T-AADSP Requirements - Gerenciamento de Requisitos com base na abordagem AADSP

Caio Rosário Dias [*]
Instituto Federal de Educação, Ciência e Tecnologia da Bahia
Rua Emídio dos Santos, S/N, Barbalho
Salvador, Bahia
caiodias@ifba.edu.br

Antônio Carlos Santos Souza [†]
Instituto Federal de Educação, Ciência e Tecnologia da Bahia
Rua Emídio dos Santos, S/N, Barbalho
Salvador, Bahia
antoniocarlos@ifba.edu.br

Resumo

Os requisitos de *software* possuem uma natureza volátil e podem ser modificados durante o ciclo de vida do mesmo. O mal gerenciamento dessas mudanças e a inclusão de novos requisitos podem comprometer a qualidade do *software*. As ferramentas desenvolvidas para gerenciamento de requisitos muitas vezes não seguem os critérios estabelecidos pela literatura tais como identificação, controle de mudanças e rastreabilidade dos requisitos e não fornecem dados para a aplicação de algoritmos da área de Otimização da Engenharia de *Software* pois são focadas apenas em atender necessidades mercadológicas e o cumprimento de critérios estabelecidos por empresas certificadoras. Esse trabalho propõe a criação de um metamodelo para rastreabilidade de requisitos com base em metamodelos já existentes na literatura e a criação de uma ferramenta para auxiliar no Gerenciamento de Requisitos que implemente esse metamodelo e baseie-se na abordagem AADSP de forma que critérios como identificação, controle de mudanças e rastreabilidade dos requisitos sejam atendidos. Além disso, a ferramenta fornecerá um conjunto de *DataSets* para a área de Otimização da Engenharia de *Software* e será aplicada em empresas de desenvolvimento de *software* para avaliar o gerenciamento dos requisitos e espera-se que a mesma disponibilize aos interessados na confecção do *software* um acompanhamento dos requisitos durante todo o ciclo de vida do *software* e informações que permitam a gerência controlar o custo, tempo e qualidade do *software*.

Palavras-chave

Engenharia de Software; Engenharia de Requisitos; Gerenciamento de Requisitos; Rastreabilidade;

[*]Graduando em Análise e Desenvolvimento de Sistemas
[†]Doutor em Ciências da Computação e Professor do Curso de Análise e Desenvolvimento de Sistemas.

1. INTRODUÇÃO

A indústria do *software* mudou e mais importante do que desenvolver e entregar um *software* funcional, tornou-se obrigatória a entrega de *softwares* com qualidade e que adaptem-se rapidamente as mudanças solicitadas pelos clientes [1, 2, 3]. *Softwares* projetados com qualidade não representam apenas satisfação para o(s) cliente(s) mas também um considerável aumento na produtividade, redução dos custos e cumprimento de cronogramas estipulados. Nesse sentido, existem diversas propostas de soluções para auxiliar na qualidade de *software*, tais como modelos, padrões e métodos para o processo de desenvolvimento de software [4, 5].

A Engenharia de Requisitos é uma das disciplinas da Engenharia de *Software* que busca promover as melhores práticas e metodologias para a obtenção, análise, validação, gerenciamento e evolução dos requisitos dentro do ciclo de vida de um software [1, 2, 3]. O gerenciamento dos requisitos permite que os responsáveis pelo negócio consigam identificar os requisitos, quais são os *stakeholders*[1] do projeto, rastreamento dos requisitos em todos seus níveis de derivação e controlar as mudanças dos requisitos com a justificativa da referida mudança [4].

Tal gerenciamento proporciona melhorias na qualidade do *software* desenvolvido e auxilia nas tomadas de decisões preventivas e corretivas sobre os requisitos para que impactos negativos não comprometam o tempo, qualidade e orçamento do projeto [6, 7]. Para atender as variáveis de tempo, qualidade e orçamento, a área de Engenharia de *Software* conta com a *Search-Based Software Engineering* (SBSE), também conhecida como Otimização em Engenharia de *Software*, que tem como objetivo aplicar um conjunto de técnicas de seleção e otimização para problemas recorrentes da Engenharia de *Software* [8, 9].

