis e artefatos, administrando as relações e interações entre eles (SCHWABER *et al.*, 2014).

No Scrum, os projetos são divididos em ciclos chamados de Sprints. Um Sprint é um período definido de um mês ou menos, durante o qual uma versão incremental potencialmente utilizável do produto é criada. Cada Sprint tem a definição do que é para ser construído, um plano projetado e flexível que guiará a construção, o trabalho e o resultado do produto (SCHWABER *et al.*, 2014).

As funcionalidades a serem implementadas em um projeto são mantidas em uma lista que é conhecida como Backlog do Produto. No início de cada Sprint, faz-se uma Reunião de Planejamento, na qual o dono do produto prioriza os itens do Backlog do Produto e a equipe seleciona as atividades que ela será capaz de implementar durante o Sprint que se inicia (DESENVOLVIMENTO ÁGIL, 2013).

A cada dia de um Sprint, a equipe faz uma breve reunião (normalmente de manhã), chamada Reunião Diária. O objetivo é disseminar conhecimento sobre o que foi feito no dia anterior, identificar impedimentos e priorizar o trabalho do dia que se inicia (DESENVOLVIMENTO ÁGIL, 2013).

Ao final de um Sprint, a equipe apresenta as funcionalidades implementadas em uma Reunião de Revisão do Sprint. Finalmente, faz-se uma Reunião de Retrospectiva do Sprint e a equipe parte para o planejamento do próximo Sprint. Assim reinicia-se o ciclo (DESENVOLVIMENTO ÁGIL, 2013).

3.4 Extreme Programming (XP)

Extreme Programming, ou XP, foi criada a partir de ideias de Kent Beck e Ward Cunningham e foi utilizada pela primeira vez em 1996 durante um projeto com a participação de Beck. Este afirma que XP "trata-se de uma metodologia de desenvolvimento de software ágil para equipes pequenas e médias desenvolvendo software com requisitos vagos e em constante mudança" (BECK, 2000).

O XP é um processo de desenvolvimento que busca assegurar que o cliente receba o máximo de valor de cada dia de trabalho da equipe de desenvolvimento. Ele é organizado em torno de um conjunto de valores

e práticas que atuam de forma harmônica e coesa para assegurar que o cliente sempre receba um alto retorno do investimento em software.

(TELES, 2004).

Teles (2004) afirma que o XP se baseia em quatro valores fundamentais: *feedback*, comunicação, simplicidade e coragem. Bonato (2002) acrescenta que "XP é uma maneira ágil, eficiente, de baixo risco, flexível, científica e até divertida de desenvolver software". Segundo Bonato (2002), o ciclo de vida de um projeto XP consiste de uma pequena fase inicial de desenvolvimento seguida por um longo período de refinamento e suporte à produção. O ciclo de vida da XP pode ser dividido nas fases de exploração, planejamento, iterações, produção, manutenção e fim do projeto.

Beck (2000) afirma que o XP faz uso de um conjunto de boas práticas de desenvolvimento já conhecidas e utilizadas com eficiência há muitos anos em projetos de software e as leva ao extremo. Estas práticas são executadas com base nos ideais pregados pelos valores do XP, e incluem Integração Contínua, Refatoração, Programação em Pares, Desenvolvimento Orientado a Testes e outras, combinando práticas técnicas e gerencias.

3.5 Kanban

O Nome Kanban é de origem japonesa e sua tradução seria como "sinal" ou "cartão". O nome surgiu dos sistemas de cartão utilizados nas indústrias de produção, que tinham como finalidade o gerenciamento do fluxo de trabalho através da organização de desenvolvimento (MARIOTTI, 2012).

Segundo Mariotti (2012), o Kanban tem como objetivo apresentar uma atividade de trabalho em processo. O autor ainda afirma que outra característica importante do Kanban é o conceito de "puxar tarefa" quando há capacidade de processá-la, diferente dos modelos tradicionais de "empurrar tarefa" conforme sua demanda, mantendo assim o bom desempenho da equipe. Ao invés de os membros que produzem o produto receberem atividades conforme suas demandas, os requisitos são adicionados à lista de *backlog* e "puxados" pelos membros que liberam suas atividades correntes e se tornam disponíveis para iniciar uma nova tarefa.

A implementação do Kanban se resume em três etapas: visualizar os processos, limitar o trabalho em processo e gerenciar o tempo que a atividade leva para passar por todas as suas fases até a sua entrega. Um quadro de tarefas do Kanban pode conter etapas como análise, desenvolvimento, aceitação e implantação (MARIOTTI, 2012).

No contexto de desenvolvimento de software, cada coluna do quadro do Kanban terá um processo em andamento estabelecido e representados pelo número máximo de cartões em cada fase. O cartão é composto por histórias de usuário, descrevendo seus requisitos. Todo cartão entra na fila de *backlog* e aguarda a liberação de capacidade para entrar nas colunas seguintes. Quando as atividades envolvidas com o cartão na coluna em andamento são finalizadas, o cartão é movido para a coluna seguinte, liberando espaço para entrada de um novo cartão (MARIOTTI, 2012). A Figura 3 apresenta um exemplo de um sinalizador visual Kanban. Nesta representação, o limite de cartões está representado pelos números no cabeçalho e os cartões ilustrados pelos retângulos representam uma breve história de usuário, ou seja, as demandas.

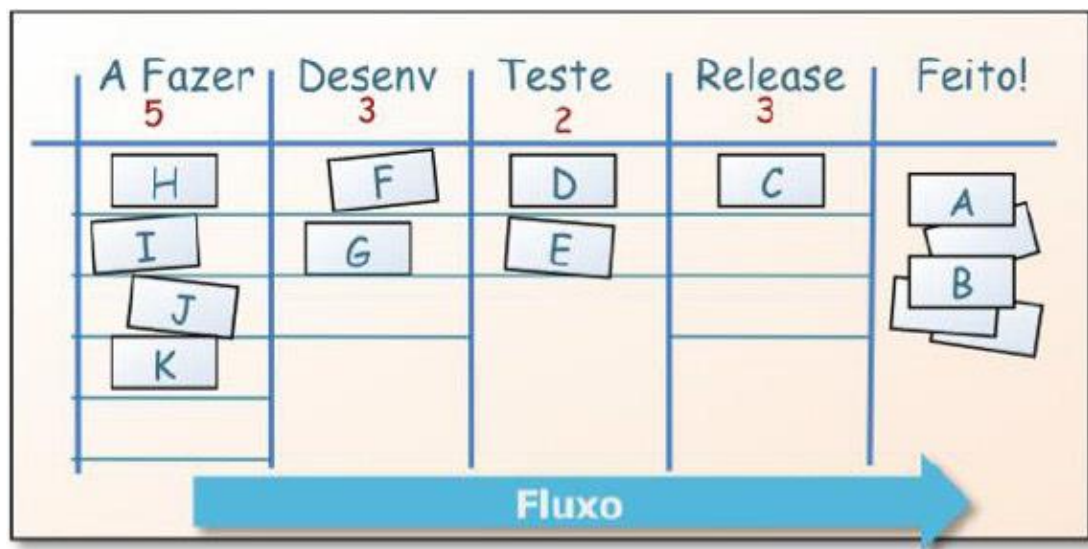


Figura 3. Ilustração de um sinalizador visual Kanban

Fonte: (KNIBERG *et al.*, 2009)

3.6 Lean

O *Lean Manufacturing* surgiu no Japão, na fábrica de automóveis Toyota, logo após a Segunda Guerra Mundial. Nesta época, a indústria japonesa tinha baixa produtividade e grande falta de recursos, o que a impedia de adotar o modelo de

produção em massa. O método Lean possui sete princípios: eliminar o desperdício, amplificar o aprendizado, adiar comprometerimentos e manter a flexibilidade, entregar rápido, tornar a equipe responsável, construir integridade e visualizar o todo (FADEL; SILVEIRA, 2010).

O desenvolvimento de software Lean é a aplicação dos princípios do sistema de desenvolvimento de produtos da Toyota para o desenvolvimento de software. Quando ele é aplicado corretamente, o desenvolvimento possui alta qualidade, velocidade e baixo custo (POPPENDIECK, 2007). Segundo a MICROSOFT DEVELOPER NETWORK (2016), o método de desenvolvimento de software Lean não prescreve práticas, sendo mais importante demonstrar que as definições dos processos estão alinhadas com os princípios e valores do Lean. No entanto, algumas práticas adotadas normalmente com o Lean incluem Reuniões Diárias, Visibilidade do Projeto, Cliente Presente, Integração Contínua, Desenvolvimento Orientado a Testes e Equipe Completa.

3.7 Considerações Finais

Este capítulo apresentou uma revisão da literatura sobre métodos ágeis. Para a construção da ferramenta descrita nessa monografia, foram selecionadas 18 práticas ágeis utilizadas nos métodos ágeis abordados:

- Backlog do Produto;
- Cliente Presente;
- Pequenas Liberações;
- Reuniões de Planejamento;
- Jogos de Planejamento;
- Equipe Completa;
- Visibilidade do Projeto;
- Reuniões Diárias;
- Scrum de Scrums;
- Propriedade Coletiva do Código;
- Desenvolvimento Orientado por Comportamento;
- Programação em Pares;

- Refatoração;
- Padronização de Código;
- Integração Contínua;
- Design Simples;
- Ritmo Sustentável;
- Desenvolvimento Orientado a Testes.

As práticas ágeis foram associadas aos fatores críticos de sucesso em projetos de software mencionados no Capítulo 2. Para cada fator crítico, foram associadas práticas ágeis que contribuem para a presença destes fatores em projetos. A Tabela 4 representa o relacionamento entre os fatores e as práticas ágeis.

Tabela 4. Fatores Críticos de Sucesso e Práticas Ágeis

Fator Crítico de Sucesso	Prática Ágil
Requisitos e especificações claras	Backlog do Produto
	Cliente Presente
	Pequenas Liberações
Objetivos e metas claros	Backlog do Produto
Prazos realistas	Reuniões de Planejamento
	Jogos de Planejamento
	Equipe Completa
Apoio da alta administração	Cliente Presente
	Visibilidade do Projeto
Usuário / envolvimento do cliente	Pequenas Liberações
	Cliente Presente
Comunicação eficaz e feedback	Backlog do Produto
	Reuniões Diárias
	Scrum de Scrums
	Visibilidade do Projeto
	Cliente Presente
	Propriedade Coletiva do Código
	Desenvolvimento Orientado por Comportamento
Orçamento realista	Reuniões de Planejamento
	Jogos de Planejamento
	Backlog do Produto
Pessoal qualificado e suficiente	Equipe Completa
A familiaridade com a tecnologia de desenvolvimento	Pequenas Liberações
	Programação em Pares
A familiaridade com a metodologia de desenvolvimento	Pequenas Liberações
	Reuniões diárias
	Scrum de Scrums
Planejamento adequado	Backlog do Produto
	Reuniões de Planejamento
	Jogos de Planejamento
	Visibilidade do projeto

Fator Crítico de Sucesso	Prática Ágil
Relatório de progresso atualizado	Reuniões diárias
	Scrum de Scrums
Monitoramento e controle efetivo do projeto	Reuniões diárias
	Scrum de Scrums
	Visibilidade do Projeto
Gerenciamento de riscos	Reuniões diárias
	Reuniões de Planejamento
	Scrum de Scrums
	Visibilidade do Projeto
Controle efetivo de mudanças e gerenciamento de configuração	Refatoração
	Padronização de Código
	Pequenas Liberações
	Cliente Presente
	Integração Contínua
	Design Simples
Equipe comprometida e motivada	Pequenas Liberações
	Ritmo Sustentável
Bom gerenciamento da qualidade	Integração Contínua
	Desenvolvimento Orientado por Testes
	Desenvolvimento Orientado por Comportamento
	Programação em Pares
	Refatoração
Designação clara de papéis e responsabilidades	Equipe Completa
Ferramentas de apoio	Integração Contínua
	Visibilidade do Projeto

Em seguida foi realizado um mapeamento do uso das práticas ágeis selecionadas por cada um dos métodos ágeis abordados, contido na Tabela 5:

Tabela 5. Métodos e Práticas Ágeis

Método	Prática
Scrum	<ol style="list-style-type: none"> 1. Backlog do Produto 2. Cliente Presente 3. Pequenas Liberações 4. Reuniões de Planejamento 5. Reuniões Diárias 6. Scrum de Scrums 7. Visibilidade do Projeto 8. Equipe Completa
XP	<ol style="list-style-type: none"> 1. Desenvolvimento Orientado a Testes 2. Visibilidade do Projeto 3. Equipe Completa 4. Ritmo Sustentável 5. Integração Contínua 6. Programação em Pares 7. Propriedade Coletiva do Código 8. Cliente Presente 9. Reuniões Diárias 10. Refatoração 11. Design Simples 12. Pequenas Liberações

Método	Prática
	13. Padronização de Código 14. Jogos de Planejamento
Kanban	1. Visibilidade do Projeto 2. Backlog do Produto
Lean	1. Visibilidade do Projeto 2. Cliente Presente 3. Pequenas Liberações 4. Desenvolvimento Orientado a Testes 5. Equipe Completa 6. Reuniões Diárias

As ferramentas de apoio utilizadas nos métodos Scrum e Kanban, como o gráfico *burndown*, as histórias de usuários e os quadros de tarefas foram consideradas como parte da prática ágil "Visibilidade do Projeto", e citadas nas descrições de suas respectivas práticas, contidas no Anexo B. As definições das práticas foram retiradas de Abrantes (2012), Desenvolvimento Ágil (2013), Fonseca (2016), Knowledge21 (2016), Ramos Júnior (2015) e Sousa (2016). A Tabela 6 contém um exemplo de descrição de prática ágil:

Tabela 6. Exemplo de Descrição de Prática Ágil

Prática	Descrição
Backlog do Produto	O Backlog é uma lista de todas as funcionalidades pensadas para um produto. Inicialmente, o Backlog começa com as necessidades mais básicas, mas geralmente muda com o tempo, de acordo com o aprendizado do time sobre o produto que está desenvolvendo e seus usuários (reais ou potenciais). Manter uma lista priorizada é importante para o fluxo de produção, para dar mais visibilidade às trocas que acontecem e aos efeitos dessas mudanças. Fazem parte do Backlog tarefas técnicas ou atividades diretamente relacionadas às funcionalidades solicitadas.

O Anexo C possui a descrição que foi inserida no sistema como fonte alternativa de informações sobre as metodologias ágeis abordadas. As definições destas metodologias foram retiradas de Beck (2000), Bernardo (2015), Desenvolvimento Ágil (2013), Fadel e Silveira (2010), Lean Institute Brasil (2010), Mariotti (2012), Medeiros (2016), Prikladnicki (2014), Schwaber e Sutherland (2014), Stefanini (2013) e Telles (2004). A Tabela 7 contém um exemplo de descrição de metodologia ágil:

Tabela 7. Exemplo de Descrição Metodologia Ágil

Metodologia	Descrição
Scrum	<p>Scrum é uma metodologia ágil para gestão e planejamento de projetos e desenvolvimento ágil de software. É utilizado para trabalhos complexos nos quais é impossível prever tudo o que irá ocorrer. O Scrum consiste nos times do Scrum associados a papéis, eventos, artefatos e regras. As regras do Scrum integram os eventos, papéis e artefatos, administrando as relações e interações entre eles.</p> <p>No Scrum, os projetos são divididos em ciclos (tipicamente mensais) chamados de Sprints. Cada Sprint representa uma iteração dentro da qual um conjunto de atividades deve ser executado.</p> <p>O Time Scrum é composto pelo Product Owner, o Time de Desenvolvimento e o Scrum Master. O Product Owner, ou dono do produto, é o responsável por maximizar o valor do produto e do trabalho do Time de Desenvolvimento. O Time de Desenvolvimento consiste de profissionais que realizam o trabalho de entregar uma versão usável que potencialmente incrementa o produto "pronto" ao final de cada Sprint. O Scrum Master é responsável por garantir que o Scrum seja entendido e aplicado, e ajuda aqueles que estão fora do Time Scrum a entender quais as suas interações com o Time Scrum são úteis e quais não são. O Scrum Master ajuda todos a mudarem estas interações para maximizar o valor criado pelo Time Scrum.</p>

CAPÍTULO 4 - SISTEMAS DE APOIO À DECISÃO

Neste capítulo serão discutidos os Sistemas de apoio à decisão (SAD), com a apresentação de breve histórico até os dias atuais, sua subdivisão de categorias e algumas definições de um SAD. Será possível entender as características que um sistema precisa para que possa ser considerado de Apoio a Decisão, e serão apresentados seus componentes, necessários para auxiliar no desenvolvimento de novos sistemas. Por fim, serão discutidas as vantagens e desvantagens da aplicação de um sistema de apoio.

4.1 Histórico

Com a ausência de uma definição única sobre o que é um Sistema de Apoio a Decisão, pode-se ter um maior entendimento sobre SAD através do seu histórico. O conceito de sistema capaz de apoiar as decisões surgiu muito cedo, segundo Keen e Scott Morton (1978), sendo o sistema de suporte informatizado a decisão é fruto da evolução das pesquisas:

- O estudo teórico da tomada de decisões nas organizações, realizado no Instituto de Tecnologia de Carnegie no fim dos anos 50, ano em que máquinas eram rudimentares e não interagiam com o usuário, esta área está presente também no início dos anos 60;
- Trabalhos técnicos em sistemas computacionais interativos, realizados pelo MIT (Instituto de Tecnologia de Massachusetts) nos anos 70, em que surgem os primeiros sistemas de apoio à decisão.

Durante a década de 1970, grandes empresas e vários grupos de pesquisa começaram a desenvolver Sistemas de apoio à decisão, que passaram a ser caracterizados como sistemas informáticos interativos, que ajudavam no processo decisivo de problemas considerados não estruturados (Sprague e Watson 1991).

Segundo os autores, já na década seguinte, 1980, pesquisadores evoluíram a definição destes sistemas com o objetivo de que estes pudessem auxiliar na formulação do processo de decisão, tendo como principais características:

- Orientados para problemas com menor especificação e não-estruturados;

- Aplicação de modelos ou técnicas analíticas em funções tradicionais de acesso e de recuperação de informações;
- Foco em recursos que auxiliem o uso dos sistemas por usuários com conhecimento limitado do tema;
- Permitir e incitar a flexibilidade e dinamismo em mudanças de ambiente e no processo da tomada de decisão.

Power (2007) indica que entre o final dos anos 1970 e década de 1980 iniciou-se o estudo dos sistemas de informação executiva (EIS), sistemas de apoio à decisão em grupo (GDSS) e sistemas de apoio à decisão organizacionais (ODSS).

Em 1979, John Rockart, da Harvard Business School, publicou um artigo que permitiu o desenvolvimento dos Sistemas de Apoio Executivos (ESS). Rockart desenvolveu o conceito de utilização de sistemas de informação para exibir métricas de sucesso crítico para gestores.

No início de 1980, foram desenvolvidos os Sistemas de apoio à decisão em Grupo (GDSS) por pesquisadores acadêmicos, como uma nova categoria de software, com o propósito de apoiar a tomada de decisão em grupo, facilitando os esforços na busca de soluções. Ainda nos anos 80, temos o estudo do Sistema de Apoio a Decisão Organizacional (ODSS) que se concentra em uma tarefa ou atividade organizacional que envolve uma sequência de operações de tomadores de decisão.

No início dos anos 90, surgiram a partir do SAD os conceitos de *data warehouse* e processamento analítico on-line (OLAP). Novas aplicações analíticas baseadas na web foram então introduzidas. Em 1999, muitos desenvolvedores e fornecedores de software mudaram seu foco e introduziram novas aplicações analíticas e soluções de negócio *web-based*.

Em 2000, os provedores de serviços de aplicação (ASPs) começaram a hospedar o software de aplicação e infraestrutura técnica voltadas para as capacidades de suporte a decisão.

Por mais interessante que seja conhecer os fatos importantes que ajudaram a moldar Sistemas de apoio à decisão como os conhecemos hoje, este não é o foco deste trabalho.

Fatos complementares entre os momentos principais apresentados são detalhados em Power (2007).

Com a evolução dos computadores, o desenvolvimento de sistemas informatizados de Apoio a Decisão tornou-se prática. Ao longo dos anos, diferentes pessoas perceberam estes fatos de vários pontos de visão diferentes, relatando suas experiências sobre o que aconteceu e o que é importante relatar. Como a tecnologia evolui, novas aplicações de Apoio a Decisão computadorizadas são desenvolvidas e estudadas. Pesquisadores usaram várias estruturas para ajudar a construir e compreender estes sistemas.