Um dos problemas pesquisados pela **SBSE** está presente na Engenharia de Requisitos, mais especificamente o problema *The Next Release Problem* (NRP) [9]. O **NRP** consiste na seleção de um conjunto de requisitos para serem implementados em um determinado tempo do projeto de forma que as necessidades e expectativas dos clientes envolvidos no projeto sejam atendidas e o cumprimento de prazos e custos sejam mantidos [9].

Os requisitos são funcionalidades que os sistemas devem

[1]Conjunto de pessoas (clientes, gestores e desenvolvedores) interessadas no desenvolvimento do sistema e que muitas vezes são detentoras dos recursos e conhecimentos sobre regras de negócio.

implementar para atender necessidades do mundo real fornecidos por pessoas que possuem algum problema e desejam resolvê-lo com uso de *softwares* e uma de suas principais características é sua volatilidade e não previsibilidade [1]. Eles podem mudar durante todas as fases de concepção do *software* devido a fatores internos ou externos como melhorias, incompatibilidade, análises de risco, mudanças políticas, mudanças na tecnologia e outros [1].

Atualmente tem-se a convicção que mudanças em requisitos ao longo do processo de desenvolvimento de *software* fazem parte do processo e devem ser bem vistas pois elas objetivam o melhor para os clientes [2]. Segundo Malcher [5], a gerência de requisitos envolve um conjunto de atividades que apoiam a identificação, o controle e rastreamento de requisitos, bem como o gerenciamento de mudanças de requisitos já que estes podem sofrer modificações em qualquer momento ao longo do ciclo de vida do *software*. Já o Guia de Melhoria de Processos de *Softwares* Brasileiro (MPS.BR)[2], destaca que o propósito dessa gerência é o acompanhamento dos requisitos do produto e seus componentes e identificar inconsistências entre os requisitos, os planos do projeto e os produtos de trabalho do projeto [4]. Um dos critérios da gerência de requisitos é a rastreabilidade dos requisitos de um projeto de *software*.

O rastreamento de requisitos é utilizado para prover relacionamentos entre requisitos, fornecedores dos requisitos, arquitetura e implementação final do sistema e possibilita uma adequada compreensão dos relacionamentos de dependência entre os requisitos e os artefatos gerados para atender requisitos como diagramas, planos de teste, casos de uso e outros [2, 7, 4, 10, 1]. A rastreabilidade não tem sido adequadamente trabalhada por parte dos desenvolvedores devido à pouca flexibilidade das ferramentas disponíveis no mercado, o que obriga os desenvolvedores a registrarem manualmente as informações de rastreabilidade e que uma possível alternativa para solucionar esse problema consiste na apresentação de ferramentas mais poderosas, que possibilitem o registro das informações de rastreabilidade e sua integração com demais artefatos gerados por outras ferramentas utilizadas no processo de desenvolvimento [2, 7].

Como consequência da ausência do gerenciamento de requisitos a qualidade do *software* pode ser comprometida, pois os requisitos mudam durante todo o ciclo de vida do *software* [7]. O gerenciamento de requisitos proporciona um maior controle dos requisitos com a identificação, rastreamento em artefatos do *software* e o controle de mudanças, contudo, a implantação dessa gerência pode ser custosa e seus benefícios são visualizados apenas a longo prazo [1]. Como a concepção de uma ferramenta de gerenciamento de requisitos pode auxiliar os *stakeholders* do projeto a prever os impactos em qualidade, custo e orçamento gerados pelas mudanças nos requisitos de forma que não comprometam a entrega do *software*?

Esse projeto propõe a elaboração de uma ferramenta que implemente as práticas de gerenciamento de requisitos previstas pela literatura tais como identificação, controle de mudanças e rastreabilidade dos requisitos e as práticas previstas pela abordagem AADSP. Criada pelo grupo de pesquisa, LABRASOFT - IFBA [3], o AADSP propõe uma abordagem adaptativa para implantação do processo de *software*. A ferramenta será aplicada em empresas e instituições de desenvolvimento de *software* que estão em busca da contínua melhoria em seus produtos ou que desejam alcançar padrões de qualidade e certificações previstos pelo guia MPS.BR [4].

Este artigo está fragmentado em seções que discutem os seguintes tópicos:

1. **Seção 2**: São discutidos os conceitos de requisitos, a área de Engenharia de Requisitos, Gerenciamento e Rastreabilidade de Requisitos, os Metamodelos criados para auxiliar no Gerenciamento de Requisitos e a área de *Search-Based Software Engineering*;
2. **Seção 3**: Apresenta os trabalhos relacionados e um quadro comparativo entre funcionalidades presentes em outras ferramentas;
3. **Seção 4**: Apresentação a Solução Desenvolvida, Requisitos Funcionais e Não Funcionais elicitados para o desenvolvimento da ferramenta;
4. **Seção 5**: Discussão da arquitetura proposta para o sistema e as tecnologias utilizadas para a confecção do mesmo;
5. **Seção 6**: Apresenta os diagramas criados no processo de modelagem do sistema;
6. **Seção 7**: Apresenta a avaliação do trabalho e os resultados encontrados com base na aplicação de questionários;
7. **Seção 8**: Apresenta a conclusão do trabalho e os trabalhos futuros;

2. REFERENCIAL BIBLIOGRÁFICO

Esta seção apresenta os principais assuntos relacionados a este trabalho. A subseção 2.1 define os conceitos de requisitos de *software* e destaca o conjunto de boas características para requisitos. A subseção 2.2 apresenta a definição, fases e atividades referentes a Engenharia de Requisitos de *software*. O Gerenciamento de Requisitos e suas principais atividades são apresentados na seção 2.3 e a implantação e gerenciamento de requisitos de acordo com o guia MPS.BR na seção 2.3.1. A subseção 2.5 conceitua a rastreabilidade de requisitos e a subseção 2.6 apresenta as classificações da rastreabilidade dos requisitos. Já a subseção 2.7 elenca os aspectos positivos e negativos do uso da rastreabilidade. Metamodelos para auxiliar no processo de gerenciamento e rastreabilidade de requisitos são apresentados na subseção 2.9, analisados na subseção 2.9.4. O metamodelo proposto e utilizado para confecção da ferramenta é apresentado na subseção 2.10. E por fim a subseção 2.8 discute alguns dos conceitos da crescente área de Otimização em Engenharia de *Software* que foram adotados para a confecção desse projeto.

2.1 Requisitos de Software

Os requisitos representam o conjunto de funcionalidades que um *software* deve atender e são descobertos juntamente aos solicitantes do *software* (*stakeholders*) durante todo o ciclo de vida do *software* principalmente no início do projeto em entrevistas, *brainstorms*, questionários e outras atividades que buscam fomentar uma melhor compreensão dos problemas a serem resolvidos pelo *software* em questão [1, 3, 7]. Os requisitos de *software* devem possuir um conjunto

[2]https://www.softex.br/mpsbr/
[3]http://www.labrasoft.ifba.edu.br/

de características para que sejam considerados bons requisitos de forma que os mesmos possam trazer benefícios ao projeto.

De acordo com o guia MPS.BR [4] os requisitos representam uma capacidade requerida pelos solicitantes do *software* e que deve ser encontrada no produto para resolver um problema, alcançar um objetivo ou atender contratos, padrões, normas e especificações. Apesar dessas definições transmitirem claramente o conceito de requisitos, para Lopes [1] elas ainda são insuficientes para esclarecer aspectos não funcionais tais como desempenho, integridade, disponibilidade e segurança. Segundo Lopes [1] os requisitos podem ser classificados em requisitos funcionais e requisitos não funcionais [3]. Os primeiros são responsáveis por funcionalidades do *software* e o segundo grupo representa como o *software* deve se comportar em questões de desempenho, segurança, integridade e outros aspectos.