4.2 Categorias de um Sistema de Apoio a Decisão

Ao longo do tempo surgem novas tecnologias e conseqüentemente ramificações e possibilidades de aplicações de um tema. O mesmo ocorreu com os Sistemas de apoio à decisão. De acordo com Power (2007), hoje pode-se organizar a história de SAD em seis grandes categorias:

- ***Model-driven***, que enfatiza o acesso e a manipulação de recursos financeiros, otimização e modelos de simulação. Modelos quantitativos simples fornecem o nível mais básico de funcionalidade. *Model-Driven* utiliza dados limitados e parâmetros fornecidos pelos tomadores de decisão para auxiliar a análise de uma situação. Em geral grandes bases de dados gerais não são necessárias.
- ***Data-driven***, que acessa e manipula de dados internos da empresa, e algumas vezes dados externos e em tempo real, com a possibilidade de ações em um intervalo de tempo. Abrange desde sistemas de arquivos simples acessados por consultas à sistemas OLAP (*On-line Analytical Processing*), os quais fornecem o mais alto nível de funcionalidade e de Apoio a Decisão ligado à análise de grandes conjuntos de dados históricos.
- ***Communications-driven***, que auxilia mais de uma pessoa trabalhando em tarefas compartilhadas, utilizando tecnologias de rede e de comunicações para facilitar a decisão colaborativa e comunicação. Pode utilizar ferramentas como videoconferência ou quadro de anotações.
- ***Document-driven***, que utiliza tecnologias de armazenamento e processamento do computador para fornecer recuperação de documentos e análises. Grandes bancos de dados de documentos podem incluir documentos digitalizados, imagens, sons e vídeo.

- **Knowledge-driven**, que são sistemas especializados na resolução de problemas. Consiste no conhecimento sobre um determinado tema, através de informações armazenadas como fatos, regras, procedimentos ou estruturas similares, e a partir da compreensão do problema, apresenta soluções e alternativas para um problema específico, recomendando ações para o tomador de decisão.
- **Web-based driven**, oferece informação de Apoio a Decisão ou ferramentas de suporte de decisão para um usuário por meio de um navegador de internet. O servidor que está hospedando a aplicação SAD se conecta ao computador do utilizador através de uma rede com o protocolo TCP/IP.

O sistema de apoio a decisão desenvolvido neste projeto de final de curso é um sistema informatizado de apoio a decisão utilizado por meio de navegadores da internet e tem como funcionalidade apresentar soluções ao usuário baseada em informações armazenadas no sistema. De acordo com a classificação do autor, é então possível inserir o sistema de apoio a decisão desenvolvido neste projeto em duas categorias: *Knowledge-driven* e *Web-based driven*.

4.3 Conceitos

Seria possível apresentar dos mais diversos ângulos e generalizações, definições aceitas sobre o que torna um sistema, de decisão. Este fato pode ser confirmado por Druzdzel e Flynn (1999), já que os autores apontam que o conceito de Sistemas de apoio à decisão é extremamente amplo e varia de acordo com o ponto de vista de um autor. Para auxiliar na escolha da definição mais adequada, opta-se pela versão de Averweg (2012), que apresenta os termos gerais presentes na terminologia do conceito de ‘Sistemas de Suporte de Decisão’:

- **Sistema** - destaca a natureza integrada da abordagem global, entre máquina, utilizador e ambiente de decisão;
- **Suporte** - esclarece o papel do computador em auxiliar ao invés de substituir o tomador de decisão;
- **Decisão** - enfatiza o foco principal na tomada de decisões em um problema, e não somente em recuperação, processamento e relatórios de informações.

Entre os autores que já cunharam alguma definição para o SAD, e analisando o detalhamento dos itens apresentados por Averweg (2012), foi possível identificar de maneira mais exata, uma definição que melhor atenda aos itens apresentados pelo autor, esta definição é apresentada a seguir:

“Um sistema de informação interativo, flexível e adaptável baseado em computador, especialmente desenvolvido para apoiar a solução de um problema não-estruturado de gestão para melhorar a tomada de decisões. Ele utiliza dados, fornece uma interface fácil de usar, e permite as próprias percepções do tomador de decisão” (Turban, 1995).

Outras definições seguem a mesma lógica.

“Uma classe de sistema de informação que se baseia em sistemas de processamento de transações e interage com as outras partes do sistema de informação para apoiar as atividades de tomada de decisão dos gerentes e outros gestores do conhecimento nas organizações” (Sprague e Carlson, 1982).

“Um sistema baseado em computador que auxilia o processo de tomada de decisão” (Finlay, 1994).

“Sistema de Apoio a Decisão é um termo geral para qualquer aplicação de computador que melhora a capacidade de uma pessoa ou grupo para tomar decisões” (Power, 2005).

Quando analisadas, as definições citadas levam ao entendimento de que o SAD é uma ferramenta interativa, que possui uma base de informações que é acessada por meio de uma interface entre o sistema e o usuário, desenvolvida para apoiar a solução de um problema não estruturado e suportar a tomada de decisão pelo gerente. Nota-se que um SAD deve melhorar a qualidade das decisões, e não simplesmente automatizar um processo de escolha a ser seguido pelo gerente de projetos.

4.4 Características do SAD

Identificar características, funcionalidades distinguíveis, atributos ou aspectos de todos os Sistemas de apoio à decisão ajuda a distinguir um SAD de outros sistemas.

Uma vez que uma classificação precisa ocorre, somos mais propensos a identificar padrões e generalizações (POWER, 2005).

Exatamente por cada SAD ser desenvolvido com um objetivo específico e único dentro de uma organização, ainda hoje não há um consenso sobre as características definitivas de um SAD. Autores costumam identificar características específicas em sua definição. Cada uma destas definições incluem um número de características.

Alter (1980) identifica três características principais dos SAD: (1) são desenvolvidos especificamente para facilitar processos de decisão; (2) devem apoiar mais do que automatizar a tomada de decisão; e (3) devem responder rapidamente às mudanças de necessidades do tomador de decisão.

Já Holsapple e Whinston (1996) identificam quatro características que se espera encontrar em um SAD: (1) deve ter um corpo de conhecimento, ou seja, uma base de dados; (2) capacidade de manutenção de registros que podem apresentar conhecimentos numa base *ad hoc* em várias formas personalizadas bem como em relatórios padrões; (3) capacidade para selecionar um subconjunto de conhecimento armazenado para gerar um relatório ou derivar novos conhecimentos; e (4) deve ser projetado para interagir diretamente com um tomador de decisão de tal forma que o usuário tenha uma escolha flexível e uma sequência de atividades de gestão de conhecimento.

Para este trabalho, as características de um Sistema de Apoio a Decisão, serão descritas de acordo com a lista desenvolvida por Power (2005), que utiliza como base a lista de Turban e Aronson (1995).

Esta lista também é utilizada como base do sistema de apoio a decisão desenvolvido para este trabalho, pois o sistema em questão atende a todas as características apresentadas na lista.

- **Facilitação:** SAD facilitam e apoiam atividades específicas de tomada e processos de decisão.
- **Interação:** SAD são sistemas baseados em computadores projetados para uso interativo por tomadores de decisão ou usuário da equipe que controlam a sequência de interação e as operações realizadas.

- **Auxiliar:** SAD apoiam os tomadores de decisão em qualquer nível de uma organização. Eles não têm a intenção de substituir os tomadores de decisão.
- **Reutilização:** SAD são desenvolvidos para serem reutilizados. Um SAD específico pode ser usado rotineiramente ou somente quando necessário para tarefas de Apoio a Decisão específicas.
- **Orientado a tarefas:** SAD fornecem recursos específicos que suportam uma ou mais tarefas relacionadas com a tomada de decisões, entre elas: inteligência e análise de dados; identificação e concepção de alternativas; escolha entre alternativas; e implementação de decisões.
- **Identificação:** SAD podem ser sistemas independentes que recolhem ou replicam dados de outros sistemas de informação ou subsistemas de um sistema de informação maior e mais integrado.
- **Decisão de impacto:** SAD destinam-se a melhorar a precisão, pontualidade, qualidade e eficácia global de uma decisão específica ou um conjunto de decisões relacionadas.

4.5 Componentes do SAD

Componente é uma parte que se destaca dentro de uma entidade maior. Componentes podem ser implementados com diferentes tecnologias e cada componente tem um propósito diferente (Power, 2005). Um SAD deve possuir partes bem definidas com suas determinadas funções, que devem trabalhar em conjunto para que um sistema funcione. Sprague e Carlson (1982) classificam a construção de um Sistema de Apoio a Decisão em quatro componentes principais:

1. Interface do usuário;
2. Banco de dados;
3. Modelos e ferramentas analíticas;
4. Arquitetura do SAD e rede;

De acordo com Sprague (1980), um Sistema de Apoio a Decisão é constituído por três conjuntos de capacidades, que fornecem um esquema conveniente para identificar a capacidade técnica que um SAD deve atender:

- Software de gerenciamento de banco de dados (SGBD);
- Software de gerenciamento de base do modelo (SGBM);

- Software para gerenciar a interface entre o usuário e o sistema;

Esta lista é bem vista ainda hoje, pois serve como base para que se possa identificar as similaridades e diferenças entre os tipos existentes de sistemas. Além disso, entender os componentes técnicos de um Sistema de Apoio a Decisão pode ajudar gerentes e analistas de sistemas de informação a desenvolver SAD inovadores (POWER, 2005).

De acordo com uma análise de Power (2005), um componente pode ser desenvolvido para um fim específico dentro de um sistema específico ou pode ser desenvolvido como um módulo, pacote ou plug-in, para que possa ser utilizado por outros sistemas já em funcionamento. Entender a arquitetura e requisitos técnicos de um SAD reduzem seus custos de desenvolvimento e manutenção, permitem sua reutilização, mais funcionalidades e maior complexidade destes sistemas.

Ao observar as classificações dos autores, entende-se que há um foco bastante técnico em suas análises. Dividir um Sistema de Apoio a Decisão em subsistemas, permite a possibilidade de reutilização de funcionalidades e alguma independência entre as partes. Mas é importante citar que mesmo com subsistemas bem definidos, não se deve ignorar um componente vital para o sucesso do sistema, o componente humano. Os subsistemas devem ser desenvolvidos voltados sempre para a interação e o auxílio de decisão do usuário.

4.6 Vantagens e desvantagens

Antes dos Sistemas de apoio à decisão serem criados e evoluírem como ferramenta comum entre empresas e profissionais de diferentes áreas, já havia a necessidade de se tomar decisões sobre diversos temas específicos, não somente em projetos, mas qualquer decisão crítica em um determinado processo, como visto na seção 3.1.

Com o grande histórico dos SAD até hoje, temos a possibilidade de apontar vantagens em relação a outros métodos e também perceber em que momentos um SAD pode não ser a melhor opção dependendo do problema a ser tratado.

Uma fonte adequada de informação é a lista criada por Power (2007). Esta lista foi criada baseada no histórico de questões observadas pelo autor em seu website, e com a análise de artigos públicos no *Decision Support Systems Journal*¹.

4.6.1 Vantagens

1. **Economia de tempo.** Ciclo reduzido de tempo de decisão, o aumento da produtividade dos funcionários e informação mais completa para tomada de decisão.
2. **Aumento da eficácia.** Tomada de decisões mais eficaz e melhores decisões.
3. **Melhora na comunicação interpessoal.** SAD podem melhorar a comunicação e colaboração entre os tomadores de decisão. Um SAD pode criar "uma versão da verdade" sobre as operações da empresa disponíveis para gestores e, portanto, pode incentivar a tomada de decisão baseada em fatos.
4. **Vantagem competitiva.** Embora seja possível ganhar uma vantagem competitiva com um Apoio a Decisão informatizado, este não é um provável resultado, pois fornecedores costumam vender o mesmo software para muitas empresas, o que não diminui a importância desta pequena vantagem.
5. **Redução de custos.** Redução de trabalho na tomada de decisões e de menores custos de infraestrutura ou de tecnologia.
6. **Aumento da satisfação do tomador de decisão.** SAD pode reduzir frustrações dos tomadores de decisão, criar percepções que uma melhor informação está sendo usada e criar percepções de que o usuário é um melhor tomador de decisão.
7. **Estímulo da aprendizagem.** Aprendizagem de novos conceitos e o desenvolvimento de um melhor entendimento do negócio e do ambiente de tomada de decisão.
8. **Aumento do controle organizacional.** SAD muitas vezes tornam as informações disponíveis para monitoramento de desempenho e consulta, fornecem dados de resumo sobre as decisões tomadas, os usos e as

¹ ISSN: 0167-9236.

recomendações do sistema. Tais sistemas podem melhorar o entendimento de gerenciamento de operações de negócios, o que é útil para gestores.

4.6.2 Desvantagens

1. **Superestimar a tomada de decisão.** É evidente que o foco de todos aqueles interessados em Apoio a Decisão computadorizado é sobre tomar decisões. Mas é importante entender o contexto mais amplo da tomada de decisões e os fatores sociais, políticos e emocionais que impactam o sucesso organizacional. É importante analisar quando e em que circunstâncias um SAD deve ser utilizado e se a decisão a ser tomada é adequada para qualquer SAD ou se um SAD específico deve ser desenvolvido.
2. **Suposição de relevância.** Um cuidado a ser tomado quando um SAD se torna comum nas organizações, é que os gestores o utilizem de forma adequada. Não se pode considerar que, quando um SAD é implementado, ele será a única e principal tarefa e preocupação do gerente ou usuário.
3. **Transferência de poder.** Implementar um SAD pode ser entendido como a transferência de poder de decisão a um software. Um SAD precisa manter um tomador de decisão humano no "Ciclo de decisão", já que o fator humano é um dos componentes chave na estrutura de um SAD.
4. **Efeitos inesperados.** Implementar tecnologias de Apoio a Decisão podem ter consequências inesperadas, como a redução da qualidade da tomada de decisão. Alguns SAD sobrecarregam o usuário com muitas informações, e isso pode ter um efeito contrário, reduzindo a eficácia da tomada de decisões.
5. **Responsabilidade indefinida.** Algumas pessoas podem desviar a responsabilidade pessoal para um SAD. Mas sabe-se que um Sistema de Apoio a Decisão é um intermediário entre as pessoas que desenvolveram o sistema e os usuários do sistema, logo, a responsabilidade associada com a tomada de uma decisão usando um SAD deve persistir em um dos dois atores.
6. **Falsa crença na objetividade.** Os usuários de um SAD podem ser mais objetivos em sua tomada de decisão. Um software pode incentivar a ação mais racional, mas os gerentes também podem usar tecnologias de Apoio a Decisão para racionalizar suas ações.

7. **Redução de Status.** Alguns usuários argumentam que utilizar um SAD irá diminuir seu status e forçá-los a realizar funções menores na equipe. Gerentes e equipes que defendem a construção e utilização de Apoio a Decisão informatizado precisam lidar com todas as questões de status que possam surgir.
8. **Sobrecarga de informação.** Muita informação é um grande problema para pessoas e muitas DSS aumentar a carga de informações. SAD devem auxiliar os usuários a organizar e usar a base de informações existente. Um SAD deve poder reduzir e gerenciar a carga de informações de um usuário. Desenvolvedores precisam medir a carga de informação gerada pelo usuário e base de dados do sistema e monitorar suas percepções sobre quanta informação deve apresentar.

4.7 Considerações Finais

Neste capítulo, passando pela história dos sistemas de apoio é possível entender que há muito tempo já se havia a necessidade de tomadas de decisão mais acertadas. Consequentemente, houve o surgimento dos Sistemas de apoio à decisão.

Com o passar dos anos, muitos sistemas de apoio foram desenvolvidos em diversas áreas de aplicação, surgindo então, a necessidade de classificá-los em categorias.

De acordo com Power (2007), hoje temos seis categorias que ajudam a organizar e identificar sistemas de apoio.

Após identificadas as necessidades de um sistema de apoio a decisão, foram apresentadas definições, que ajudam a identificar e classificar um sistema de software como um Sistema de Apoio a Decisão. Como complemento, foram apresentadas as características e os componentes que identificam um SAD, que estão presentes no sistema de apoio desenvolvido neste trabalho.

Uma seção de vantagens e desvantagens é listada no trabalho e ajuda o leitor a entender melhor porque os Sistemas de apoio à decisão existem e quando são

necessários. A lista foi criada baseada em uma série de perguntas recebidas no site de Power² e agrupam dúvidas recorrentes entre os profissionais.

No próximo capítulo será apresentado o sistema de apoio GAPRO, desenvolvido para representar de forma prática o que foi exposto até agora neste trabalho.

² <http://dssresources.com/>

CAPÍTULO 5 - DESENVOLVIMENTO DO SISTEMA

Neste capítulo, serão apresentadas as funcionalidades do sistema GAPRO, os diagramas do projeto, conceitos de design de software, ferramentas utilizadas no desenvolvimento e, por fim, as telas criadas.

5.1 Funcionalidades do GAPRO

Para que se possa demonstrar de maneira prática os conceitos apresentados neste trabalho, foi desenvolvido o protótipo de um sistema de Apoio a Decisão. O sistema GAPRO (Gerenciamento Ágil de Projetos) tem como público alvo profissionais da área de gestão de projetos, e é recomendada sua utilização quando este percebe alguma dificuldade em solucionar problemas em um projeto ou equipe.

O sistema atende, ao mesmo tempo, usuários com pouca experiência que buscam assistência com relação a conceitos ágeis, e também usuários experientes, que, já com bons conhecimentos sobre o tema, conseguem identificar problemas a serem tratados, e com o auxílio do sistema de apoio GAPRO, encontram recomendações que atendem a suas necessidades.

Ao final da execução do sistema, espera-se que o gerente de projetos possa consultar um relatório com informações relevantes de recomendações de ações utilizando práticas e metodologias ágeis, que auxiliem na resolução dos problemas identificados em sua equipe ou projeto.

Para apresentar as funcionalidades do sistema, é preciso entender os papéis que um usuário pode ter no sistema de apoio. Dois grupos de usuários podem interagir com o sistema: Administrador e Gerente. As funcionalidades do sistema listadas nesta seção são detalhadas durante todo o capítulo por meio de diagramas e imagens.

Um Administrador, como o nome sugere, tem o papel de administrar o sistema, sendo o responsável por manter toda a base de dados do sistema atualizada e coerente. Com o conjunto de dados apresentados e estudados neste trabalho, primeiramente são definidas as relações entre estas informações, e em seguida, estes elementos são adicionados ao sistema, para que haja a interação com os usuários.

Após a inserção dos dados e suas relações no sistema, o administrador passa a ser responsável pela manutenção da base de dados e adição de novas informações, como cadastro de novos fatores críticos, práticas e metodologias. Este usuário também pode excluir qualquer informação da base de dados por meio do sistema.

Funcionalidades do sistema para um Administrador:

- Cadastrar e gerenciar fatores críticos
- Cadastrar e gerenciar práticas ágeis
- Cadastrar e gerenciar metodologias ágeis
- Cadastrar e gerenciar problemas do questionário
- Relacionar problemas com fatores críticos
- Relacionar práticas ágeis com fatores críticos
- Relacionar metodologias ágeis com práticas ágeis

O outro grupo pertencente ao sistema GAPRO é o grupo de usuário Gerente, que de uma forma geral, interage com o sistema adicionando informações, as quais são relacionadas com as informações cadastradas na base de dados do sistema por meio de um algoritmo lógico, que gera como saída um relatório de informações com recomendações de práticas ágeis e metodologias ágeis para tratar situações problemáticas em um ambiente de projetos.

Funcionalidades do sistema para um Gerente:

- Criar conta de usuário no sistema
- Perfil
 - Criar e gerenciar um ou mais perfis
 - Definir perfis criados como ativo
 - Classificar fatores críticos de sucesso em um perfil
- Questionário
 - Visualizar detalhes de problemas
 - Responder problemas do questionário
 - Salvar respostas, vinculando a um perfil ativo
 - Atualizar respostas de problemas
- Relatório
 - Gerar relatório para o perfil ativo do Gerente

- Apresentar tipos de visualização do relatório por meio de abas
- Salvar o relatório no formato .PDF³
- Imprimir o relatório

5.2 Arquitetura do sistema

De acordo com Hirama (2011), modelos de arquitetura são representações do sistema através de subsistemas e suas interfaces. O agrupamento de subsistemas pode ser feito seguindo algumas estruturas ou estilos de arquitetura de sistema. O autor apresenta uma lista de modelos, com definições e exemplos para cada um deles: (1) Modelo baseado em repositório; (2) modelo cliente-servidor; (3) modelo de implantação; (4) modelo de domínio específico; (5) modelo de controle e (6) modelo em camadas.

O sistema GAPRO é baseado no modelo em camadas e modelo cliente-servidor.

No modelo em camadas, é utilizado o modelo MVC, que permite uma organização dos subsistemas em camadas especializadas que se comunicam por meio de interfaces.

O modelo organiza o sistema em conjunto de camadas sendo que cada uma delas fornece um conjunto de serviços. A estrutura em camadas possibilita o desenvolvimento incremental dos subsistemas em camadas diferentes, deste modo, quando uma camada muda, somente a camada adjacente é afetada (Hirama, 2011).