Para o **IEEE** [11] e **MPS.BR** [4], bons requisitos de *software* apresentam um conjunto de características para que possam ser avaliados pela equipe técnica para uma posterior avaliação dos clientes sendo elas:

1. **Identificação única**: um identificador único deve ser atributo aos requisitos para que eles possam ser identificados em qualquer projeto da empresa de forma que não possam ser confundido com outros requisitos, rastreados e promover reuso [4];

2. **Claro**: as especificações dos requisitos devem ser descritas em uma linguagem clara de forma que qualquer interessado no requisito consiga entender a funcionalidade que o requisito expressa [4];

3. **Não ser ambíguo**: a descrição do requisito não deve apresentar margem para mais de uma interpretação pelos interessados no projeto. A ambiguidade pode provocar divergências entre o que foi acordado e o que foi implementado no *software* [11];

4. **Relevante**: a implementação do requisito no sistema atenderá as expectativas dos *stakeholders* [4] e se todo requisito do documento será implementado pelo *software* [1, 11];

5. **Completo**: a funcionalidade que o requisito expressa está implementada no produto final e atende todas as expectativas dos *stakeholders* do projeto [11];

6. **Consistente**: o requisito foi submetido a analises, negociações e validações e está em conformidade com o interesse dos *stakeholders* [11] e não possui conflitos com outros requisitos [1];

7. **Classificação**: se existem indicações quanto o grau de importância do requisito para o projeto [11];

8. **Implementável**: as tecnologias para desenvolvimento *software* oferecidas no contexto no qual o *software* será desenvolvido atendem as expectativas dos *stakeholders* [1];

9. **Testável**: o requisito pode passar por práticas da Gerência de Teste de *Software* [4];

10. **Rastreável**: é possível rastrear as fontes do requisito, requisitos vinculados a ele, solicitantes, *rationale*, artefatos que foram originados a partir do requisitos [11];

Espindola [12] ressalta que uma grande quantidade de projetos de *software* são cancelados ou falham por não atenderem completamente as necessidades dos *stakeholders*. De acordo com Genvigir [6], requisitos são de extrema importância para o sucesso do *software* e que nenhuma outra parte do desenvolvimento de *software* é tão difícil de corrigir quantos os requisitos e para garantir o sucesso do projeto o guia MPS.BR [4] sugere uma boa comunicação com os clientes para garantir um entendimento uniforme das necessidades que o *software* atenderá. Para Sayão, Lopes, Ramesh e Toranzo [2, 1, 7, 10] os requisitos de *software* são voláteis, instáveis e propícios a mudanças durante o processo de desenvolvimento de *software* e tais mudanças não devem ser vistas como impedimentos e devem ser atendidas. Contudo, essas mudanças devem ser devidamente controladas e registradas para se possa estimar custos, prazos e impactos no projeto.

Para auxiliar os projetos de *software* a garantir qualidade, não ultrapassar custos e prazos de entrega previamente estipulados juntamente aos clientes, a Engenharia de Requisitos, subárea da Engenharia de *Software*, preocupa-se com a identificação dos requisitos, os envolvidos e suas necessidades, analise, classificação, documentação, delimitação de comportamentos, validação, evolução e gerenciar os requisitos durante o processo de desenvolvimento de *software* [3, 12, 13, 6] de forma que os requisitos sejam previamente conhecidos e controlados durante todo o processo de desenvolvimento.

2.2 Engenharia de Requisitos

O conhecimento correto dos requisitos evita que os *softwares* sejam desenvolvidos incorretamente, possuam funcionalidades inutilizáveis e/ou conflitantes e que atendam às reais necessidades e expectativas dos *stakeholders* [2]. A necessidade de desenvolvimento de um *software* tem como base a resolução de um dado problema do mundo real sendo assim, entender precisamente tal problema e como um *software* pode auxiliar é essencial para o sucesso do *software*. Os requisitos são elicitados inicialmente no projeto e conhecê-los adequadamente é uma das condições básicas para as fases posteriores da construção do *software* [6].

Para Lopes [1] a Engenharia de Requisitos é uma disciplina relacionada ao estudo dos requisitos, sua especificação, documentação, validação e negociação entre todos os *stakeholders* de um sistema. A Engenharia de *Software* cita diversos modelos de ciclo de vida de *software* tais como cascata, espiral, incremental, iterativo, evolucional, prototipação e outros. Esses modelos possuem as atividades de Análise, Projeto, Codificação e Teste [3, 1], sendo que as atividades de Análise e Projeto estão ligadas a atividades da Engenharia de Requisitos que aborda com maior importância e detalhamento os requisitos.