O modelo cliente-servidor é possível pela utilização do PHP no desenvolvimento do sistema.

Este modelo é adequado quando o sistema é organizado como um conjunto de serviços e servidores/clientes que acessam esses serviços. Os componentes desse modelo são servidores dedicados que fornecem serviços específicos tais como impressão, gerenciamento de dados, etc., clientes que

³ Extensão de formato de arquivo que pode conter textos imagens e gráficos em um formato independente de dispositivo ou resolução.

chamam estes serviços e uma rede que permite aos clientes acessar os servidores (Hirama, 2011).

5.2.1 Padrão MVC

MVC é um padrão de design de projetos de software que separa lógica e negócio da apresentação. Um MVC implementa 3 distintas, que se comunicam entre si e que dão nome a sua sigla, Model (Modelo), View (Vista) e Controller (Controlador), cada parte com suas características e atribuições em uma aplicação. Gabardo (2012) conceitua de forma prática cada camada:

- **Modelo** – componentes da parte de abstração de dados, sendo cada entidade do modelo uma classe. O mais comum na utilização prática dos modelos é que a eles sejam atribuídas as tarefas de gravar e recuperar dados do banco de dados. Essa abordagem de estruturação dos objetos também facilita a integração em bancos de dados ou frameworks que fazem o mapeamento das entidades do banco de dados, gerando classes capazes de executar tarefas de rotina, como inserções e leituras dessas entidades.
- **Vista** – Parte que apresenta os dados gerados pela aplicação para o usuário, recebem dados dos controladores, mas não recebem e nem enviam dados diretamente aos modelos ou banco de dados. Vistas, em geral, são arquivos contendo estruturas HTML e outros formatos de saída..
- **Controlador** – Responsável pela comunicação entre a parte de dados (Modelo) e de apresentação (Vista). São responsáveis por carregar os modelos e vistas, receber informações e validá-las antes de encaminhá-las a próxima camada. Sempre que estamos acessando uma URL, obrigatoriamente estamos acessando um método de um controlador.

Com a proposta de escalabilidade do sistema em projetos futuros, onde novas funcionalidades serão implementadas, o padrão MVC foi escolhido.

5.2.2 Ferramentas utilizadas

Um software, necessariamente precisa ser desenvolvido com o auxílio de ferramentas que, em conjunto, permitem a codificação, armazenamento de dados e formatação de informações para que se atinja seu objetivo proposto. Serão apresentadas a seguir as ferramentas utilizadas no desenvolvimento do software.

5.2.2.1 PHP

PHP (Hypertext Preprocessor), Pré-processador de Hypertexto, linguagem que permite criar sites Web dinâmicos, possibilitando uma interação com o usuário através de formulários, parâmetros da URL e links. O código PHP é executado no servidor, sendo enviado para o cliente apenas HTML puro. Desta maneira, é possível interagir com banco de dados e aplicações existentes no servidor, com a vantagem de não permitir o acesso do código fonte pelo cliente (Xavier, 2008).

5.2.2.2 CodeIgniter

Criado por Rick Ellis⁴, o CodeIgniter⁵ é um framework de código aberto para desenvolvimento de aplicações em linguagem PHP. Seu objetivo é permitir desenvolver projetos com muito mais rapidez. Oferece um conjunto rico de bibliotecas para tarefas de uso comum, assim como uma interface simples e estrutura lógica para acessar as bibliotecas. O *framework* CodeIgniter permite que o desenvolvedor se concentre na criatividade e no desenvolvimento de um projeto, minimizando a quantidade escrita de código-fonte necessário para executar uma mesma tarefa (Gabardo, 2012). O framework CodeIgniter possibilita que se possa focar no negócio da aplicação, tornando tarefas repetitivas e complexas fáceis de serem implementadas. Por este motivo, ela foi a ferramenta escolhida para a implementação do protótipo.

Mais duas bibliotecas foram utilizadas em conjunto com o CodeIgniter para dar ainda mais agilidade no desenvolvimento e aumentar o foco no objetivo do projeto. A biblioteca *GroceryCrud*⁶ foi utilizada para criar rapidamente funcionalidades de

⁴ <https://ellislab.com/team/member/rick-ellis>

⁵ <https://www.codeigniter.com>

⁶ Biblioteca *open source* que permite a criação de operações CRUD (acrônimo de Create, Read, Update e Delete), <http://www.grocerycrud.com/>

CRUD e tabelas de apresentação na vista com o tema Twitter Bootstrap⁷ e a biblioteca *IonAuth* 2⁸, que integra funcionalidades de autenticação e gerenciamento de usuários.

5.2.2.3 MySQL e phpMyAdmin

Para que a aplicação possa apresentar informações na interface com as quais o usuário possa interagir, é necessária a utilização de um banco de dados e de um gerenciador que permita guardar e manipular os dados obtidos. *MySQL*⁹ é um software de banco de dados que suporta linguagem de consulta de banco de dados chamada de *SQL*. A *SQL* é um padrão de comunicação com banco de dados de qualquer tipo, não importado os métodos subjacentes de escrever e ler os dados (Maxfield, 2002).

Neste trabalho utilizou-se o software phpMyAdmin para gerenciar o MySQL.

5.2.2.4 Lista de ferramentas

Para referência, as ferramentas utilizadas e suas versões podem ser consultadas na tabela 8.

Tabela 8. Lista de ferramentas e suas respectivas versões utilizadas na aplicação

Item	Versão
Codeigniter	3.0.6
PHP	5.5.15
HTML	5
CSS	3
JavaScript	1.8
phpMyAdmin	4.2.7.1
Astah Professional Trial	7.0.0
Sublime Text	2.0.2

5.3 Modelagem do sistema

No início do desenvolvimento, é importante optar pela criação de documentos e diagramas que auxiliem na compreensão da construção de um sistema. Esta seção contém a especificação formal do sistema, onde serão apresentados:

⁷ Framework front-end para desenvolvimento responsivo de aplicações web, <http://getbootstrap.com/>

⁸ Biblioteca de autenticação de acesso para o framework CodeIgniter, <https://github.com/benedmunds/CodeIgniter-Ion-Auth>

⁹ Sistema de gerenciamento de banco de dados (SGBD), que utiliza a linguagem SQL como interface, <https://www.mysql.com/>

- Modelo Lógico;
- Diagrama de Fluxo de Dados;
- Diagrama de Sequência para múltiplas ações.

5.3.1 Modelo Lógico

O design do banco de dados lógico implica em um conjunto de tabelas, juntamente com relações-chave externas (Tsui e Karam, 2013).

Este diagrama tem como principal objetivo facilitar o projeto de banco de dados, possibilitando a especificação da estrutura lógica geral do banco de dados.

Com o modelo lógico foi possível identificar com maior precisão como os dados deveriam estar organizados. O modelo é representado pela Figura 4 e suas principais entidades são apresentadas a seguir.

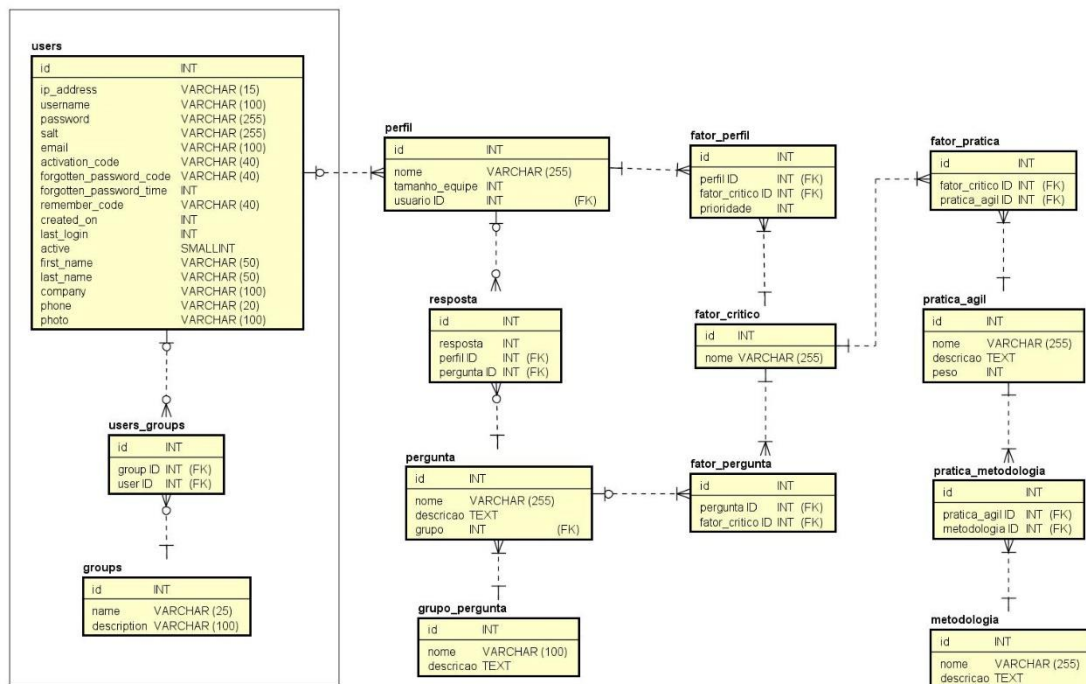


Figura 3. Modelo Lógico

- **perfil** – entidade que armazena o perfil de um Gerente, registra o nome do perfil e o tamanho da equipe que o usuário é responsável.
- **resposta** – entidade que armazena as respostas de um Gerente quando este responde ao questionário do sistema.

- **pergunta** – armazena as perguntas sobre os problemas cadastrados por um Administrador e são relacionadas com um ou mais fatores críticos.
- **grupo_pergunta** – cada pergunta do sistema fica obrigatoriamente dentro de um grupo de pergunta. Cada um destes grupos é apresentado como uma aba no questionário do sistema de apoio.
- **fator_critico** – armazena os fatores críticos do sistema.
- **pratica_agil** – armazena as práticas ágeis do sistema, são relacionados a um ou mais fatores críticos. Esta entidade também possui uma descrição que pode ser apresentada no relatório do sistema gerado por um Gerente.
- **metodologia_agil** – armazena as metodologias ágeis do sistema e são relacionados a uma ou mais práticas ágeis. Esta entidade também possui uma descrição que pode ser apresentada no relatório do sistema gerado por um Gerente.

As entidades ‘fator_pratica’, ‘fator_pergunta’, ‘fator_perfil’ e ‘pratica_metodologia’ são Entidades associativas, estas entidades que surgem quando há um relacionamento do tipo muitos para muitos. Nestes casos, é necessária a criação de uma entidade intermediária cuja identificação é formada pelas chaves primárias das outras duas entidades (DevMedia, 2014).

As entidades ‘users’, ‘user_groups’ e ‘groups’ são entidades necessárias para controlar o acesso e permissões de usuários no sistema, sendo obrigatórias no desenvolvimento do sistema ao usar a biblioteca IonAuth 2, citada na Seção 5.2.2.2. Por este motivo, os campos destas tabelas se mantêm com seus nomes em inglês, para evitar erros durante o desenvolvimento.

5.3.2 Diagrama de Fluxo de Dados

Há quatro símbolos na linguagem dos diagramas de fluxo de dados: (1) Processos; (2) Fluxo de dados; (3); Depósito de dados e (4) Entidades externas (Dennis *et al.*, 2014).

A Figura 5 mostra um DFD que representa o fluxo de dados de todo o sistema, englobando os dois grupos presentes: Administrador e Gerente.

Analisando a figura, é possível identificar a relação entre os grupos de usuários. O Administrador é responsável por alimentar toda a base de dados do sistema GAPRO.

Com estes dados previamente cadastrados, o Gerente pode então, utilizar as ferramentas do sistema para interagir com as informações disponíveis.

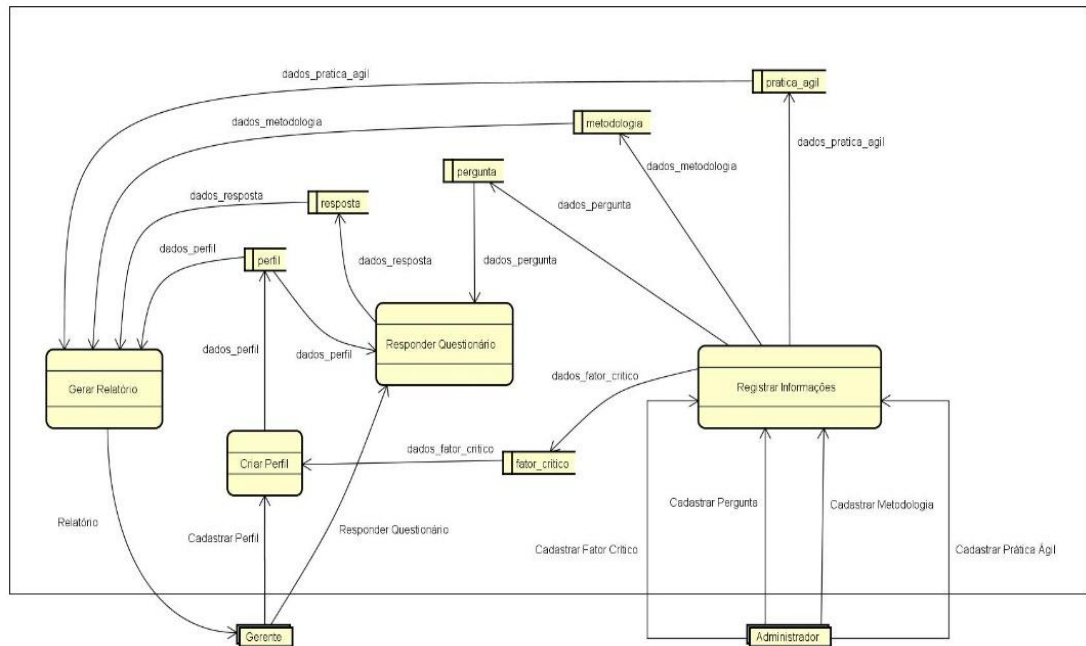


Figura 4. Diagrama de fluxo de dados

A seguir, são apresentadas as principais ações do diagrama de fluxo de dados do sistema GAPRO, na ordem em que devem ser executadas no sistema de apoio.

- **Registrar Informações** – única ação do Administrador no sistema, esta ação é responsável por gerenciar os dados armazenados no banco de dados e relacionar as entidades do sistema: pergunta, grupo_pergunta, fator_critico, pratica_agil e metodologia_agil.
- **Criar Perfil** – primeira ação do Gerente, o usuário criar perfis que serão utilizados para registrar respostas do questionário do sistema.
- **Responder Questionário** – ação que o Gerente realiza com um dos seus perfis criados, responder as perguntas do questionário do sistema. E estas respostas são armazenadas no banco de dados e são relacionadas ao perfil do Gerente.
- **Gerar Relatório** – ação final do ciclo de uso do sistema, onde o Gerente solicita o relatório e o sistema consulta todas as informações armazenadas no banco de dados e executa um algoritmo para gerar o relatório.

5.3.3 Diagrama de Sequência

Um tipo importante de diagramação UML é o diagrama de sequência, um modelo comportamental que dá suporte à visualização da sequência explícita de mensagens entre objetos em uma interação, sendo útil para a compreensão das especificações de um sistema (Dennis *et al.*, 2014).

Foram criados quatro diagramas de sequência que representam o sistema de apoio GAPRO: (1) Diagrama de sequência do Administrador; (2) Diagrama de sequência de gerenciamento de perfis; (3) Diagrama de sequência de respostas ao questionário e (4) Diagrama de sequência de geração de relatório.

Os quatro diagramas de sequência apresentados nesta seção possuem fluxo semelhante, onde um usuário interage com o sistema por meio de uma interface (Vista), os comandos executados são gerenciados por meio de uma instância do Controlador, e as informações geradas são registradas no banco de dados com o auxílio do Modelo. O usuário recebe por fim, um feedback confirmando a ação solicitada. Estes diagramas são detalhados a seguir.

Diagrama de sequência do Administrador – representado pela Figura 6, apresenta todas as funções de um Administrador durante a utilização do sistema, que são funções de gerenciamento de informações do banco de dados e envolvem basicamente operações CRUD.

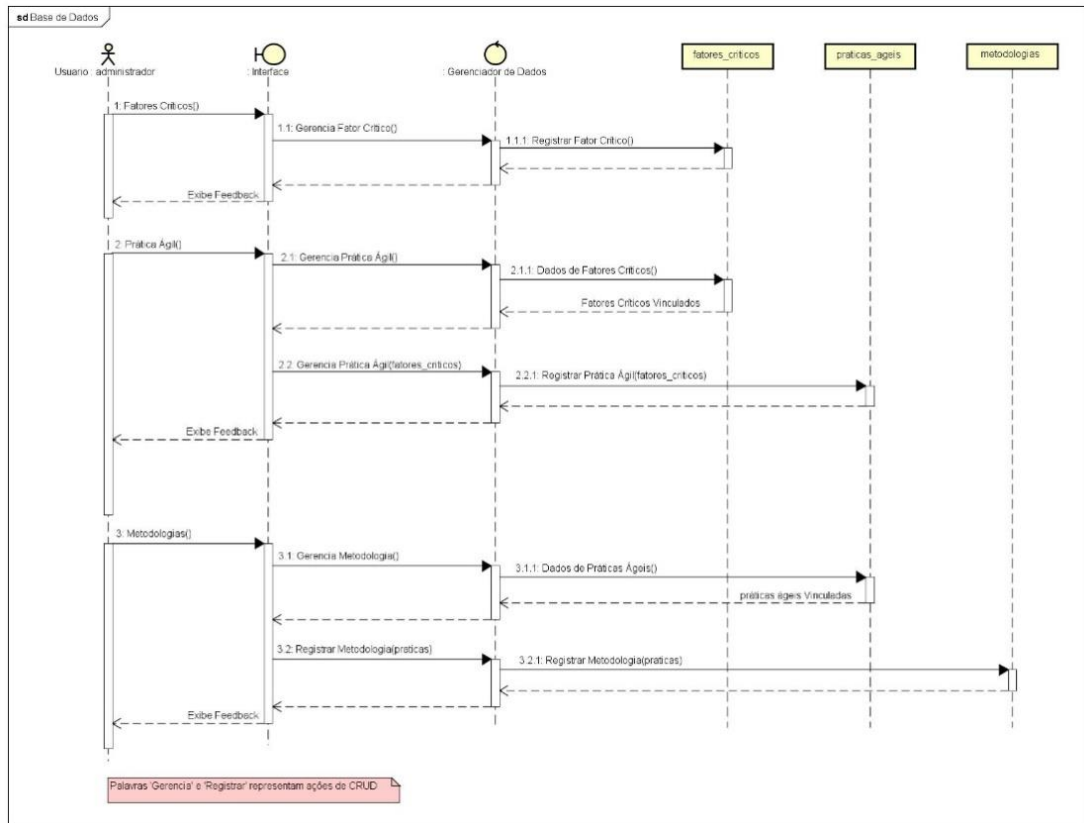


Figura 5. Diagrama de sequência com funções do Administrador no sistema

Os próximos diagramas apresentados são exclusivos de um Gerente do sistema.

Diagrama de sequência de gerenciamento de perfis – Representado pela Figura 7, este diagrama apresenta as operações de gerenciamento de perfis de um Gerente e incluem ações para criar um novo perfil, editar um perfil existente e remover um perfil.

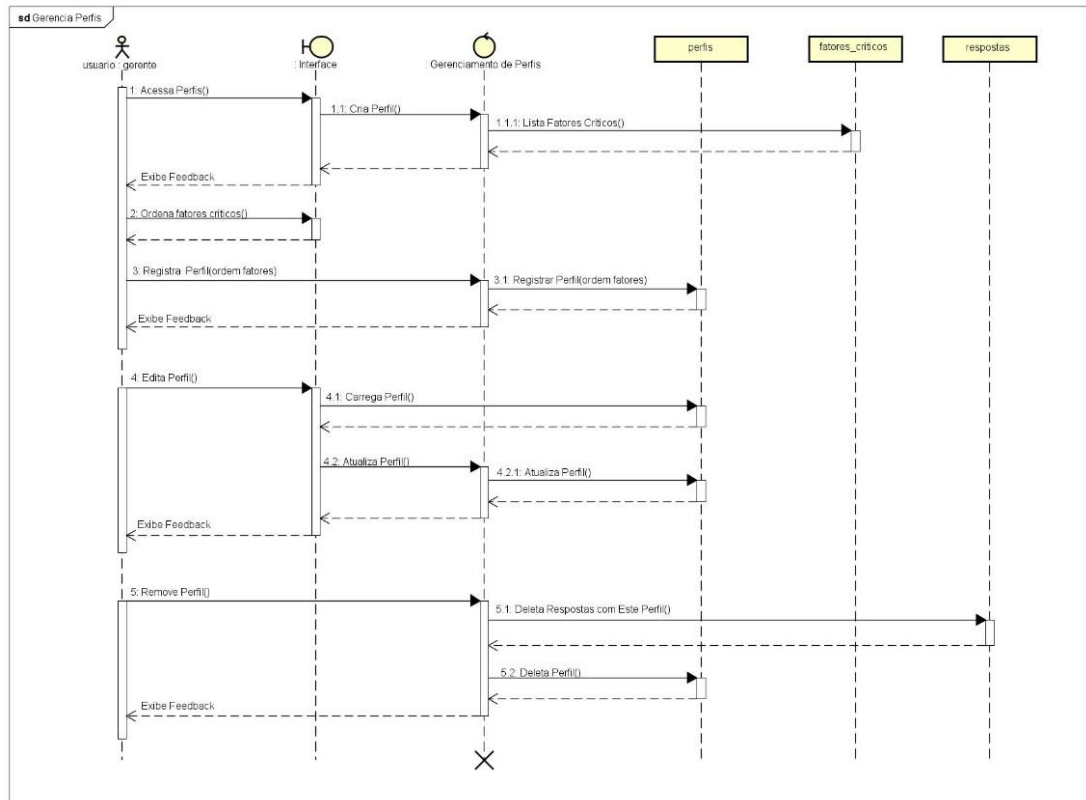


Figura 6. Diagrama de sequência de gerenciamento de perfis

Diagrama de sequência de respostas ao questionário – Representado pela Figura 8, o diagrama demonstra a sequência de ações que um Gerente deve seguir para responder a questões no questionário do sistema.

O gerente acessa a página com o questionário, o controlador verifica o perfil ativo do usuário e carrega as respostas no questionário caso o gerente já tenha respondido a alguma pergunta. Após responder às questões desta página, o gerente pode salvar o questionário e as informações são registradas na tabela de respostas do sistema.

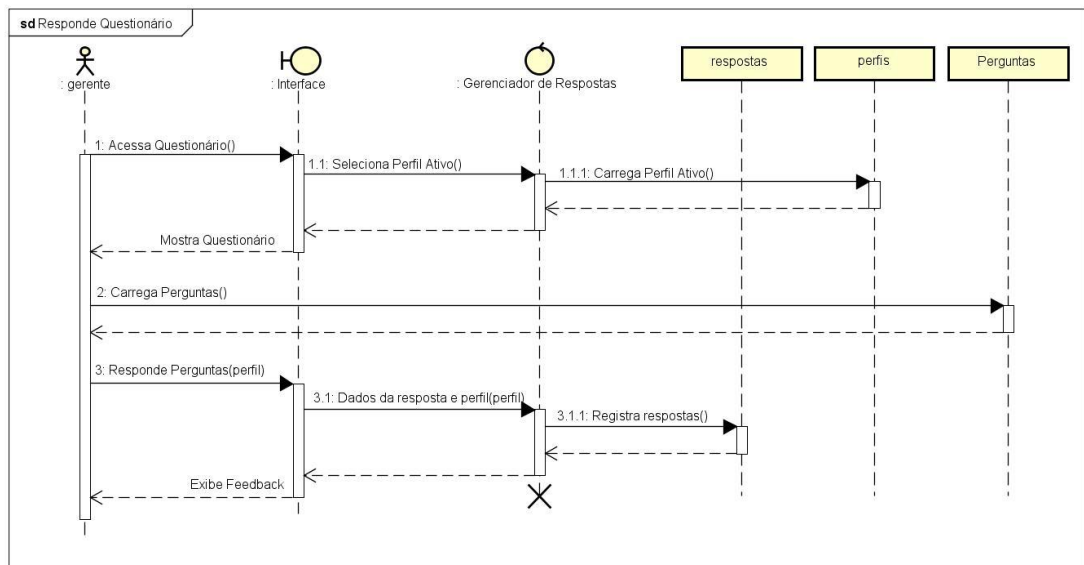


Figura 7. Diagrama de sequência de respostas ao questionário

Diagrama de sequência de geração de relatório – A ação final do sistema GAPRO é gerar um relatório de informações para o Gerente.

Na interface do sistema, uma lista com todos os perfis criados pelo Gerente é apresentada, ao lado de cada perfil há um botão ‘Gerar relatório’ respectivo a este perfil.

Nesta ação, representada pela Figura 9, pode-se observar que o sistema de apoio necessita primeiro carregar diversos dados do banco de dados.

Entre estes dados estão os que foram previamente alimentados pelo administrador do sistema e os dados armazenados das ações de um Gerente, como seu perfil e as respostas do questionário.

A última etapa do ciclo no diagrama é executar o algoritmo do sistema utilizando os dados carregados, transformando estes dados em informações que serão apresentadas no relatório para o gerente.

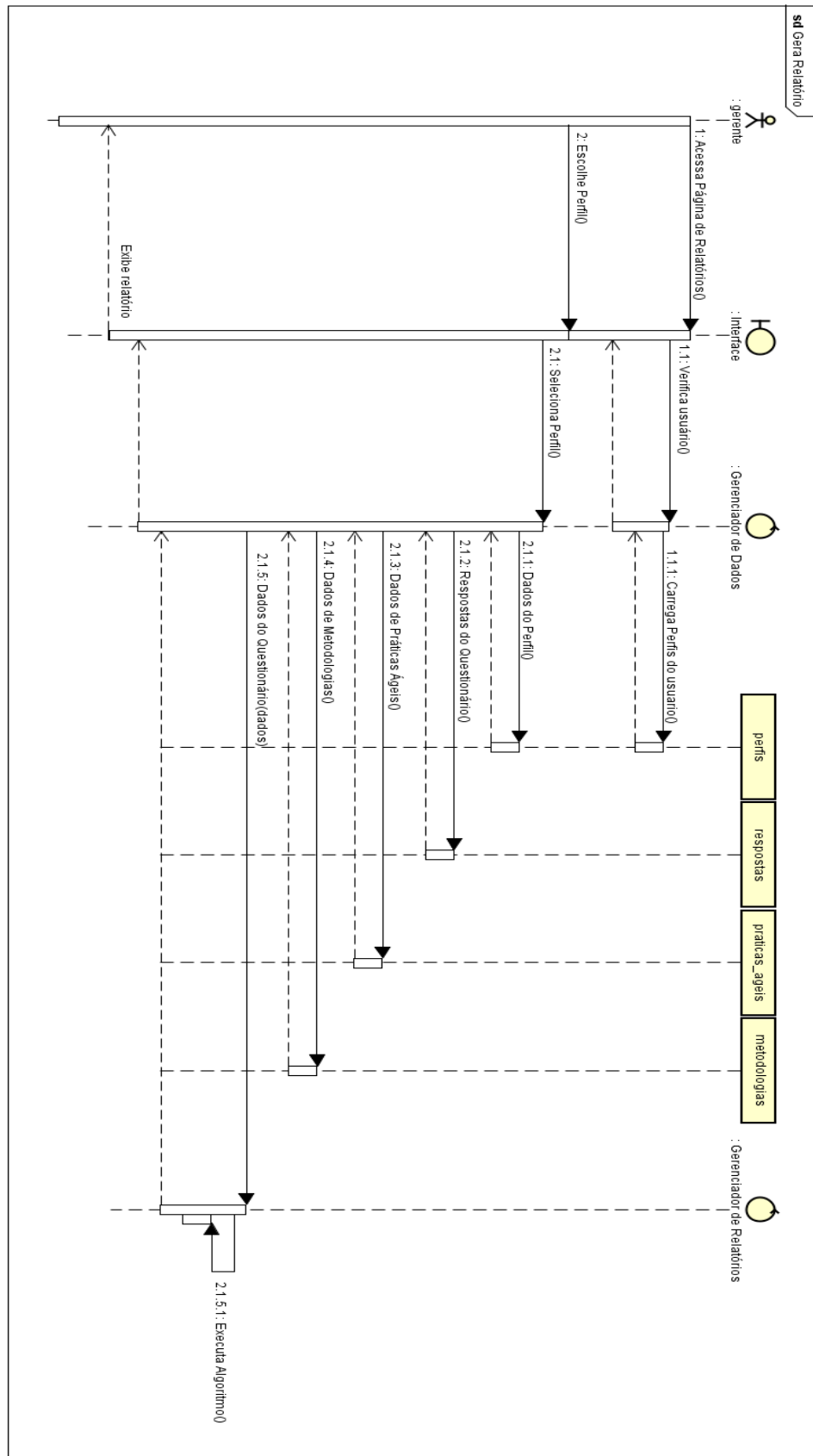
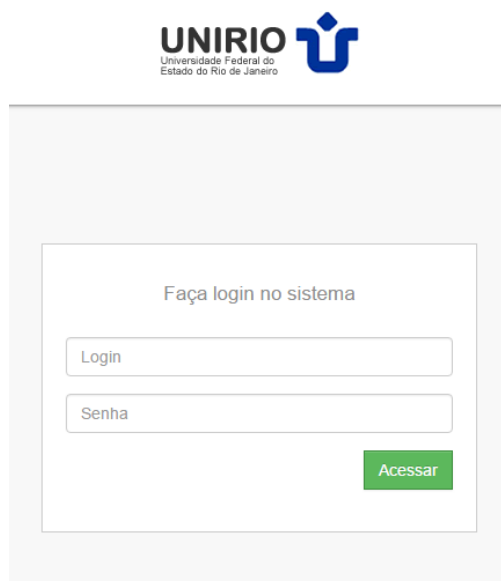


Figura 8. Diagrama de sequência de geração de relatório

5.4 Telas do sistema

Nesta seção, serão apresentadas algumas telas do protótipo. O sistema possui dois grupos de usuários: ‘Administrador’ e ‘Gerente’. Cada grupo de usuário tem uma interface com funcionalidades exclusivas. As telas serão exibidas e descritas de acordo com suas funções.

Administradores e Gerentes iniciam o sistema na tela de login (Figura 10) e acessam sua conta por meio de um nome de usuário e uma senha.



A imagem mostra a interface de login do sistema. No topo, há o logotipo da UNIRIO (Universidade Federal do Estado do Rio de Janeiro) com o nome da instituição e o nome da universidade. Abaixo, há um formulário com o título "Faça login no sistema". O formulário contém dois campos de entrada: "Login" e "Senha". À direita dos campos, há um botão verde com o texto "Acessar".

Figura 9. Tela de login

5.4.1 Administrador

O Administrador é responsável por alimentar o sistema com os dados que serão armazenados no banco de dados e gerenciar os usuários cadastrados. O acesso do Administrador tem como as opções no menu principal: Início, Fatores Críticos, Práticas Ágeis, Metodologias Ágeis, Questionário e Sair.

A Tela de início apresenta as instruções do sistema, explicando cada função disponível, como pode ser visto na Figura 11.

Administração do sistema

O papel do administrador no sistema é alimentar a base de dados com informações e a relação entre os dados. Consulta de estatísticas e gráficos devem ser implementadas em uma versão futura do sistema.

Fatores Críticos

Cadastro dos Fatores Críticos no sistema.

Fatores críticos possuem peso dinâmico e dependem da ordem de fatores definida pelo usuário do sistema em seu perfil.

Práticas Ágeis

Cadastro das Práticas Ágeis no sistema.

Baseado nos resultados de uma survey apresentada na monografia, cada Prática Ágil possui um peso fixo definido, este peso utilizado na fórmula geral do cálculo de pontos do sistema.

Uma Prática Ágil é relacionada a N Fatores Críticos do sistema, esta relação é definida por uma tabela apresentada na monografia.

Metodologias Ágeis

Cadastro das Metodologias Ágeis no sistema.

A lista de Metodologias Ágeis é relacionada a N Práticas Ágeis do sistema, ela será exibida no relatório se pelo menos 1 Prática Ágil recomendada ao usuário pertencer a uma Metodologia Ágil.

Figura 10. Tela de início do Administrador

Na tela de Fatores Críticos, o Administrador fica responsável por cadastrar cada um dos fatores críticos no banco de dados (Figura 12). Alguns padrões importantes são identificados não somente na tela de Fatores Críticos, mas também na tela de Metodologias Ágeis e Questionário. Um desses padrões é a coluna de Ações da lista, onde o Administrador pode editar um item já criado, visualizar os dados cadastrados e também excluir um item. Outro padrão compartilhado entre as telas do sistema é a ação de adicionar um novo dado. Ao clicar no botão ‘+ Adicionar’ acima da lista, o Administrador acessa uma tela onde deve entrar com as informações do novo item, ainda visualizado no exemplo da Figura 12.

Fatores Críticos

+ Adicionar Fator Crítico
Q













Nome	Ações
A familiaridade com a metodologia de desenvolvimento	  
A familiaridade com a tecnologia de desenvolvimento	  
Apoio da alta administração	  
Bom gerenciamento da qualidade	  

Figura 11. Lista de Fatores Críticos

O cadastro de informações no sistema começa pela inclusão de Fatores Críticos, pois outros itens dependem deles. Esta relação pode ser verificada na Figura 4. Na tela onde um novo fator crítico é adicionado, somente é necessário identificar o nome do fator, como pode ser visualizado na Figura 13.

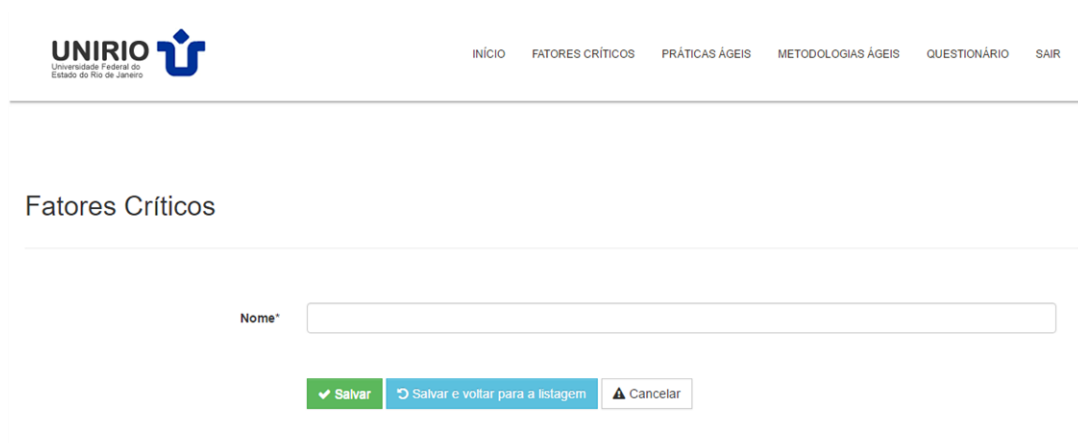


Figura 12. Tela de adição de item

O próximo passo para o Administrador é o cadastro de Práticas Ágeis. Mais informações são necessárias além do nome, como pode ser visto na Figura 14. Uma Prática Ágil tem uma descrição que, posteriormente, será incluída em relatórios gerados pelos Gerentes. Este relatório é a principal informação que o gerente levará em consideração na escolha e aplicação de uma prática ágil.

Práticas Ágeis

Nome* Backlog do Produto

Descrição

O Backlog é uma lista de todas as funcionalidades pensadas para um produto. Inicialmente, o Backlog começa com as necessidades mais básicas, mas geralmente muda com o tempo, de acordo com o aprendizado do time sobre o produto que está desenvolvendo e seus usuários (reais ou potenciais).

Manter uma lista priorizada é importante para o fluxo de produção, para dar mais visibilidade às trocas que acontecem e aos efeitos dessas mudanças. Fazem parte do Backlog tarefas técnicas ou atividades diretamente relacionadas às funcionalidades solicitadas.

Peso* 5

Fatores Críticos

Comunicação eficaz e feedback x Objetivos e metas claros x

Orçamento realista x Planejamento adequado x

Requisitos e especificações claras x

✓ Aplicar alterações Aplicar alterações e voltar para a listagem Cancelar

Figura 13. Adição de Prática Ágil

Outra importante informação é o peso da prática ágil, que é utilizado na Fórmula Geral do sistema. A lista de pesos das práticas ágeis pode ser verificada na Tabela 13 do projeto.

A última informação a ser inserida no formulário é o relacionamento da prática ágil com os fatores críticos (novamente, esta relação é relevante para as funcionalidades disponibilizadas ao Gerente, como será visto posteriormente). A tabela de relações entre práticas ágeis e fatores críticos pode ser verificada na Tabela 4.

Na tela referente ao cadastro de metodologias ágeis da Figura 15, deve ser informado o nome da metodologia, um campo para descrição e a relação de práticas ágeis associadas. A relação entre metodologias e práticas ágeis pode ser verificada na Tabela 5. Como exemplo, dentro da metodologia *SCRUM*, estão presentes práticas ágeis como Backlog do produto, Cliente presente, Pequenas liberações, entre outras.

Metodologias Ágeis

Nome* Scrum

Descrição

Scrum é uma metodologia ágil para gestão e planejamento de projetos e desenvolvimento ágil de software. É utilizado para trabalhos complexos nos quais é impossível prever tudo o que irá ocorrer. O Scrum consiste nos times do Scrum associados a papéis, eventos, artefatos e regras. As regras do Scrum integram os eventos, papéis e artefatos, administrando as relações e interações entre eles.

No Scrum, os projetos são divididos em ciclos (tipicamente mensais) chamados de Sprints. Cada Sprint representa uma iteração dentro da qual um conjunto de atividades deve ser executado.

O Time Scrum é composto pelo Product Owner, o Time de Desenvolvimento e o Scrum Master. O Product Owner, ou dono do produto, é o responsável por maximizar o valor do produto e do trabalho do Time de Desenvolvimento. O Time de Desenvolvimento consiste de profissionais que realizam o trabalho de entregar uma versão usável que potencialmente incrementa o produto "pronto" ao final de cada Sprint. O Scrum Master é responsável por garantir que o Scrum seja entendido e aplicado, e ajuda aqueles que estão fora do Time Scrum a entender quais as suas interações com o Time Scrum são úteis e quais não são. O Scrum Master ajuda todos a mudarem estas interações para maximizar o valor criado pelo Time Scrum.

Práticas Ágeis

Backlog do Produto ✕ Cliente Presente ✕ Equipe Completa ✕

Pequenas Liberações ✕ Reuniões de Planejamento ✕ Reuniões Diárias ✕

Scrum de Scrums ✕ Visibilidade do Projeto ✕

✓ Aplicar alterações ↻ Aplicar alterações e voltar para a listagem ⚠ Cancelar

Figura 14. Adição de Metodologia Ágil

Finalmente, o Administrador pode então cadastrar os problemas que estarão presentes no questionário que deverá ser respondido por um Gerente durante sua sessão de acesso.

Para tornar a interface ainda mais amigável e de fácil utilização, os problemas são divididos em grupos, como apresentado na Figura 16. Deste modo, o Gerente poderá responder a uma sequência de questões com um mesmo tema, tornando suas respostas mais confiáveis por estar focado em um único tema por vez.

Grupos de Pergunta

+ Adicionar Grupo de Pergunta

Q

Nome	Ações
Pessoas	Excluir
Processos	Excluir
Técnicos	Excluir

Exibir 10 entradas

Registros de 1 à 10, num total de 3 registros

⏪ ⏩ 1 ⏪ ⏩

Figura 15. Grupos de perguntas do questionário

Nesta versão do sistema, três grupos foram identificados:

- **Pessoas:** relacionado à gestão de pessoas, como comunicação e quadro dos profissionais envolvidos nos projetos.
- **Técnicos:** relacionado aos aspectos técnicos do projeto, como tecnologias utilizadas e qualidade.
- **Processos:** relacionado aos processos de projetos, como planejamento, metodologias e gerenciamento de riscos.

Com grupos bem definidos, o Administrador pode inserir os problemas no sistema, como mostrado na Figura 17. A relação que acontece nesta tela é entre o problema e os fatores críticos cadastrados. A relação entre problemas e fatores críticos pode ser verificada no Anexo A.

Para exemplificar, ‘Falta de apoio da alta administração’ é um dos problemas apresentados no questionário e é representada pela sua descrição: ‘Nos meus projetos foi verificado que a alta administração não reconheceu a importância dos produtos/serviços que seriam gerados, dessa forma não apoiou a execução dos projetos de forma adequada’. Para este problema, estão relacionados três fatores críticos: (1) Objetivos e metas claros; (2) Apoio da alta administração, e (3) Comunicação eficaz e feedback. Dependendo da ordem em que estes fatores críticos estejam ordenados por um Gerente, este problema poderá ter maior ou menor prioridade de tratamento quando um relatório for gerado.