1. **Análise**: fase relacionada ao entendimento inicial do problema que o cliente deseja resolver com a criação do *software*. Essa fase é caracterizada pela obtenção dos requisitos. Segundo Lopes e Sommervile [1, 3] essa atividade visa especificar o problema a ser resolvido, declarando-o e entendendo-o com o auxílio de técnicas desenvolvidas no contexto da Engenharia de Requisitos;

2. **Projeto**: essa fase visa a descrição e avaliação do problema a ser solucionado com o *software*. Os esforços

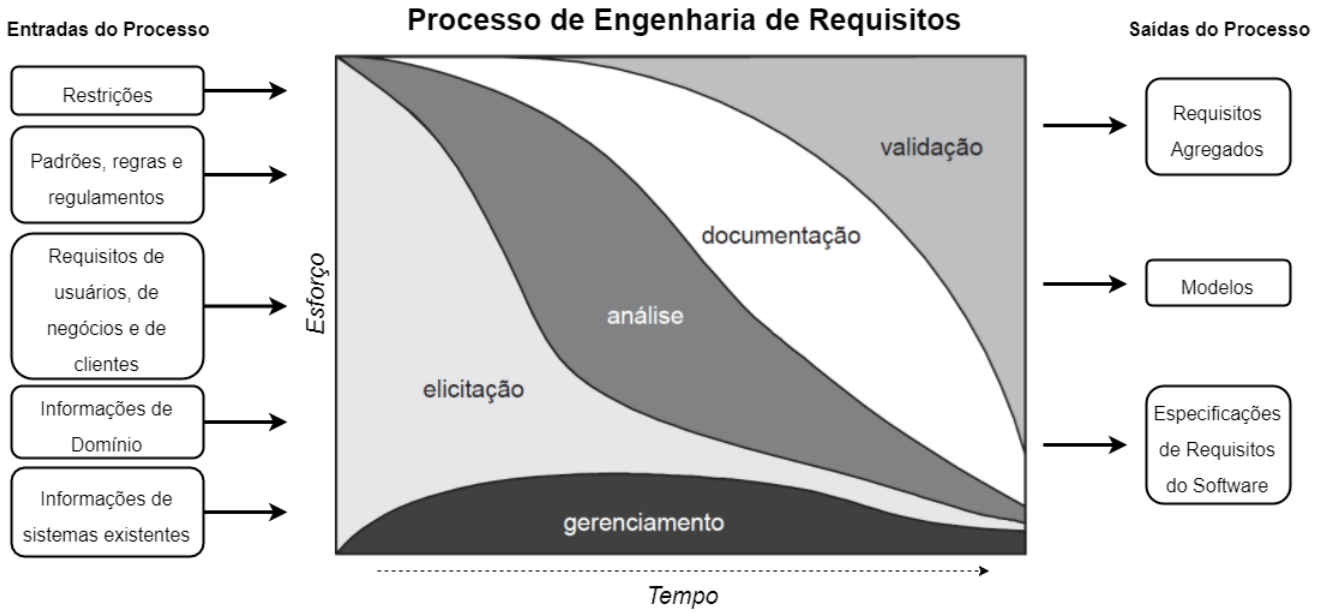


Figura 1: Processo Geral da Engenharia de Requisitos. Adaptado de [6]

são voltados a transformação dos requisitos iniciais obtidos com os clientes e uma posterior análise da qualidade, viabilidade antes da codificação [1, 3];

3. **Codificação**: visa construir o *software* com base nos requisitos coletados e analisados nas fases anteriores com a utilização de ferramentas computacionais [1, 3];

4. **Teste**: destina-se a verificar que o programa construído esteja de acordo com o projeto e com os requisitos levantados e documentados [1, 3];

De acordo com Genvigir [6], todos os modelos de ciclo de vida de *software* citados anteriormente tem início com a Engenharia de Requisitos, pois ela trata da identificação dos envolvidos e suas necessidades de descoberta dos requisitos e de documentação. Em seu artigo, Lopes [1] traz diversas definições de autores renomados na área da Engenharia de Requisitos e destaca que as atividades ligadas à essa disciplina ocorrem durante todo ciclo de vida do *software* desde as atividades iniciais (elicitação e análise dos requisitos) e as que perduram durante toda a sua vida útil (codificação, testes e evolução).