UNIRIO
Universidade Federal do
Estado do Rio de Janeiro

INÍCIO FATORES CRÍTICOS PRÁTICAS ÁGEIS METODOLOGIAS ÁGEIS QUESTIONÁRIO SAIR

Questionário

Nome*

Descrição

Grupo da Pergunta

Fatores Críticos

Figura 16. Adição de problemas

5.4.2 Gerente

O Gerente é o usuário do sistema GAPRO. Ele irá interagir com a interface do sistema inserindo seus dados para criação de seu perfil. Com base nisso, e em respostas a um questionário, o sistema poderá gerar um relatório com recomendações do uso de práticas ágeis e metodologias ágeis em sua equipe.

Recomenda-se o uso do sistema no início do projeto e que as respostas do questionário sejam baseadas em projetos anteriores. Desta forma, as práticas recomendadas pelo sistema podem ser adotadas causando o menor impacto possível no projeto.

A Tela Principal do protótipo tem como as opções no menu principal: Início, Perfil, Questionário, Relatório e Sair, conforme pode ser visto na Figura 18.

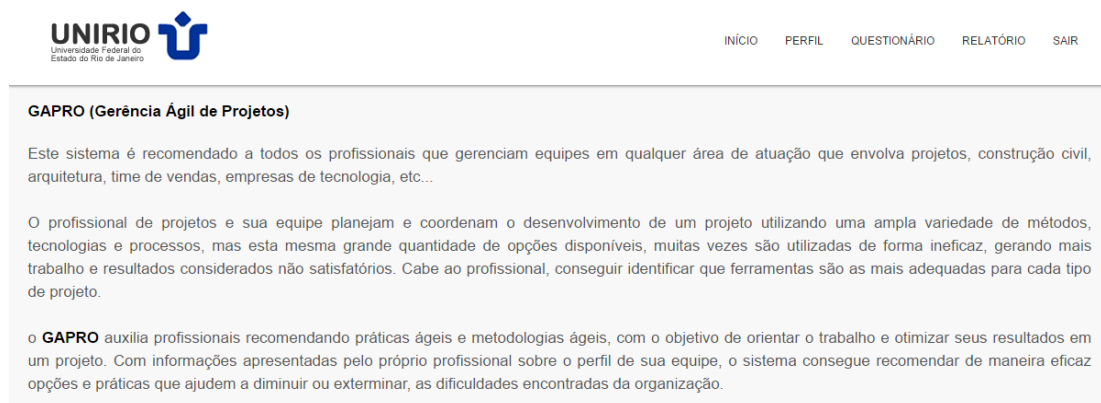


Figura 17. Tela de início do Gerente

Após ler e entender as instruções do sistema, o Gerente pode começar a criar seu perfil. A tela contém informações de ajuda para auxiliar o usuário e lista todos os perfis criados, como mostra a Figura 19.

Meus Perfis

Nesta página você tem acesso a sua lista de perfis da sua conta!

Ações Disponíveis

- ✎ Edição do perfil, mude o nome ou o tamanho da sua equipe
- ☰ Lista de Fatores, a primeira ação que você deve fazer após criar um perfil. ordene os fatores seguindo as orientações na tela
- ☆ Você pode ter um perfil ativo por vez, ao responder ao questionário, as respostas serão salvas em seu perfil ativo
- 🗑 Botão para deletar um perfil, mas lembre-se, todas as informações do questionário e relatório também serão deletadas

[+ Adicionar Perfil](#) Q

Nome	Ativo	Ações
Perfil UNIRIO	★ Perfil Ativo	✎ ☰ ☆ 🗑

Exibir entradas Registros de 1 à 10, num total de 1 registros ⏪ < 1 > ⏩

Figura 18. Tela de perfil do Gerente

A criação de um perfil é feita ao clicar no botão ‘+ Adicionar Perfil’ (Figura 19). Nesta tela, representada pela Figura 20, o gerente define um nome único para seu perfil e o tamanho da sua equipe de desenvolvedores. Um Gerente pode ter quantos perfis achar necessário, pode ter por exemplo, um perfil para cada equipe que coordena.

Meus Perfis

Informe os dados do seu novo perfil. Ele será automaticamente definido como seu perfil ativo

Nome*

Tamanho da equipe*

[✓ Salvar](#) [↺ Salvar e voltar para a listagem](#) [⚠ Cancelar](#)

Figura 19. Tela de adição de perfil

O segundo passo é definir a prioridade de fatores que considera como mais críticos em um projeto. Seguindo as instruções presentes na Figura 19, o usuário terá acesso à tela onde essa definição de prioridades é feita.

Nesta tela deve ser feita a ordenação da lista de fatores, como apresentado na Figura 21. A ordenação é feita arrastando os ícones para sua nova posição. A cada mudança da ordem de um fator, os dados são atualizados no banco de dados de forma automática.

Seguindo as instruções na tela da Figura 21, o usuário deve deixar no topo os fatores que considera como mais importantes em um projeto ou organização.

A imagem mostra a interface web 'Perfil UNIRIO'. No topo, há o logo da UNIRIO e um menu de navegação com links: INÍCIO, PERFIL, QUESTIONÁRIO, RELATÓRIO e SAIR. Abaixo do menu, o título 'Perfil UNIRIO' é exibido em vermelho. À direita, há um botão 'Meu Perfil Ativo' e o texto 'Perfil UNIRIO - 30 pessoas'. A seção principal, intitulada 'Definindo um perfil', contém instruções sobre como definir a prioridade dos fatores, um exemplo de como classificar os fatores e uma dica sobre a salvagem automática. Abaixo, há uma lista de fatores críticos, cada um com um ícone de setas para cima e para baixo para facilitar a ordenação por arrastar e soltar.

Fator
Usuário / envolvimento do cliente
Requisitos e especificações claras
Objetivos e metas claros
Prazos realistas
Apoio da alta administração

Figura 20. Definição de prioridades de fatores críticos

Com um perfil criado e sua prioridade de fatores críticos definidos, o Gerente tem, então, acesso liberado para responder as questões do questionário.

Caso o Gerente não tenha criado um perfil ou definido as prioridades de fatores críticos, uma mensagem será exibida informando ao usuário sobre as pendências que deve tratar antes de iniciar o questionário, conforme pode ser visto na Figura 22.

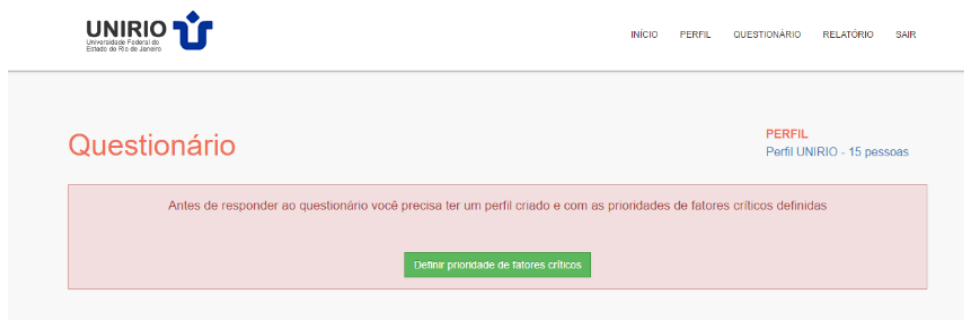


Figura 21. Tela de alerta do sistema

Com todas as informações corretamente cadastradas, o Gerente tem acesso ao questionário. Mais uma vez está presente na tela uma lista de instruções de utilização, para auxiliar o usuário na navegação (Figura 23).



Figura 22. Tela de instruções do questionário

Para cada grupo de perguntas cadastrados pelo Administrador, o sistema apresenta uma aba, onde exibirá as questões do tema. As listas de problemas utilizados nas questões desta versão do sistema estão presentes no Anexo A.

Verificando a Figura 24, temos a lista de questões de um dos temas. Para cada questão, o usuário tem acesso a uma descrição da questão que auxilia sua resposta: a

descrição pode ser acessada ao pausar sobre o ícone ‘?’, que se encontra do lado esquerdo da questão. Ao lado direito da questão, 5 estrelas apresentam as opções possíveis de resposta do Gerente. A classificação de estrelas é levada em consideração na Fórmula Geral do sistema, detalhada na próxima seção.

As estrelas são utilizadas para classificar a relevância de uma questão para o Gerente. Como ilustração, para o caso onde o Gerente classifique uma questão com a opção máxima de 5 estrelas, ela receberá um peso maior e uma determinada sequência de práticas ágeis será recomendada com maior foco na resolução dessa questão.

O Gerente, ao classificar uma questão com menos estrelas, consequentemente, apresenta uma sequência de práticas ágeis recomendadas diferente. Essa sequência continua possibilitando que a questão seja resolvida individualmente, mas pode não ser uma escolha relevante dependendo de que parte do relatório final o Gerente pretenda consultar, pois o relatório final do GAPRO apresenta diferentes modos de visualização do relatório.

Caso o Gerente entenda que uma questão não representa um problema identificado em sua equipe ou que a questão não é relevante para o perfil criado por ele, o Gerente manterá a quantidade padrão de estrelas, que é representado pela estrela preta na tela (Figura 24).

UNIRIO
Universidade Federal do
Estado do Rio de Janeiro

INÍCIO PERFIL QUESTIONÁRIO RELATÓRIO SAIR

Questionário

Meu Perfil Ativo
Perfil UNIRIO - 30 pessoas

Instruções Pessoas Técnicos Processos

Salvar Respostas

① Falta de comunicação entre a equipe ★ ★ ★ ★ ★

① Falta de apoio da alta administração ★ ★ ★ ★ ★

① Falta de envolvimento do cliente ★ ★ ★ ★ ★

Figura 23. Lista de questões em um grupo do questionário

Após a classificação das questões do questionário pelo Gerente, o sistema de apoio GAPRO tem neste momento todas as informações necessárias para que se possa avaliar as necessidades do Gerente. Estas informações serão utilizadas no momento em que o Gerente gerar o relatório final do sistema.

As informações são:

- Um perfil com fatores críticos de sucesso ordenados por sua prioridade em um projeto ou equipe;
- Pesos de questões, definidas ao escolher a quantidade de estrelas para uma questão durante o preenchimento do questionário;
- Pesos de práticas ágeis previamente definidos no sistema pelo administrador, e que será detalhada no próximo capítulo.

Na tela de relatório do sistema, representada pela Figura 25, o Gerente tem a lista de seus perfis criados no sistema de apoio GAPRO. Junto a cada perfil criado pelo Gerente, existe a ação ‘Gerar Relatório’.

The screenshot shows the 'Relatórios de Análise' (Analysis Reports) page of the UNIRIO system. At the top, there is a navigation bar with links: INÍCIO, PERFIL, QUESTIONÁRIO, RELATÓRIO, and SAIR. Below the navigation bar, the page title 'Relatórios de Análise' is displayed. A light blue informational box contains text about creating a profile and generating reports. Below this, there is a search bar with a magnifying glass icon. The main content is a table with two columns: 'Nome' and 'Ação'. The table has one row with the name 'Perfil UNIRIO' and a button labeled 'Gerar Relatório'. At the bottom of the page, there are pagination controls showing 'Exibir 10 entradas' and 'Registros de 1 à 10, num total de 1 registros'.

Figura 24. Lista de perfis de um Gerente na tela de relatórios

Com as informações corretamente inseridas, o Gerente poderá gerar o relatório de recomendação de práticas. Ao gerar um relatório, o sistema apresenta em uma nova tela o resultado da análise do perfil e respostas do usuário (Figura 26).

O relatório é dividido em 3 seções por meio de abas, onde cada seção apresenta uma visão diferente de recomendações aos Gerentes, englobando práticas e

metodologias ágeis. A seguir, cada uma dessas seções será apresentada: Práticas ágeis por questão, Ranking de práticas ágeis e Metodologias Ágeis.

- **Práticas ágeis por questão** - Nesta seção, apresentada na Figura 26, para cada uma das questões classificadas com estrelas pelo Gerente durante o preenchimento do questionário, o sistema apresenta uma lista de práticas ágeis que podem ser aplicadas para ajudar a melhorar ou resolver os problemas relacionados à questão.
- **Ranking de práticas ágeis** - A segunda aba do relatório (Figura 27), apresenta uma visão mais compacta e objetiva de recomendações de práticas ágeis. Enquanto na aba anterior o foco é o tratamento de cada questão individualmente, nessa aba temos uma lista que apresenta a quantidade de vezes que cada prática ágil foi recomendada para utilização pelo Gerente. Ainda na Figura 27, pode-se observar que a prática ágil ‘*Backlog* do produto’ possui 8 recomendações.

O objetivo nesta seção, é permitir que um Gerente possa, ao invés de aplicar uma prática ágil para cada questão apresentada, visualizar uma lista única de recomendações de práticas ágeis. Esse conjunto de práticas ágeis poderá auxiliar nas questões analisadas de uma forma conjunta, pois pelo grande número de recomendações de uma mesma prática ágil no relatório, pode-se admitir que a adoção de tal prática ágil é benéfica para todo um projeto, e não somente para uma questão específica.


- **Metodologias Ágeis** - A última aba do relatório segue uma lógica parecida com a aba anterior. Práticas ágeis são recomendadas para utilização visando beneficiar todo um projeto, e não focadas em tratar uma questão isolada, como a visão apresentada na seção ‘Práticas ágeis por Questão’.

O objetivo desta seção é apresentar ao Gerente as práticas ágeis que fazem parte de um escopo mais amplo, que são as metodologias ágeis, já apresentadas no capítulo anterior.

Com a apresentação de uma metodologia ágil, espera-se que um Gerente com pouca experiência possa entender que algumas práticas ágeis podem fazer parte de um conjunto já definido de práticas e regras, que são as metodologias ágeis. Deste modo, além do relatório do sistema, um Gerente pode procurar conhecer

novas opções e alternativas de práticas ágeis que ainda não fazem parte do banco de dados do sistema GAPRO.

Nesta seção, representada pela Figura 28, as metodologias ágeis cadastradas no sistema são apresentadas. Dentro de cada uma dessas metodologias ágeis, o Gerente poderá encontrar uma lista de práticas ágeis recomendadas para seu perfil.



UNIRIO
Universidade Federal do
Estado do Rio de Janeiro

INICIO

PERFIL

QUESTIONÁRIO

RELATÓRIO

SAIR

Relatório Geral

PERFIL
Projeto X Equipe E - 9 pessoas

Objetivo do Relatório

Auxiliar o profissional responsável pela gerência de projetos, na escolha de práticas ágeis para adoção em sua equipe, o relatório será baseado nas questões classificadas pelo profissional durante o questionário, a ordem de fatores definidos em seu perfil e informações estatísticas do próprio sistema. O relatório tem o objetivo de apresentar as práticas recomendadas afim de amenizar e/ou resolver o problema identificado.

Práticas Ágeis Por Questão

Ranking de Práticas Ágeis

Metodologias Ágeis

Imprimir Relatório

Para cada pergunta do questionário que você classificou com pelo menos 2 estrelas, o sistema recomenda que práticas ágeis você pode utilizar para melhorar a questão em sua equipe. Clique em cada pergunta para verificar as recomendações.

Abriu quadros

Falta de comunicação entre a equipe

Exemplo de problema

Nos meus projetos foi verificado que a equipe não se comunica o suficiente. Os membros estão frequentemente desinformados sobre o que seus companheiros de equipe estão fazendo, resultando em problemas como perda de tempo de projeto, desenvolvimento de código já existente e retrabalho.

Prática Ágil Recomendada	Descrição
Backlog do Produto (800)	<p>O Backlog é uma lista de todas as funcionalidades pensadas para um produto. Inicialmente, o Backlog começa com as necessidades mais básicas, mas geralmente muda com o tempo, de acordo com o aprendizado do time sobre o produto que está desenvolvendo e seus usuários (reais ou potenciais).</p> <p>Manter uma lista priorizada é importante para o fluxo de produção, para dar mais visibilidade às trocas que acontecem e aos efeitos dessas mudanças. Fazem parte do Backlog tarefas técnicas ou atividades diretamente relacionadas às funcionalidades solicitadas.</p>
Visibilidade do Projeto (700)	<p>Projetos ágeis por sua natureza estão continuamente mudando (planos, modelos, código e demais artefatos). O objetivo da prática de Visibilidade do Projeto é manter e monitorar a qualquer tempo as medições do progresso do projeto. As informações devem estar acessíveis para todos os envolvidos no projeto.</p> <p>Pode ser criado um painel na web para manter a qualquer tempo o status e as métricas relacionadas com o progresso do projeto. Múltiplos painéis podem ser usados para disponibilizar diferentes tipos de informação que atendam a todos os níveis organizacionais necessários.</p> <p>Para esta prática são utilizadas ferramentas de apoio como quadro de tarefas do Scrum ou Kanban, Gráfico Burndown e Histórias de Usuários.</p>

Figura 25. Aba do relatório com práticas ágeis que tratam cada questão

Relatório Geral

PERFIL
Perfil UNIRIO - 30 pessoas

Objetivo do Relatório

Auxiliar o profissional responsável pela gerência de projetos, na escolha de práticas ágeis para adoção em sua equipe, o relatório será baseado nas questões classificadas pelo profissional durante o questionário, a ordem de fatores definidos em seu perfil e informações estatísticas do próprio sistema. O relatório tem o objetivo de apresentar as práticas recomendadas afim de amenizar e/ou resolver o problema identificado.

Práticas Ágeis Por Questão

Ranking de Práticas Ágeis

Metodologias Ágeis

Imprimir Relatório

A lista abaixo é um complemento da aba 'Práticas Ágeis Por Questão'. A lista apresenta todas as práticas ágeis recomendadas e a quantidade de vezes que uma prática foi recomendada após analisar todas as suas questões respondidas no questionário. Você pode utilizar esta lista para criar um ambiente de projeto com um menor número de práticas que podem resolver o maior número de problemas.

Prática Ágil Recomendada	Descrição	Pontos
Backlog do Produto	<p>O Backlog é uma lista de todas as funcionalidades pensadas para um produto. Inicialmente, o Backlog começa com as necessidades mais básicas, mas geralmente muda com o tempo, de acordo com o aprendizado do time sobre o produto que está desenvolvendo e seus usuários (reais ou potenciais).</p> <p>Manter uma lista priorizada é importante para o fluxo de produção, para dar mais visibilidade às trocas que acontecem e aos efeitos dessas mudanças. Fazem parte do Backlog tarefas técnicas ou atividades diretamente relacionadas às funcionalidades solicitadas.</p>	8

Figura 26. Aba do relatório com as práticas ágeis mais recomendadas

Relatório Geral

PERFIL
Projeto X Equipe E - 9 pessoas

Objetivo do Relatório

Auxiliar o profissional responsável pela gerência de projetos, na escolha de práticas ágeis para adoção em sua equipe, o relatório será baseado nas questões classificadas pelo profissional durante o questionário, a ordem de fatores definidos em seu perfil e informações estatísticas do próprio sistema. O relatório tem o objetivo de apresentar as práticas recomendadas afim de amenizar e/ou resolver o problema identificado.

Práticas Ágeis Por Questão

Ranking de Práticas Ágeis

Metodologias Ágeis

Imprimir Relatório

Uma prática ágil faz parte de uma metodologia Ágil, que é um conjunto de práticas e regras que juntas podem ser aplicadas em um projeto. O quadro abaixo apresenta dentro de cada metodologia ágil cadastrada no sistema, uma lista com as práticas ágeis que o sistema recomendou para seu perfil. Você pode utilizar estas informações para verificar a metodologia ágil que possui mais práticas recomendadas e verificar que outras práticas dentro desta mesma metodologia você também poderia adotar em seus projetos. Algumas vezes seguir uma única metodologia pode ser o melhor caminho em um projeto.

Metodologia Ágil	Descrição
Scrum	<p>Scrum é uma metodologia ágil para gestão e planejamento de projetos e desenvolvimento ágil de software. É utilizado para trabalhos complexos nos quais é impossível prever tudo o que irá ocorrer. O Scrum consiste nos times do Scrum associados a papéis, eventos, artefatos e regras. As regras do Scrum integram os eventos, papéis e artefatos, administrando as relações e interações entre eles.</p> <p>No Scrum, os projetos são divididos em ciclos (tipicamente mensais) chamados de Sprints. Cada Sprint representa uma iteração dentro da qual um conjunto de atividades deve ser executado.</p> <p>O Time Scrum é composto pelo Product Owner, o Time de Desenvolvimento e o Scrum Master. O Product Owner, ou dono do produto, é o responsável por maximizar o valor do produto e do trabalho do Time de Desenvolvimento. O Time de Desenvolvimento consiste de profissionais que realizam o trabalho de entregar uma versão usável que potencialmente incrementa o produto "pronto" ao final de cada Sprint. O Scrum Master é responsável por garantir que o Scrum seja entendido e aplicado, e ajuda aqueles que estão fora do Time Scrum a entender quais as suas interações com o Time Scrum são úteis e quais não são. O Scrum Master ajuda todos a mudarem estas interações para maximizar o valor criado pelo Time Scrum.</p> <p>Práticas recomendadas que fazem parte desta metodologia</p> <p>Backlog do Produto</p>

Figura 27. Aba do relatório com uma lista de metodologias ágeis recomendadas

5.5 Fórmula Geral de classificação de Práticas Ágeis

Após a utilização de todo o ciclo do sistema de apoio, envolvendo desde o Administrador alimentando o banco de dados até o Gerente criando seu perfil e respondendo ao questionário, as informações necessárias para a geração do relatório de recomendação de práticas terão sido coletadas.