As atividades que contemplam a Engenharia de Requisitos são cinco: Elicitação, Análise e Negociação, Documentação, Validação e Gerenciamento [3, 1, 6]. Essas atividades fazem parte do processo da engenharia de requisitos (figura 1) que compreende na aquisição de um conjunto de dados como entradas, o processamento desses dados com as cinco atividades listadas anteriormente e as saídas que são os requisitos analisados e agregados ao sistema.

1. **Elicitação**: essa fase é caracterizada pela identificação dos requisitos do sistema com base em interações com os solicitantes do *software*. De acordo com Lopes [1] essa fase consiste na consulta dos *stakeholders*, análise da documentação, da análise de informações do domínio e/ou de estudos de mercado;

2. **Análise e Negociação**: avaliação da consistência, conflitos e inconsistências entre requisitos com relação a necessidade dos clientes. Para Genvigir [6], essa atividade tem como objetivo analisar os detalhes dos requisitos e verificar se os requisitos possuem conflitos se atendem as regras de negócio do cliente;

3. **Documentação**: os requisitos aprovados na fase anterior e que serão codificados devem ser documentados de forma clara e que não traga ambiguidade. Genvigir [6] destaca que "...os requisitos devem ser documentados a fim de servir de base para o restante do processo de desenvolvimento. Essa documentação deve ser feita de forma consistente, seguindo-se um padrão que permita demonstrar, em vários níveis de detalhes, a especificação dos requisitos levantados";

4. **Validação**: após a fase de documentação, os requisitos são validados com critérios baseados em consistência, completude, corretude e viabilidade [3, 1, 6].

5. **Gerenciamento**: essa atividade tem como objetivo o controle dos requisitos já elicitados, analisados e validados com os usuários e esses novos requisitos elencados serão agregados ao sistema [6]. Essa atividade é uma das mais importantes dentro da Engenharia de Requisitos pois acompanha o estado do requisito durante todo ciclo de vida do *software* [2, 1, 6];

O gerenciamento dos requisitos acompanha os requisitos desde a sua concepção junto aos *stakeholders* à sua evolução. Esse gerenciamento é essencial para o sucesso do projeto já que mantém o controle dos requisitos durante todo ciclo de vida do *software* auxiliando os *stakeholders* a prever impactos a custos, prazos e a qualidade do produto em questão.

2.3 Gerenciamento dos Requisitos

O gerenciamento de requisitos objetiva o acompanhamento dos requisitos de um *software* em todas as fases de concep-

ção do mesmo, desde a elicitação à evolução do *software* [3, 7]. Diversos autores e abordagens conceituam essa gerência e todos apontam a importância da mesma como fator determinante na qualidade do *software* entregue [7, 2, 1, 10, 6].

De acordo com Sommervile [3] o papel dessa gerência é o acompanhamento entre requisitos, relacionamentos entre requisitos, gerenciamento das dependências entre a documentação de requisitos e outros artefatos originados durante outros processos da engenharia de *software*.

Lengwell e Widrig, citados por [6], definem o gerenciamento de requisitos como um esforço sistemático para elicitar, organizar, e documentar um processo de engenharia de requisitos afim de estabelecer acordo entre clientes, usuários e o grupo de desenvolvimento no que tange às mudanças dos requisitos de um sistema.

A opinião dos autores convergem em um ponto em comum: o gerenciamento de requisitos pode tornar-se uma tarefa difícil e árdua. Os requisitos de *software* mudam constantemente e a documentação e atualização dos mesmos torna-se custosa [2, 1]. Atualmente, tem-se a convicção que mudanças em requisitos ao longo do processo de desenvolvimento de *software* fazem parte do processo [2]. Lopes [1] afirma que os requisitos possuem uma natureza volátil e são instáveis, fatores como mudanças na legislação, adaptações no requisito, atualizações tecnológicas, mudanças no mercado, mudanças na política da empresa e correção de requisitos entendidos incorretamente fazem com que os requisitos sejam modificados e possam afetar outros requisitos e artefatos.

Para que o gerenciamento de requisitos seja efetivo e traga benefícios ao projeto, ele deve ser implementado desde o início do projeto do *software*. Segundo Lopes [1] os benefícios dessa atividade não são visualizados imediatamente e que são "percebidos no médio prazo sendo que são necessários investimentos no curto prazo". Lopes [1] ainda destaca que a não implementação do gerenciamento de requisitos pode proporcionar economias de curto prazo mas que geram impactos em custos e prazos posteriores ao projeto. Visando garantir a qualidade e melhoria de processos de *software*, diversos guias foram desenvolvidos dentre eles o guia MPS.BR [4] [4] e o guia AADSP [14] [5].

2.3.1 Implantação e Gerenciamento de Requisitos com MPS.BR

A Associação para Promoção da Excelência do *Software* Brasileiro também conhecida como SOFTEX, com o apoio do Ministério da Ciência, Tecnologia e Inovação (MCTI), Financiadora de Estudos e Projetos (FINEP), Serviço Brasileiro de Apoio às Micro e Pequenas Empresas (SEBRAE) e Banco Interamericano de Desenvolvimento (BID/FUMIN), coordena o Programa MPS.BR. Criado em 2003, é um programa mobilizador e de longo prazo que tem como principal objetivo o aumento da competitividade das organizações através da melhoria nos processos de *softwares* e que seus modelos MPS sejam adequados a diferentes empresas e sejam compatíveis com padrões de qualidade aceitos internacionalmente. Para isso é necessário a implantação e contínuo aprimoramento dos Guias que contém uma descrição geral do modelo de referência com todas as definições necessárias para entendimento e aplicação [4].

[4]https://www.softex.br/mpsbr/

[5]http://www.labrasoft.ifba.edu.br/ wp-content/uploads/2016/10/Guia-AADSP.pdf

O MPS.BR possui três modelos de referência: MPS-SW voltado para Melhoria de Processos de Software, MPS-SV voltado para Melhoria de Processo de Serviços e MPS-RH voltado para Melhoria de Processo de Recursos Humanos. Segundo MPS.BR [4], o modelo MPS para *software* (MR-MPS-SW) tem como base os requisitos de processos definidos nos modelos de melhoria de processo e atende a necessidade de implantar os princípios de engenharia de *software* de forma adequada ao contexto das empresas, estando em conformidade com as principais abordagens internacionais para definição, avaliação e melhoria de processos de *software* [4].

O guia geral do MPS.BR divide os processos de melhoria em sete níveis de maturidade, sendo eles:

1. Nível G - Parcialmente Gerenciado;
2. Nível F - Gerenciado;
3. Nível E - Parcialmente definido;
4. Nível D - Largamente definido;
5. Nível C - Definido;
6. Nível B - Gerenciado quantitativamente;
7. Nível A - Em otimização;

Cada nível possui um guia específico para sua implementação e um conjunto de gerências que deve ser atendido para que a empresa possa alcançar o nível de maturidade desejado. O nível que será abordado nesse trabalho é o Nível G, Parcialmente Gerenciado, que possui as Gerência de Projeto (GPR) e Gerência de Requisitos (GRE) sendo que dessas gerências apenas a segunda será discutida.

Para o guia MPS.BR [4], a GRE deve gerenciar os requisitos do produto e identificar inconsistências entre os requisitos, planos de projeto e os produtos de trabalho do projeto. Essa gerência também deve controlar a evolução dos requisitos, controle de mudanças, revisão continua, documentar mudanças e suas devidas justificativas e manter a rastreabilidade.

Os seguintes resultados são esperados para a Gerência de Requisitos do MPS.BR:

1. **GRE1** - O entendimento dos requisitos é obtido junto aos fornecedores de requisitos;
2. **GRE2** - Os requisitos são avaliados com base em critérios objetivos e um comprometimento da equipe técnica com estes requisitos é obtido;
3. **GRE3** - A rastreabilidade bidirecional entre os requisitos e os produtos de trabalho é estabelecida e mantida;
4. **GRE4** - Revisões em planos e produtos de trabalho do projeto são realizadas visando a identificação e correção de inconsistências em relação aos requisitos;
5. **GRE5** - Mudanças nos requisitos são gerenciadas ao longo do projeto;

2.3.2 Implantação e Gerenciamento de Requisitos com AADSP

O LABRASOFT, Grupo de Pesquisa Laboratório de Desenvolvimento de Software, propõe uma abordagem adaptativa para implantação de processo de software em micro e pequenas empresas - MPEs soteropolitanas. A abordagem denominada de *Adaptive Approach for Deployment of Software Process* (AADSP) tem como alicerce práticas inovadoras em conformidade com o modelo MPS-BR, desenvolvido no Brasil pela Softex (SOFTEX, 2012), algumas práticas contidas em metodologias ágeis e do guia de conhecimento PMBOK da PMI (*Project Management Institute*) para gerenciamento de projetos [14].