Para a última etapa de uso do sistema GAPRO, que é gerar o relatório representado na Figura 25, as informações coletadas serão utilizadas por um algoritmo que analisará estas informações coletadas por meio de cálculos e apresentará recomendações de uso de práticas e metodologias ágeis para aplicação em sua equipe.

Antes de apresentar a fórmula, no entanto, é necessário conhecer os tipos de pesos que foram utilizados na fórmula: (1) Pesos de Práticas Ágeis; (2) Pesos de Fatores Críticos e (3) Pesos de Respostas do Questionário.

5.5.1 Peso de Práticas Ágeis

O peso de uma prática ágil é um valor fixo do sistema, e é inserido no sistema pelo Administrador. Algumas etapas foram necessárias para a definição do peso de cada prática ágil.

Etapa 1 - A fonte principal para esta etapa foi o trabalho de Mello *et al.* (2014). Os autores, por meio de um *survey* e com o apoio de uma amostragem estratificada¹⁰, definiram grupos de usuários, onde cada grupo possui características semelhantes.

De acordo com os autores, o *survey* em questão teve o propósito de caracterizar a pertinência e a relevância de características e práticas para a definição de um processo de software como ágil sob o ponto de vista de profissionais de engenharia de software, no contexto dos projetos de software que adotam abordagens de desenvolvimento ágil.

Em nosso projeto, será utilizado como base o extrato S1 do *survey*. Este extrato é composto por sete grupos do *LinkedIn*¹¹, que promovem debates e disseminam

¹⁰ Amostragem em que a população está dividida em estratos ou grupos diferenciados

¹¹ Rede social de negócios utilizada por profissionais

conceitos referentes à agilidade no processo de software, tais como gerência de projetos “ágil” (Mello *et al.*, 2014). O resultado do survey é apresentado na Tabela 9.

Tabela 9. Extratos com taxas de relevância referentes às práticas ágeis.

Prática	S1	S2	S3	S4	S5
Integração Contínua	83%	82%	78%	81%	84%
Backlog do Produto	83%	77%	72%	78%	84%
Visibilidade do Projeto	77%	71%	65%	79%	59%
Pequenas liberações	76%	71%	76%	63%	71%
Equipe Completa	74%	73%	57%	71%	55%
Propriedade Coletiva de Código	72%	73%	60%	49%	66%
Ritmo Sustentável	72%	71%	75%	76%	58%
Reuniões Diárias	71%	68%	65%	70%	59%
Refatoração	67%	72%	62%	47%	70%
Design Simples	60%	68%	66%	47%	58%
Test Driven Development	53%	69%	53%	45%	45%
Padrões de Código	53%	62%	52%	43%	46%
Jogo do Planejamento	42%	48%	37%	-	38%
Cliente Presente	38%	44%	-	-	35%

Fonte: (Mello *et al.*, 2014)

As práticas Reunião de Planejamento, Scrum de Scrums, Desenvolvimento Orientado por Comportamento e Programação em Pares fazem parte do escopo desta monografia, porém não foram contempladas no *survey* em questão. Para estas práticas foi atribuída a taxa de relevância de 38%, mais baixa dentre as taxas do *survey*.

Etapa 2 - Com um grupo de valores definidos no extrato S1, como apresentado na Tabela 9, a próxima etapa consiste em definir um peso para cada uma das práticas ágeis da lista. Para esta etapa, optou-se pela criação de uma Tabela de Distribuição de Frequência¹². Deste modo, foi possível definir intervalos na tabela, e para cada intervalo, um peso para o conjunto de práticas ágeis.

¹² analisando um conjunto de dados, começa-se por considerar as diferentes categorias ou classes, e para cada uma delas calcula-se a sua frequência absoluta, obtendo-se a distribuição de frequências do conjunto de dados. Esta distribuição de frequências é representada na forma de uma tabela (Martins, 2012).

Tabela 10. Primeira parte para montagem de uma Tabela de Frequência

# Dados	Amplitude	Intervalo	Intervalo aproximado	Tamanho do intervalo
18	45	4,2426407	5	10,60660172

Na Tabela 10, temos as primeiras informações necessárias para a montagem da tabela de frequência, existem 18 práticas ágeis, 14 presentes no extrato S1 da Tabela 10 e 4 foram incluídas como detalhado ao fim da etapa 1 desta seção.

De acordo com as porcentagens do extrato S1 da Tabela 9, existe uma amplitude de valores igual a 45, resultado da diferença entre a maior porcentagem e a menor porcentagem da coluna do extrato S1.

Para que se possa encontrar a quantidade de pesos que as práticas ágeis podem ter, verifica-se os intervalos existentes no extrato, para isso, aplica-se a raiz quadrada no número de práticas ágeis (18), encontrado o intervalo 4,2426407. Como não é possível ter 4,2426407 grupos de pesos, utilizamos como resultado um intervalo aproximado, o valor inteiro logo acima. Portanto deve haver 5 grupos de pesos para as práticas ágeis.

O próximo passo é definir a cada qual dos 5 grupos de peso uma determinada prática ágil pertence. Para isso é necessário saber o tamanho do intervalo dos valores.

Na Tabela 10, divide-se a amplitude pelo intervalo encontrado. Deste modo temos como resultado para o tamanho do intervalo o valor 10,60660172.

Com as informações obtidas até o momento, já é possível montar a tabela de frequência para cada intervalo de dados, representada pela Tabela 11.

Para a definição dos valores que separam cada grupo de pesos, soma-se cada linha da coluna Início dos Dados ao tamanho do intervalo calculado anteriormente. Com esta soma temos os resultados apresentados na coluna Final dos Dados na Tabela 11.

Com estas únicas duas colunas já é possível montar a tabela de frequência. As outras colunas da Tabela 11 são utilizadas para confirmar que todos os cálculos realizados estão corretos.

Somando toda a coluna de Frequência, é necessário ter como resultado o número de práticas ágeis utilizadas no cálculo, no caso, o valor 18. A frequência de

cada linha representa a quantidade de práticas ágeis que tem sua taxa de relevância da Tabela 11, dentro do intervalo ‘Final dos Dados’ – ‘Início dos Dados’.

Para a Frequência Relativa Fr, divide-se uma Frequência Fi pelo número total de práticas ágeis, 18. A soma das frequências relativas deve ser sempre igual a 100.

Na linha ‘Totais’ da Tabela 11, pode-se confirmar os valores 18 e 100 respectivamente nas colunas de Frequência Fi e Freq. Relativa Fr.

Tabela 11. Tabela de Distribuição de Frequência

Início dos Dados	Final dos Dados	Média Xi	Frequência Fi	Freq. Relativa Fr
38	48,60660172	43,30330086	6	33,33333333
48,60660172	59,21320344	53,90990258	2	11,11111111
59,21320344	69,81980515	64,51650429	2	11,11111111
69,81980515	80,42640687	75,12310601	6	33,33333333
80,42640687	91,03300859	85,72970773	2	11,11111111
Totais			18	100

Com os intervalos definidos, os pesos foram adicionados a cada intervalo, iniciando com um peso igual a 1 e incrementando a cada intervalo da tabela. A inserção dos pesos começa na base da tabela até chegar ao intervalo do topo. Ao final deste processo, temos intervalos de pesos 1 até 5. Por fim, cada prática ágil tem agora seu respectivo peso fixo. A tabela final com cada prática ágil e seu respectivo peso pode ser vista na Tabela 12.

Tabela 12. Pesos de Práticas do sistema de apoio

PRÁTICAS ÁGEIS	%	MARGEM	PESO
Integração Contínua	83	81,65 - 92,57	5
Backlog do Produto	83		
Visibilidade do Projeto	77	70,74 - 81,65	4
Pequenas Liberações	76		
Equipe Completa	74		
Ritmo Sustentável	72		
Propriedade Coletiva do Código	72		
Reuniões Diárias	71		
Refatoração	67	59,82 - 70,74	3
Design Simples	60		
Desenvolvimento Orientado por Testes	53	49,81 - 59,82	2
Padronização de Código	53		
Jogos de Planejamento	42	38,00 - 48,91	1
Cliente Presente	38		
Desenvolvimento Orientado a comportamento	38		
Programação em Pares	38		
Scrum de Scrums	38		
Reuniões de Planejamento	38		

5.5.2 Peso de fatores críticos

O peso de fatores críticos é um valor dinâmico, relativo à ordenação de fatores críticos que é realizada pelo gerente durante a criação de seu perfil. As variáveis presentes no algoritmo que calcula o peso de cada fator crítico são: (1) Base; (2) nFatores e (3) posicaoRanking.

A variável **Base** é um valor fixo que independe da quantidade de fatores críticos cadastrados no sistema ou de qualquer outra informação, Esta variável é uma variável de controle e tem valor sempre igual a 100.

A variável **nFatores** representa a quantidade atual de fatores críticos cadastrados no banco de dados do sistema de apoio. Esta variável consulta o valor atual no banco sempre que um relatório é gerado.

posicaoRanking é um valor dinâmico e diferente para cada fator crítico dentro de cada perfil criado por um Gerente no sistema. Após criar um perfil, o Gerente deve ordenar os fatores críticos cadastrados no sistema por ordem de prioridade, formando então um ranking de posições de fatores críticos, como detalhado no início do capítulo.

Podemos definir *posicaoRanking*, como a posição atual de um fator crítico dentro da ordenação de fatores críticos em um perfil de Gerente.

Com as variáveis definidas, apresentamos a primeira equação do algoritmo, com o cálculo do peso de um fator crítico no sistema:

$$\text{Peso} = (\text{Base} / \text{nFatores}) * ((\text{nFatores} - \text{posicaoRanking}) + 1)$$

O valor 1 é utilizado na equação para garantir que a equação não tenha como resultado o valor 0.

5.5.3 Peso de respostas do questionário

Outro valor presente no algoritmo do sistema de apoio é o peso que uma questão respondida pelo Gerente no questionário possui. A classificação de uma questão no questionário é feita definindo um número de estrelas como resposta, como apresentado na Figura 24. A cada questão do questionário que um Gerente opte por responder, é atribuído um peso e o valor é salvo no banco de dados.

Os pesos de respostas do questionário são simples e podem ser conferidos na Tabela 13.

Tabela 13. Pesos de questões do questionário

RESPOSTA	PESO
1 ESTRELA	Questão Não Respondida
2 ESTRELAS	1
3 ESTRELAS	2
4 ESTRELAS	3
5 ESTRELAS	3

Observando a Tabela 13, algumas análises podem ser feitas. O questionário já é iniciado com uma estrela selecionada para todas as questões por padrão, mas possuem uma cor diferente para facilitar sua identificação. Isso indica que o Gerente não respondeu à questão, logo, ela não é armazenada no banco de dados e a questão é ignorada durante a criação do relatório.

Quando o gerente classifica uma questão com 2 estrelas, o menor valor de resposta possível, atribui-se peso igual a 1 na questão. Selecionando 3 estrelas, entende-se que o Gerente marcou a opção do meio e a resposta é classificada com peso

igual a 2. Quando um Gerente marca 4 ou 5 estrelas em uma questão, o sistema reconhece que há uma necessidade maior de tratamento para tal questão, por isso, ambas respostas são classificadas com peso igual a 3.

5.5.4 Fórmula Geral do Sistema

Nas seções anteriores, foram apresentados os pesos que são utilizados para compor a Fórmula Geral do Sistema. Com a dinâmica dos pesos dos fatores críticos definidos no perfil de um gerente, pesos para cada questão do questionário dependentes da resposta do gerente; e pesos para as práticas definidas por uma base de pesquisa confiável e alinhada a fórmulas estatísticas de frequência, temos informação suficiente e conseguimos definir pesos para cada item de avaliação do questionário. Assim, podemos definir a fórmula Geral do questionário aplicado a cada problema com a equação abaixo:

$$\text{Pontos} = \text{PFC} * (\text{PPA} + \text{PRQ})$$

Onde **PFC** é igual ao peso do Fator Crítico, **PPA** é o peso da Prática Ágil e **PRQ** o peso da Resposta do Questionário. A equação é executada para cada uma das questões respondidas no questionário pelo Gerente. O resultado da equação é armazenado em uma variável **Pontos** e relacionado a questão que utilizou a fórmula.

O peso do fator crítico é considerado o mais relevante, possuindo maior peso na fórmula geral do sistema, já que é multiplicado pela soma dos outros dois pesos.

Como todas as questões obrigatoriamente utilizam esta fórmula, o sistema GAPRO consegue definir recomendações de práticas ágeis para cada questão respondida pelo gerente individualmente, e as mesmas informações são utilizadas para gerar as outras seções do relatório, como detalhado na Seção 5.4.2.

5.6 Avaliação de uso

Com o sistema de apoio GAPRO finalizado, foi feita uma avaliação com o objetivo de entender em que pontos o sistema pode ser melhorado e como um usuário se comporta durante sua utilização.

Para esta tarefa, foi convidada uma profissional com experiência em gestão de projetos, com forte atuação na área de análise de requisitos e de processos de negócio, experiência em processo de desenvolvimento de software e em planejamento e liderança de projetos. Além de já ter trabalhado para grandes empresas como CSN¹³, Microsoft¹⁴ e Zoom¹⁵, ela é professora de um curso de extensão em Gestão e Especificação de Requisitos Ágeis da PUC-Rio e mestranda do PPGI/UNIRIO.

Durante a avaliação, foi possível registrar todos os comentários expostos pela avaliadora, como críticas e sugestões de otimização do sistema e de sua interface. A avaliadora utilizou o software em um notebook com o auxílio de um mouse. A avaliação foi realizada dentro de uma das salas de estudo da faculdade. A utilização do software foi observada de uma distância razoável, permitindo que a avaliadora permanecesse focada durante o uso.

Os comentários da avaliadora foram realizados ao fim da interação de cada tela do software. Nesse momento os comentários eram registrados em uma planilha. Após os comentários de uma tela, a avaliadora seguia com a utilização do software.

Durante a avaliação, foi possível registrar todos os comentários expostos pela avaliadora, como críticas e sugestões de otimização do sistema e de sua interface.

Com a lista de comentários, a próxima etapa foi analisar cada item registrado e avaliar o impacto dessas mudanças no sistema de apoio. Na Tabela 14, para cada item da lista é possível visualizar como ficou sua situação ao final da análise dos comentários da avaliação.

Tabela 14. Avaliação de uso do sistema GAPRO

ITEM	FEEDBACK	SITUAÇÃO
#1	Melhorar a descrição dos fatores críticos na tela inicial do gerente, que conta com as instruções de uso do sistema.	Implementado
#2	Inserir um atalho na tela inicial do gerente, para que ele possa criar seu perfil logo após ler as instruções do sistema.	Implementado
#3	Inserir <i>tooltips</i> (balões de comentários) para os botões de ação das telas, como: editar, visualizar, excluir e outros.	Trabalho Futuro

¹³ Companhia Siderúrgica Nacional de Volta Redonda - RJ

¹⁴ Empresa de tecnologia

¹⁵ Empresa de tecnologia

ITEM	FEEDBACK	SITUAÇÃO
#4	Criação de forma de ordenação por 'cards' ou similar para melhor ordenação dos fatores críticos pelo gerente. Talvez classificar em criticidade alta, média ou baixa.	Trabalho Futuro
#5	Apesar de ser feita apenas uma vez por projeto/equipe, a ordenação dos fatores críticos foi uma tarefa um pouco demorada e trabalhosa.	Trabalho Futuro
#6	Explicar melhor como classificar as estrelas na tela do questionário	Implementado
#7	Deixar claro quando o gerente precisa usar o sistema. É no início de um projeto, durante ou após?	Implementado
#8	Revisar a descrição de algumas perguntas. Deixar claro que as consequências dos problemas são apenas exemplos.	Implementado
#9	Deixar claro no sistema que as práticas recomendadas para um determinado problema não precisam ser utilizadas todas juntas, e que talvez a adoção de apenas uma das práticas recomendadas não resolva o problema. Depende do problema e das práticas em questão.	Implementado
#10	Revisar práticas relacionadas ao fator "gerenciamento de riscos".	Implementado
#11	Na descrição de reuniões diárias, talvez seja melhor tirar o gerente e o cliente, pois apesar de estar descrito assim em algumas referências, na prática é bem difícil que o gerente e o cliente participem desta reunião.	Implementado
#12	Revisar a descrição da prática BDD.	Implementado
#13	Em práticas que são conhecidas por mais de um nome, inserir na descrição da prática o segundo nome ou mencionar "também conhecida como..."	Implementado
#14	Nas práticas recomendadas, fazer a lista das práticas de forma similar à das perguntas. Ao clicar na prática o sistema exibe a descrição.	Trabalho Futuro
#15	Na aba 'Ranking de Práticas Ágeis' do relatório, ao invés de exibir pontos por cada prática, classificar as práticas por recomendação alta, média ou baixa.	Trabalho Futuro

Analisando os itens da Tabela 14, a maioria dos comentários é relacionada diretamente ao modo como os textos do sistema foram escritos e apresentados ao usuário (comentários podem ser revistos nos itens #1, #6, #7, #9, #10, #11, #12).

Textos com instruções foram rapidamente revisados e textos envolvendo fatores críticos e práticas ágeis exigiram que novas fontes de informação fossem pesquisadas para tornar estes itens mais claros e objetivos para os usuários do sistema.

Para usabilidade, um ponto importante foi registrado: houve grande demora para que o usuário ordenasse os fatores críticos ao criar seu perfil no sistema. Com as orientações dos itens ‘#4’ e ‘#5’ da Tabela 14, espera-se otimizar o tempo de execução desta tarefa pelo usuário, ao tornar a interface mais amigável e simples.

Ações como responder ao questionário, gerar o relatório e navegar entre as telas ocorreu de forma bastante fluida. A avaliadora não teve dúvidas em nenhum momento sobre em qual botão deveria clicar ou qual tela deveria acessar a seguir.

As caixas com instruções presentes em cada tela do sistema auxiliam a navegação do usuário pelo sistema. Mesmo que um usuário acesse uma página que não deveria no processo, o sistema consegue exibir uma mensagem amigável orientando-o sobre como prosseguir com a utilização do sistema de apoio. Durante a avaliação, não foram registrados erros do sistema ou lentidão de acesso às páginas.

Outros itens, classificados como ‘Trabalhos Futuros’ na Tabela 14, mesmo possuindo implementações relativamente simples, serão implementados em novas versões do sistema de apoio, pois neste projeto a prioridade e objetivo é poder demonstrar a validade de uso do sistema de apoio, e comprovar que possui real contribuição como ferramenta. Os itens descritos não terão neste momento impacto no modelo de negócio ou conteúdo do trabalho.

Ao final da avaliação, a avaliadora foi questionada se o sistema GAPRO de fato consegue auxiliar ou não um gerente a escolher práticas ágeis para o seu projeto. A avaliadora acredita que sim, mas segundo ela, isso dependerá do nível de conhecimento que o gerente teria previamente sobre as práticas.

O avaliador apresentou algumas críticas sobre o relatório do sistema gerado para seu perfil: algumas práticas não deveriam ser recomendadas em alguns casos que aparecerem no relatório; há um volume muito grande de práticas recomendadas, e em alguns casos várias destas práticas são recomendadas para o mesmo propósito, e o gerente precisaria escolher que práticas usaria.

Por fim, o avaliador acredita que um gerente sem uma mínima experiência com métodos ágeis ficaria perdido e um gerente com muita experiência talvez não achasse muito valor na utilização do sistema, mas os iniciantes que já entendam um pouco sobre métodos ágeis podem sim ser auxiliados e tirar proveito da ferramenta.

5.7 Considerações Finais

Neste capítulo foram apresentadas as funcionalidades do sistema GAPRO, passando por sua arquitetura, modelagem e as ferramentas utilizadas no seu desenvolvimento. Foram apresentadas as telas do sistema e uma avaliação de uso com um usuário da área de projetos.

Foi visto na prática, um sistema de apoio apresentado em âmbito profissional, contribuindo efetivamente para a aprendizagem e aprimoramento de profissionais de projeto.

Sistemas de apoio à decisão podem ser aplicados como um complemento à capacitação e experiência dos profissionais. Acreditamos que a combinação experiência mais recomendações se mostra mais motivador ao profissional que um conjunto de textos teóricos, indo além do método tradicional de aprendizagem e trazendo novas experiências.

A aplicação do questionário permitiu extrair comentários significativos da experiência de uso do usuário, sendo mais efetivas tratando com prioridade as deficiências de um gerente em seu próprio ambiente.

CAPÍTULO 6 – CONCLUSÃO

Neste capítulo serão resumidos os objetivos deste trabalho e as atividades realizadas para alcançá-los. Em seguida serão discutidas as limitações encontradas e as principais contribuições. Por fim serão discutidos os trabalhos futuros.

6.1 Considerações Finais

O objetivo desse trabalho foi desenvolver um sistema de Apoio a Decisão que auxilie gerentes de projetos de software a escolher práticas propostas por métodos ágeis de desenvolvimento de software. Para isso, foram levantados problemas que impedem o cumprimento dos principais fatores que contribuem para o sucesso de projetos de software. Em seguida, foram selecionadas práticas ágeis de desenvolvimento de software que auxiliam no cumprimento destes mesmos fatores.

Desta forma, baseado no relato dos problemas encontrados pelo gerente durante o gerenciamento de seus projetos, é possível recomendar práticas ágeis que podem colaborar para a mitigação dos problemas relatados.

A avaliação de uso foi fundamental para que o sistema e o resultado final deste trabalho pudessem ser medidos de forma concreta. Além disso, a avaliação possibilitou que fossem identificadas e aperfeiçoadas questões de usabilidade do sistema e conteúdo como as descrições dos problemas e práticas ágeis.

6.2 Limitações

A grande quantidade de metodologias e práticas ágeis existentes foi vista como um fator limitante, pois foi necessário selecionar as práticas e metodologias que seriam abordadas e inseridas no sistema.

A determinação dos pesos das práticas ágeis também foi vista como um fator limitante, pois as práticas ágeis que não foram consideradas no *survey* utilizado no projeto não possuíam taxa de relevância determinada previamente, sendo necessário adotar a menor taxa dentre as práticas contempladas no *survey* para estes casos.

Outra limitação foi a impossibilidade de se utilizar a GAPRO em um ambiente real de gerenciamento de projetos.

6.3 Principais Contribuições

Entende-se que o objetivo inicial do projeto foi alcançado, pois foi possível desenvolver um sistema de Apoio a Decisão que recomende práticas ágeis que podem mitigar problemas encontrados por gerentes de projeto em projetos de software.

Com a identificação destes problemas, é possível determinar quais são os problemas mais evidentes dentre os levantados e que impactariam com mais frequência os projetos de software. É possível determinar quais dos fatores críticos de sucesso em projetos de software são considerados mais relevantes pelos gerentes de projeto, a partir da ordenação destes fatores realizada no perfil do gerente no sistema.

A flexibilidade fornecida pela possibilidade do cadastro de problemas, práticas ágeis e metodologias através do módulo de administração do sistema permite que novos problemas em projetos de software sejam relatados futuramente, e que possíveis novas soluções sejam recomendadas através das práticas ágeis.

6.4 Trabalhos Futuros

Novas funcionalidades estão previstas para versões futuras do sistema. Para o grupo Administrador, serão implementadas: gerenciamento de usuários e estatísticas do sistema, onde será possível verificar informações estatísticas dos usuários, como as práticas mais recomendadas, questões com mais votos, dentre outras. Para o grupo Gerente, estão previstas mudanças na ordenação dos fatores críticos na classificação, que será por criticidade alta, média ou baixa.

Serão implementadas melhorias de usabilidade no relatório do Gerente para a visualização de práticas recomendadas, e adição de *tooltips* nos botões de inserção, exclusão, adição e visualização presentes no sistema.

Está prevista uma evolução no modelo de dados e uma nova funcionalidade do sistema que permitirá o cadastro e armazenamento dos resultados obtidos pelos gerentes após aplicar as práticas recomendadas pelo GAPRO em seus projetos. Com o desenvolvimento desta nova funcionalidade será possível aperfeiçoar as recomendações feitas pelo sistema através do recálculo dos pesos das práticas ágeis, e novas informações serão adicionadas de acordo com o *feedback* dos usuários do sistema.

Uma possível evolução será tratar a escolha dos fatores críticos, olhando probabilidade e impacto de cada fator crítico. Esta evolução poderá ser tratada com a implementação do AHP (*Analytic Hierarchy Process*).

O AHP foi desenvolvido na década de 1970 por Thomas L. Saaty, atualmente é aplicado para a tomada de decisão em diversos cenários complexos, em que pessoas trabalham em conjunto para tomar decisões e onde percepções humanas, julgamentos e consequências possuem repercussão de longo prazo (BHUSHAN; RAI, 2004).

Expandir o uso do sistema para outros profissionais além de gerentes de projetos, por exemplo desenvolvedores, analistas, e outros profissionais, poderão acessar o sistema e responder a questionários específicos para seu grupo de usuário.

Espera-se também que o sistema GAPRO seja utilizado em um projeto real de desenvolvimento de software e que, com o feedback obtido, ele possa ser evoluído.

REFERÊNCIAS

ABRANTES, José Fortuna. **Estudos experimentais sobre agilidade no desenvolvimento de software e sua utilização no processo de teste**. Tese (Doutorado em Engenharia de Sistemas e Computação) – Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa de Engenharia, UFRJ. 2012.

Agile Manifesto. Disponível em <<http://agilemanifesto.org>> Acesso em: 15/05/2016

ALTER S.L. **Decision support systems: current practice and continuing challenges**. Addison-Wesley, London, 1980.

AVERWEG, F. R. U. **Decision-making support systems: Theory & practice**. 2012. Disponível em: < <http://bookboon.com/en/decision-making-support-systems-ebook> > Acesso em: 14/04/2016

BASSI FILHO, D. L. **Experiências com desenvolvimento ágil**. Tese de Doutorado. Universidade de São Paulo. 2008.

BECK, Kent. **Extreme Programming Explained: Embrace change**. Reading, Massachusetts: Ed. Addison-Wesley, 2000.

Berkun, Scott. **A arte do gerenciamento de projetos** / Scott Berkun . tradução Carlos Augusto Caldas de Moraes, Terese Cristina Felix de Souza. - Porto Alegre : Bookman. 2008

BERNARDO, Kleber. **Introdução ao extreme programming (XP)**. Disponível em: < <http://www.culturaagil.com.br/kanban-do-inicio-ao-fim/>> Acesso em: 19/05/2016.

BHUSHAN, N. & RAI, K. (2004). *Strategic Decision Making: Applying the Analytic Hierarchy Process*. New York: Springer.

CodeIgniter At a Glance. Disponível em < https://codeigniter.com/user_guide/overview/at_a_glance.html> Acesso em 08/06/2016

CRUZ, Fábio. **Scrum e Agile em Projetos: Guia Completo**. Rio de Janeiro: Brasport, 2015.

Desenvolvimento Ágil. Disponível em < <http://www.desenvolvimentoagil.com.br/>>
Acesso em: 15/05/2016

Decision Support System. Disponível em:
<https://en.wikipedia.org/wiki/Decision_support_system> Acesso em: 23/04/2016

DENNIS, A., WIXOM, R. e ROTH, M. **Análise e Projetos de Sistemas**. Tradução Amir Elias Abdalla Kurban. - 5. ed. - Rio de Janeiro: LTC, 2014.

DevMedia. Modelo Entidade Relacionamento (MER) e Diagrama Entidade-Relacionamento (DER). Disponível em: <<http://www.devmedia.com.br/modelo-entidade-relacionamento-mer-e-diagrama-entidade-relacionamento-der/14332>>
Acesso em 19/06/2016

DRUZDZEL, M. J. and R. R. Flynn. **Decision Support Systems. Encyclopedia of Library and Information Science**. A. Kent, Marcel Dekker, Inc. 1999.

DVIR, D.; LIPOVETSKY, S.; SHENHAR, A.; TISHLER, A. *In search of project classification: a non-universal approach to project success factors*. **Research Policy**, **27** (9), p. 915-935, 1998.

FADEL, Aline Cristine. SILVEIRA, Henrique da Mota. **Metodologias ágeis no contexto de desenvolvimento de software: XP, Scrum e Lean**. Limeira, 2010.
Disponível em:
<http://www.ceset.unicamp.br/liag/Gerenciamento/monografias/Lean%20Agil_v8.pdf>
f> Acesso em: 15/05/2016.

FINLAY, P. N. (1994). **Introducing decision support systems**. Oxford, UK Cambridge, Mass., NCC Blackwell; Blackwell Publishers.

FONSECA, Daniel. **Conceitos básicos sobre metodologias ágeis para desenvolvimento de software (metodologias clássicas x extreme programming)**. Disponível em: < <http://www.devmedia.com.br/conceitos-basicos-sobre->

metodologias-ageis-para-desenvolvimento-de-software-metodologias-classicas-x-extreme-programming/10596 > Acesso em: 19/05/2016.

FORSTER, N. S. e ROCKART, J. F. *Critical Success Factors: An Annotated Bibliography*. Working Paper no. 191. **Center for Information Systems Research, Sloan School of Management. Massachusetts Institute of Technology**, 1989.

GABARDO, Ademir Cristiano, **PHP e MVC: com CodeIgniter** -- São Paulo : Novatec Editora, 2012.

HELDMAN. K. PMP: Project Management Professonion Study Guide. 2.ed. Sybex, Inc. 2004.

HIBBS, C.; JEWETT, S.; SULLIVAN, M. *The Art of Lean Software Development: A Practical and Incremental Approach*, 1ª Edição, O'Reilly Media, Inc., CA. 2009 *apud* LEAL, Tainá Catarina Oliveira Abrahão. *Pós-Agilismo – Um estudo sobre o legado das metodologias ágeis para os processos de software*. 2014. 58 f. Monografia (Bacharelado em Sistemas de Informação) – Escola de Informática Aplicada. Universidade Estadual do Estado do Rio de Janeiro, Rio de Janeiro.

HIRAMA, Kechi. **Engenharia de Software: qualidade e produtividade com tecnologia**. Rio de Janeiro: Elsevier, 2011.

HOLSAPPLE, C.W., and A. B. Whinston. (1996). **Decision Support Systems: A Knowledge-Based Approach**. St. Paul: West Publishing.

KEEN, P. G. W. and M. S. Scott-Morton. **Decision Support Systems: An Organizational Perspective**. Reading, MA: Addison-Wesley, 1978. appeared in DSS News, May 6, 2001, Vol. 2, No. 10

KNIBERG, H.; SKARIN, M. **Kanban and Scrum making the most of both**. Managing Editor: Diana Plesa. Enterprise software development series. InfoQ. USA. ISBN 978-0-557-13832-6, 2010.

KNOWLEDGE21. **Os princípios ágeis.** Disponível em: <<http://www.knowledge21.com.br/sobreagilidade/agilidade/os-principios-ageis/>> Acesso em: 15/05/2016

LEAL, Tainá Catarina Oliveira Abrahão. **Pós-Agilismo – Um estudo sobre o legado das metodologias ágeis para os processos de software.** Monografia (Bacharelado em Sistemas de Informação) – Escola de Informática Aplicada. Universidade Estadual do Estado do Rio de Janeiro, Rio de Janeiro. 2014.

LEAN INSTITUTE BRASIL. **Desenvolvimento Lean de produtos.** 2010. Disponível em: <<http://www.lean.org.br/>> Acesso em: 15/05/2016.

LEIDECKER, J. K. e BRUNO, A. V. *Identifying and Using Critical Success Factors. Long Range Planning*, 17 (1), p. 23-32, 1984 *apud* VEZZONI, Guilherme. **Identificação e análise de fatores críticos de sucesso em projetos.** Monografia (Bacharelado em Engenharia de Produção) – Centro Universitário UNISEB. Ribeirão Preto, 2011.

LITTLE, J. D. C. (1975). **Models and Managers: The Concept of a Decision Calculus.** *Management Science*, vol. 16, n. 8, p. B466-485, April.

MARIOTTI, Flavio S. **Kanban: o ágil adaptativo.** *Revista Engenharia de Software Magazine*. n. 45. Disponível em: <<http://www.devmedia.com.br/revista-engenharia-de-software-magazine/edicoes?page=3>> Acesso em 05/06/2016

MAXFIELD, Wade. **Aprendendo MySQL e PHP.** 1.ed. Ed. Makron. 2002.

MEDEIROS, Higor. **Introdução ao extreme programming (XP).** Disponível em: <<http://www.devmedia.com.br/introducao-ao-extreme-programming-xp/29249>> Acesso em: 19/05/2016.

Microsoft Developer Network. Disponível em <[https://msdn.microsoft.com/pt-br/library/hh533841\(v=vs.110\).aspx](https://msdn.microsoft.com/pt-br/library/hh533841(v=vs.110).aspx)> Acesso em: 12/06/2016

Nasir M H N and Sahibuddin S (2011) “**Critical success factors for software projects: A comparative study**” Faculty of Computer Science and Information

Technology and Advanced Informatics School *apud* PIMENTA, D.; SANTOS, G. **Análise de Práticas Ágeis no Apoio a Fatores Críticos de Sucesso em Projetos de Software**. Relatório Técnico (Pós Graduação em Informática) – Universidade Federal do Estado do Rio de Janeiro. Rio de Janeiro, 2016.

NETO, Gert Uchôa Müller. **Métodos Tradicionais Versus Ágeis: Um Estudo Comparativo Através do Traningcad**. 2009. 53 f. Monografia (Graduação em Sistemas de Informação) - Faculdade de Ciência e Tecnologia de Caruaru. Universidade de Pernambuco. Caruaru.

PIMENTA, D.; SANTOS, G. **Análise de Práticas Ágeis no Apoio a Fatores Críticos de Sucesso em Projetos de Software**. Relatório Técnico – RelatDIA nº 0002/2016, Relatórios Técnicos do Departamento de Informática Aplicada da UNIRIO – Universidade Federal do Estado do Rio de Janeiro. Rio de Janeiro, 2016.

PINTO, J.K. & D.P.SLEVIN. **Critical Factors in Successful Project Implementation**. IEEE Transactions on Engineering Management, EM-34(1). 1987.

PMSOURVEY.ORG. **Relatório Mundial 2012**. Disponível em: <<http://pt.slideshare.net/ProjectBuilder/relatrio-2012-geral#>> Acesso em: 23/11/2015

POWER, D.J. **A Brief History of Decision Support Systems**. Disponível em: <<http://DSSResources.COM/history/dsshhistory.html>> Acesso em 20/05/2016.

POWER, D.J. **DSS Theory: What are characteristics of a decision support system?**. Disponível em: <<http://dssresources.com/faq/pdf/101.pdf>> Acesso em 20/05/2016.

POWER, D.J. **DSS Theory: What are characteristics of a decision support system?**. DSSResources.com, World Wide Web, <http://dssresources.com/faq/pdf/13.pdf>, last update September 05, 2005.

POWER, D.J. **DSS Theory: What are advantages and disadvantages of data warehouses?**. DSSResources.com, World Wide Web, <http://dssresources.com/faq/pdf/180.pdf>, last update December 03, 2008.

PRIKLADNICKI, Rafael (Org.); WILLI, Renato (Org.); MILANI, Fabiano (Org.). **Métodos Ágeis para Desenvolvimento de Software**. Porto Alegre: Bookman, 2014.

PROJECT MANAGEMET INSTITUTE – PMI. Conjunto de conhecimentos em gerenciamento de projetos, Guia PMBOK, quinta edição, 2013

Rafael M. de Mello, Pedro C. da Silva Guilherme H. Travassos. **Agilidade em Processos de Software: Evidências Sobre Características de Agilidade e Práticas Ágeis**. 2014

RAMOS JÚNIOR, Idmar. **Os Principais Conceitos Ágeis Explicados**. [s.l.], 2015. Disponível em: <http://bizstart.com.br/os-principais-conceitos-ageis-explicados/> Acesso em: 05/06/2016.

SCHWABER, K; SUTHERLAND, J. **The Scrum Guide–The Definitive Guide to Scrum: The Rules of the Game**. Scrum. org, 2014.

SILVA, Maurício Samy, **Criando Sites com HTML**, São Paulo, Editora: Novatec, 2008

SILVA, Maurício Samy, **HTML 5**, São Paulo, Editora: Novatec, 2011

SILVA, Maurício Samy, **JavaScript: Guia do programador**, São Paulo, Editora: Novatec, 2010.

SOUSA, Luís. **Test Driven Development, uma metodologia ágil**. Disponível em: <<http://www.mindsources.pt/content/test-driven-development-uma-metodologia-%C3%A1gil>>. Acesso em: 15/05/2016.

SPRAGUE, R, H., Jr., "A Framework for the Development of Decision Support Systems," **Management Information Systems Quarterly**, vol. 4, no. 4, Dec. 1980, pp. 1-26.

SPRAGUE, R. H. and E. D. Carlson. **Building Effective Decision Support Systems** Englewood Cliffs, N.J.: Prentice-Hall, Inc.: 1982.

SPRAGUE, R. H. and E. D. Carlson. **Decision Support Systems Wisdom**. Disponível em: <http://dssresources.com/dsswisdom/password/page9.html>. Acesso em: 20/05/2016

SPRAGUE Jr, Ralph H, Hugh J. Watson. **Sistemas de apoio à decisão – colocando a teoria em prática**. Campus, 1991.

STEFANINI. **Kanban: Gerenciamento ágil de projetos**. Disponível em: <https://stefanini.com/br/2013/11/kanban-gerenciamento-agil-projetos/> Acesso em: 15/05/2016

TELES, Vinícius Manhães. **Extreme Programming: Aprenda como encantar seus usuários desenvolvendo software com agilidade e alta qualidade**. São Paulo - SP: Novatec Editora Ltda, 2004.

The Standish Group. **CHAOS MANIFESTO 2013**. Disponível em: <http://versionone.com/assets/img/files/CHAOSManifesto2013.pdf> Acesso em: 23/11/2015

TSUI, F. F. e KARAM, O. **Fundamentos de engenharia de software**; tradução de Edson Tanaka. - 2. ed. - Rio de Janeiro: LTC, 2013

TURBAN, E. (1995). **Decision support and expert systems: management support systems**. Englewood Cliffs, N.J., Prentice Hall.

VARGAS, V. R. **Gerenciamento de projetos: estabelecendo diferenciais competitivos**. 5.ed. – Rio de Janeiro: Brasport, 2003

XAVIER, Fabrício S.V. **PHP – do Básico à Orientação a Objetos**. Rio de Janeiro: Editora Ciência Moderna Ltda., 2008.

ANEXO A - Relacionamentos entre Áreas de conhecimento, problemas de projetos de software e fatores críticos de sucesso afetados

Área	Problema	Descrição do problema	Fatores críticos de sucesso que foram afetados
Pessoas	Falta de comunicação entre a equipe	Nos meus projetos foi verificado que a equipe não se comunica o suficiente. Os membros estão frequentemente desinformados sobre as atividades dos seus companheiros de equipe, podendo resultar em problemas como perda de tempo de projeto e retrabalho.	1. Comunicação Eficaz e feedback
Pessoas	Falta de apoio da alta administração	Nos meus projetos foi verificado que a alta administração não reconheceu a importância dos produtos/serviços que seriam gerados, dessa forma não apoiou a execução dos projetos de forma adequada.	1. Objetivos e metas claros 2. Apoio da alta administração 3. Comunicação eficaz e feedback
Pessoas	Falta de envolvimento do cliente	Nos meus projetos foi verificado que o cliente não se envolve o suficiente, não cumprindo tarefas de sua responsabilidade (por exemplo, reuniões, feedback, homologação).	1. Usuário/ envolvimento do cliente 2. Comunicação eficaz e feedback 3. Designação clara de papéis e responsabilidades
Pessoas	Falta de comunicação com o cliente	Nos meus projetos foi verificado que a comunicação com o cliente não foi eficaz, podendo resultar em uma expectativa muito alta ou muito baixa do cliente em relação ao projeto.	1. Requisitos e especificações claras 2. Comunicação eficaz e feedback 3. Objetivos e metas claros
Pessoas	Funcionários inadequados para o projeto	Nos meus projetos foi verificado que a equipe alocada não possuía o conhecimento e/ou experiência necessários.	1. Pessoal qualificado e suficiente 2. Planejamento adequado 3. Designação clara de papéis e responsabilidades
Pessoas	Funcionários insuficientes	Nos meus projetos foi verificado que a equipe alocada para o projeto foi insuficiente.	1. Pessoal qualificado e suficiente 2. Planejamento adequado
Pessoas	Falta de comprometimento da equipe	Nos meus projetos foi verificado que houve falta de comprometimento por parte da equipe.	1. Equipe comprometida e motivada
Pessoas	Falta de motivação da equipe	Nos meus projetos foi verificado que a equipe estava desmotivada.	1. Equipe comprometida e motivada
Processos	Estimativas de tempo e prazos não condizem com o tempo real das entregas	Nos meus projetos foi verificado que o tempo estimado para a conclusão das atividades foi muito maior/menor do que o tempo efetivamente gasto na atividade até a entrega.	1. Prazos realistas 2. Planejamento adequado
Processos	Grande impacto no orçamento devido a mudanças nas necessidades do cliente durante o desenvolvimento	Nos meus projetos foi verificado que mudanças nas necessidades do cliente alteraram de forma drástica o orçamento.	1. Orçamento realista 2. Planejamento adequado 3. Gerenciamento de riscos

Área	Problema	Descrição do problema	Fatores críticos de sucesso que foram afetados
Processos	Grande impacto no prazo de entrega do projeto devido à mudanças nas necessidades do cliente durante o desenvolvimento	Nos meus projetos foi verificado que mudanças decorrentes de novas necessidades do cliente alteraram bastante o prazo de entrega.	1. Planejamento adequado 2. Gerenciamento de riscos 3. Controle efetivo de mudanças e gerenciamento de configuração 4. Monitoramento e controle efetivo do projeto 5. Bom gerenciamento da qualidade
Processos	Requisitos indefinidos no início do projeto	Nos meus projetos a maioria dos requisitos ainda não são bem definidos no início do desenvolvimento, podendo dificultar a medição do esforço e estimativas de prazos.	1. Requisitos e especificações claras 2. Prazos realistas 3. Planejamento adequado 4. Gerenciamento de riscos
Processos	Recursos limitados para o projeto	Nos meus projetos foi verificado que recursos necessários para o projeto foram comprometidos ou limitados.	1. Orçamento realista 2. Pessoal qualificado e suficiente
Processos	Orçamento inadequado	Nos meus projetos foi verificado que o orçamento estipulado foi inadequado para a realidade do projeto.	1. Orçamento realista 2. Planejamento adequado
Processos	Orçamento estourou o limite estipulado	Nos meus projetos foi verificado que o orçamento estourou o limite estipulado.	1. Planejamento adequado 2. Gerenciamento de riscos
Processos	Alto índice de mudanças nas necessidades do cliente durante o desenvolvimento	Nos meus projetos foi verificado que houve um alto índice de mudanças nas necessidades do cliente durante o desenvolvimento, podendo ter dificultado a conclusão e a entrega.	1. Requisitos e especificações claras 2. Gerenciamento de riscos
Processos	Planejamento inadequado	Nos meus projetos foi verificado que o planejamento não foi realizado de forma adequada.	1. Prazos realistas 2. Planejamento adequado 3. Monitoramento e controle efetivo do projeto
Processos	Relatórios defasados	Nos meus projetos foi verificado que os relatórios de progresso do projeto submetidos estavam defasados ou com dados incorretos.	1. Comunicação Eficaz e feedback 2. Relatório de progresso atualizado 3. Monitoramento e controle efetivo do projeto 4. Gerenciamento de riscos
Processos	Monitoramento e controle do projeto deficiente	Nos meus projetos foi verificado que o monitoramento e controle foram insuficientes ou não foram executados da forma correta. Pode não ter havido sinalização e acompanhamento dos projetos e as partes envolvidas, com isso, não tinham conhecimento do estado geral do projeto.	1. Comunicação Eficaz e feedback 2. Relatório de progresso atualizado 3. Monitoramento e controle efetivo do projeto 4. Gerenciamento de riscos
Processos	Ausência ou má execução do gerenciamento de riscos	Nos meus projetos foi verificado que não houve uma gerência adequada de riscos. Os riscos não foram gerenciados de modo a mitigar os impactos no projeto ou os riscos errados foram tratados.	1. Planejamento adequado 2. Monitoramento e controle efetivo do projeto 3. Gerenciamento de riscos

Área	Problema	Descrição do problema	Fatores críticos de sucesso que foram afetados
Processos	Controle de qualidade ausente ou insuficiente	Nos meus projetos foi verificado que a qualidade do produto/ serviço gerado não correspondia às expectativas do cliente.	1. Requisitos e especificações claras 2. Objetivos e metas claros 3. Comunicação eficaz e feedback 4. Bom gerenciamento da qualidade
Processos	Papéis e responsabilidades não foram claramente definidos	Nos meus projetos foi verificado que os papéis ou responsabilidades não foram claramente definidos para todos os membros da equipe.	1. Comunicação eficaz e feedback 2. Designação clara de papéis e responsabilidades
Processos	Estimativas/prazos foram exigidos antes da definição completa dos requisitos	Nos meus projetos foi verificado que as estimativas foram feitas e exigidas antes que os requisitos estivessem completamente definidos, resultando em prazos imprecisos.	1. Requisitos e especificações claras 2. Prazos realistas 3. Planejamento adequado
Processos	Testes do software foram insuficientes	Nos meus projetos foi verificado que os testes foram deixados para o final do desenvolvimento, resultando em um sistema com muitos <i>bugs</i> e problemas.	1. Planejamento adequado 2. Bom gerenciamento da qualidade
Processos	Testes do software impactaram prazos de entrega	Nos meus projetos foi verificado que os testes foram deixados para o final do desenvolvimento, atrasando prazos de entrega.	1. Prazos realistas 2. Planejamento adequado 3. Gerenciamento de riscos 4. Bom gerenciamento da qualidade
Processos	Prazos agressivos resultaram em prazos não cumpridos	Nos meus projetos, por necessidade do cliente foi dado um prazo agressivo que não correspondia com as estimativas reais das atividades, resultando em prazos não cumpridos.	1. Prazos realistas 2. Planejamento adequado 3. Usuário/ envolvimento do cliente
Processos	Prazos agressivos resultaram em problemas no software	Nos meus projetos, por necessidade do cliente foi dado um prazo agressivo que não correspondia com as estimativas reais das atividades, resultando em problemas no software.	1. Prazos realistas 2. Planejamento adequado 3. Bom gerenciamento da qualidade
Técnicos	Falta de familiaridade com as tecnologias utilizadas	Nos meus projetos foi verificado que a equipe não estava familiarizada com as tecnologias utilizadas ou ocorreram mudanças nas tecnologias utilizadas ao longo da execução.	1. Pessoal qualificado e suficiente 2. A familiaridade com a tecnologia de desenvolvimento
Técnicos	Falta de familiaridade com a metodologia utilizada	Nos meus projetos foi verificado que a equipe não estava familiarizada com as metodologias utilizadas no projeto, ou ocorreram mudanças na metodologia utilizada.	1. Pessoal qualificado e suficiente 2. A familiaridade com a metodologia de desenvolvimento 3. Planejamento adequado
Técnicos	Testes realizados não foram efetivos	Nos meus projetos foi verificado que os testes realizados não foram efetivos, comprometendo a qualidade do software.	1. Requisitos e especificações claras 2. Bom gerenciamento da qualidade

ANEXO B - Práticas Ágeis Selecionadas

As ferramentas de apoio utilizadas nos métodos Scrum e Kanban, como o gráfico *burndown*, as histórias de usuários e os quadros de tarefas foram consideradas como parte da prática ágil "Visibilidade do Projeto", e citadas nas descrições de suas respectivas práticas. As definições das práticas foram retiradas de Abrantes (2012), Desenvolvimento Ágil (2013), Fonseca (2016), Knowledge21 (2016), Ramos Júnior (2015) e Sousa (2016).

Prática	Descrição
Backlog do Produto	<p>O Backlog é uma lista de todas as funcionalidades pensadas para um produto. Inicialmente, o Backlog começa com as necessidades mais básicas, mas geralmente muda com o tempo, de acordo com o aprendizado do time sobre o produto que está desenvolvendo e seus usuários (reais ou potenciais).</p> <p>Manter uma lista priorizada é importante para o fluxo de produção, para dar mais visibilidade às trocas que acontecem e aos efeitos dessas mudanças. Fazem parte do Backlog tarefas técnicas ou atividades diretamente relacionadas às funcionalidades solicitadas.</p>
Cliente Presente	<p>É fundamental a participação do cliente durante todo o desenvolvimento do projeto. O cliente deve estar sempre disponível para sanar todas as dúvidas de requisitos, evitando atrasos e até mesmo construções erradas. Uma ideia interessante é manter o cliente como parte integrante da equipe que fará os testes do software.</p>
Pequenas Liberações	<p>A prática de Pequenas Liberações, também conhecida como Entregas Curtas, incentiva o time de desenvolvimento a entregar novas versões de software de maneira contínua, reduzindo o tempo do ciclo de vida da implantação de funcionalidades e trazendo mais agilidade para os negócios. Ciclos curtos podem reduzir riscos, ajudar a lidar com mudanças nos requisitos e reduzir o impacto de erros de planejamento. Ao final de cada release, o cliente revê todo o produto podendo identificar defeitos e fazer ajustes nos requisitos futuros.</p>
Reuniões de Planejamento	<p>Nas reuniões de planejamento, são escolhidos os itens da lista (Backlog) com prioridade mais alta para o desenvolvimento. Estes itens são quebrados em pequenas tarefas, que serão executadas durante a próxima iteração. Nesta reunião, o cliente tem a função de ajudar a definir e priorizar os itens do Backlog.</p>
Jogos de Planejamento	<p>Jogos de planejamento auxiliam a manter o foco no que é de maior valor para o cliente, e são executados sempre no início de uma iteração ou release. É feita uma determinação rápida do escopo do release, através da combinação de estimativas e prioridades do negócio.</p> <p>Nos jogos de planejamento, os desenvolvedores estimam o tempo para o desenvolvimento das funcionalidades através de pontos. O cliente é responsável por definir quais são as funcionalidades a serem entregues no próximo release, priorizando as que possuem maior valor. As estimativas podem ser refeitas durante as iterações à medida que os programadores aprenderem mais sobre o sistema.</p> <p>Como exemplos de jogos de planejamento podem ser citados o <i>Planning Poker</i> e a Técnica Pomodoro.</p>
Equipe Completa	<p>Refere-se à prática de incluir todos os perfis e perspectivas necessários na equipe para que ela possa ter bom desempenho, enfatizando o espírito de equipe, com todos os seus membros compartilhando um propósito e apoiando-se mutuamente. Clientes, usuários e demais interessados devem ter um envolvimento direto no projeto, a fim de</p>

Prática	Descrição
	possibilitar entender o comportamento do sistema mais cedo no ciclo de vida.
Visibilidade do Projeto	<p>Projetos ágeis por sua natureza estão continuamente mudando (planos, modelos, código e demais artefatos). O objetivo da prática de Visibilidade do Projeto é manter e monitorar a qualquer tempo as medições do progresso do projeto. As informações devem estar acessíveis para todos os envolvidos no projeto.</p> <p>Pode ser criado um painel na web para manter a qualquer tempo o status e as métricas relacionadas com o progresso do projeto. Múltiplos painéis podem ser usados para disponibilizar diferentes tipos de informação que atendam a todos os níveis organizacionais necessários.</p> <p>Para esta prática são utilizadas ferramentas de apoio como quadro de tarefas do Scrum ou Kanban, Gráfico Burndown e Histórias de Usuários.</p>
Reuniões Diárias	<p>Reuniões Diárias duram poucos minutos (geralmente de 15 a 20 minutos), incentivam a comunicação do time e a entender como caminha a evolução do projeto. Nestas reuniões, cada um dos participantes responde algumas questões fundamentais sobre sua atividade, como o que foi feito no dia anterior, o que será feito hoje e impedimentos para a execução da atividade.</p> <p>Reuniões Diárias podem auxiliar o gerenciamento e tratamento dos riscos do projeto, pois a equipe do projeto fica sabendo rapidamente de qualquer problema, que terá sido detectado há no máximo um dia de trabalho.</p>
Scrum de Scrums	<p>O objetivo desta prática é dar suporte em situações na qual a equipe é muito grande e necessita ser quebrada em várias equipes que precisam interagir constantemente em prol do progresso do projeto. O objetivo do Scrum de Scrums é similar ao das reuniões diárias, mas em maior escala. Enquanto reuniões diárias são mais viáveis com equipes pequenas de modo a ser uma reunião curta, o Scrum de Scrums visa realizar uma reunião mais especializada com o objetivo de manter as equipes atualizadas em relação aos acontecimentos desde a última reunião.</p> <p>No Scrum de Scrums, basicamente, divide-se um time Scrum em dois ou mais times Scrum. Cada um dos times possui suas próprias atividades, reuniões diárias, iterações (Sprints), e seus próprios desenvolvedores e líder de equipe. O líder de equipe de cada time comparece à reunião Scrum de Scrums, geralmente efetuada de uma à três vezes por semana, para comunicar a todos os times sobre as realizações passadas e futuras de todo o projeto.</p>
Propriedade Coletiva do Código	Com a utilização da prática de Propriedade Coletiva do Código, todos têm acesso e autorização para editar qualquer parte do código da aplicação, a qualquer momento. Ou seja, a propriedade do código é coletiva e todos são igualmente responsáveis por todas as partes. Esta prática pode contribuir para o ganho de tempo e disseminação do conhecimento, além da frequente identificação de oportunidades de melhoria e refatoração do código.
Desenvolvimento Orientado por Comportamento	O desenvolvimento orientado por comportamento é usado para integrar regras de negócios com a linguagem de programação, através da criação de testes que focam o comportamento do software. Além disso, pode melhorar a comunicação entre as equipes de desenvolvimento e testes, aumentando o compartilhamento de conhecimento.
Programação em Pares	A Programação em Pares sugere que todo e qualquer código produzido no projeto seja sempre implementado por duas pessoas juntas, diante do mesmo computador, revezando-se no teclado. Esta prática pode ajudar os desenvolvedores na criação de soluções mais simples, mais adaptáveis e mais fáceis de manter. Um dos principais benefícios da programação em pares é a permanente inspeção de código que ocorre durante seu uso, podendo reduzir a incidência de bugs em um sistema.
Refatoração	A prática de Refatoração consiste em alterar pequenas partes do sistema, frequentemente, sempre que é encontrada uma oportunidade para melhorar o código, tornando-o mais limpo, mais claro e mais fácil de ser compreendido. Tais alterações não mudam o comportamento das funcionalidades, apenas melhoram a estrutura do código. Agindo assim de forma sistemática e com frequência, as equipes de

Prática	Descrição
	desenvolvimento investem para que o software se mantenha sempre fácil de alterar.
Padronização de Código	Pelo fato de os desenvolvedores programarem diferentes partes do sistema com vários membros da equipe, a adoção de padrões de código é bastante interessante. Eles facilitam o entendimento do código, aumentam a legibilidade e melhoram a consistência entre membros da equipe. Os padrões devem ser fáceis de serem seguidos e devem ser adotados voluntariamente. Deve ser acordado pela equipe, assegurando que a comunicação possa ser feita via código, além de levar os desenvolvedores a entender facilmente o código de seus colegas.
Integração Contínua	Integração Contínua consiste em integrar o trabalho diversas vezes ao dia, contribuindo para que a base de código permaneça consistente ao final de cada integração e possibilitando que qualquer desenvolvedor possa obter todo o código do projeto, a qualquer momento. Para esta prática são utilizadas ferramentas de apoio como sistemas de controle de versões ou repositório de código.
Design Simples	A prática foca em manter o design simples, podendo contribuir para um desenvolvimento adaptável às necessidades. O design é mantido apropriado para que o projeto requer no momento. Deve ser revisado iterativamente e incrementalmente para garantir que continua apropriado.
Ritmo Sustentável	A prática foca em trabalhar com qualidade, buscando ter ritmo de trabalho saudável (40 horas/semana, 8 horas/dia). Busca-se promover um ritmo constante e sustentável para o trabalho do time que desenvolve o produto, o que se torna possível quando esse ritmo é apoiado por toda a cadeia, incluindo usuários e patrocinadores. No entanto, ao se exigir do time um compromisso com mais trabalho do que ele é capaz de produzir, são muitas vezes adotadas as horas extras, o trabalho em fins de semana e a pressa exagerada para se cumprir o prazo de entrega, por exemplo. Essas práticas podem levar à insatisfação dos membros do time de desenvolvimento, a uma menor produtividade e a uma menor qualidade no produto gerado.
Desenvolvimento Orientado por Testes	O Desenvolvimento Orientado por Testes (TDD) é uma técnica de desenvolvimento de software que consiste em pequenas iterações de desenvolvimento onde o caso de teste é escrito antes de ser implementada a própria funcionalidade. O processo de desenvolvimento de software resume-se à execução repetida dos passos: escrever um teste unitário que falhe antes de escrever qualquer código funcional; escrever o código funcional até que o teste unitário passe; no final, caso seja necessário, refatorar o código, assegurando que os testes unitários continuem todos a ter sucesso. A utilização do TDD pode diminuir o número de bugs e permitir que a aplicação final seja mais robusta uma vez que se garante que o código, para o qual se implementou testes, funciona.

ANEXO C - Metodologias Ágeis Selecionadas

O Anexo C possui a descrição que foi inserida no sistema como fonte alternativa de informações sobre os métodos ágeis abordados. As definições destes métodos foram retiradas de Beck (2000), Bernardo (2015), Desenvolvimento Ágil (2013), Fadel e Silveira (2010), Lean Institute Brasil (2010), Mariotti (2012), Medeiros (2016), Prikladnicki (2014), Schwaber e Sutherland (2014), Stefanini (2013) e Telles (2004).

Metodologia	Descrição
Scrum	<p>Scrum é uma metodologia ágil para gestão e planejamento de projetos e desenvolvimento ágil de software. É utilizado para trabalhos complexos nos quais é impossível prever tudo o que irá ocorrer. O Scrum consiste nos times do Scrum associados a papéis, eventos, artefatos e regras. As regras do Scrum integram os eventos, papéis e artefatos, administrando as relações e interações entre eles.</p> <p>No Scrum, os projetos são divididos em ciclos (tipicamente mensais) chamados de Sprints. Cada Sprint representa uma iteração dentro da qual um conjunto de atividades deve ser executado.</p> <p>O Time Scrum é composto pelo Product Owner, o Time de Desenvolvimento e o Scrum Master. O Product Owner, ou dono do produto, é o responsável por maximizar o valor do produto e do trabalho do Time de Desenvolvimento. O Time de Desenvolvimento consiste de profissionais que realizam o trabalho de entregar uma versão usável que potencialmente incrementa o produto "pronto" ao final de cada Sprint. O Scrum Master é responsável por garantir que o Scrum seja entendido e aplicado, e ajuda aqueles que estão fora do Time Scrum a entender quais as suas interações com o Time Scrum são úteis e quais não são. O Scrum Master ajuda todos a mudarem estas interações para maximizar o valor criado pelo Time Scrum.</p>
XP	<p>Extreme Programming (XP) é uma metodologia de desenvolvimento de software que adota os valores de comunicação, simplicidade, feedback e coragem. Estes quatro valores servem como critérios que norteiam as pessoas envolvidas no desenvolvimento de software.</p> <p>O objetivo principal do XP é levar ao extremo um conjunto de práticas que são ditas como boas na engenharia de software. O XP preconiza ciclos curtos que fornecem previsibilidade, redução de incertezas ou riscos, simplicidade e melhorias constantes de código (refatoração) para facilitar mudanças, testes automatizados e integração contínua para aumentar a confiança.</p>
Kanban	<p>O Kanban é uma metodologia de sinalização para controle de fluxo de operação aplicada em indústrias de forma geral. No setor de Tecnologia da Informação, kanban também é uma ferramenta visual de gestão de fluxo de desenvolvimento. É uma das metodologias de desenvolvimento de software menos prescritivas, se tornando adaptável a quase qualquer tipo de cultura.</p> <p>O Kanban é baseado na ideia onde atividades em andamento devem ser limitadas. Um novo item só pode ser iniciado quando o item em andamento é finalizado ou quando uma função automática inicia o mesmo instantaneamente. Basicamente, o Kanban tem como principal objetivo transformar o trabalho em andamento visível para toda a equipe, criando um sinal visual que indica se o novo trabalho pode ou não ser iniciado e se o limite acordado para cada fase está sendo respeitado. Uma outra característica importante do modelo Kanban é o conceito de "puxar tarefa" quando há capacidade de processá-la. Esse recurso vai de encontro ao tradicional modelo de "empurrar tarefa" conforme sua demanda, mantendo assim o bom desempenho da equipe.</p> <p>Um dos principais mecanismos utilizados pelo Kanban é o Quadro Kanban, que são quadros</p>

Metodologia	Descrição
	de cartões e post-its dispostos de acordo com o andamento do projeto para o controle visual do desenvolvimento de software ágil e do trabalho em andamento.
Lean	<p>A metodologia Lean é uma estratégia de negócios que busca aumentar a satisfação do cliente através de um melhor aproveitamento de recursos. No contexto de software, A metodologia Lean é a aplicação dos conceitos do sistema de produção da Toyota para o desenvolvimento de software, e quando aplicada corretamente tem como consequência um desenvolvimento de alta qualidade que é feito rapidamente e com um baixo custo.</p> <p>É distribuída em sete princípios: eliminar o desperdício, amplificar o aprendizado, adiar comprometerimentos e manter a flexibilidade, entregar rápido, tornar a equipe responsável, construir integridade e visualizar o todo.</p>