Valores dos Artefatos

De acordo com o AADSP, artefatos são todas as formas de conhecimento produzido pelo homem registradas em algum meio físico ou lógico [14]. AADSP qualifica os artefatos documentais de acordo com seu grau de importância, desse modo esta abordagem busca implementar estes artefatos de forma adaptativa nas MPEs. Os três graus de importância são:

1. **Essencial**: Artefatos base para implementação do modelo AADSP, de modo que sua continuidade deverá ser garantida no processo de implantação desta abordagem;
2. **Importante**: Artefatos que são consideráveis, todavia não são obrigatórios, assim sua implementação resultará em resultados adicionais ao modelo;
3. **Desejável**: Artefatos pouco consideráveis, estes não implicam necessariamente na melhoria do processo ou em resultados satisfatórios;

Gerências do AADSP

Com base nas gerências propostas pelo guia [4], o AADSP catalogou seis gerências que tem como principal objetivo definir um conjunto de artefatos que garantam a qualidade do *softwares*.

1. **Gerência de Projetos**: Estabelecer e manter planos de trabalhos que definam atividades, cronogramas, recursos e os responsáveis pelo projeto [14];
2. **Gerência de Requisitos e Modelagem**: gerenciar os requisitos e componentes do projeto, controlar a evolução e identificar inconsistências entre os requisitos, os planos do projeto e os produtos de trabalho do projeto além de proporcionar uma visão comum entre a alta gerência e os *stakeholders* [14];
3. **Gerência de Configuração e Mudanças**: O escopo do projeto e os seus requisitos podem ser modificados a qualquer momento durante o ciclo de vida do *software*. Assim, requisitos adicionais podem ser incorporados no projeto, os requisitos podem ser removidos do projeto e/ou as mudanças podem ser feitas para os requisitos existentes [14];
4. **Gerência de Colaborares e Stakeholders**: Organizar, gerenciar, delegar responsabilidades para membros envolvidos na construção do *software* sejam eles desenvolvedores ou *stakeholders* que conhecem as regras do negócio [14].
5. **Gerência de Testes**: desenvolver planos e estratégias para implementação de teste e inspeção de *software* objetivando a melhoria na qualidade e integridade dos dados apresentados nos produtos finais [14];
6. **Gerência de Reuso**: fornecer maiores artefatos de usabilidade produzidos pelos projetos por mapeamento e documentação de componentes e outros ativos reutilizáveis de um software [14];

A seguir são listadas as praticas e artefatos previstos pelas gerências do AADSP citadas anteriormente com seu respectivo grau de importância definido pelo guia:

Gerência de Requisitos e Modelagem

1. Documento de requisitos do software (essencial);
2. Controle do comprometimento da equipe com os requisitos (importante);
3. Rastreabilidade dos requisitos (essencial)
4. Documentação do código contendo a identificação dos requisitos (desejável);
5. Plano de testes relacionado a identificação dos requisitos (importante);
6. Matriz de rastreabilidade (essencial);

Gerência de Configuração e Mudança

1. Solicitação de mudança (essencial);
2. Versionamento dos requisitos e artefatos;

Gerência de Colaboradores e Stakeholders

1. Lista de competência dos colaboradores (importante);
2. Registro dos *stakeholder* (essencial);
3. Controle da equipe execução (essencial);

A abordagem AADSP é voltada para a entrega artefatos, e deste modo, os processos, qualidade e controle de mudanças são comprovados a partir da existência e comprovação dos mesmos em projetos de *software* que implementem esta metodologia. Os artefatos exigidos pela gerência de requisitos garantem confirmidade entre o que foi implementado e as solicitações dos clientes, controle de mudanças, identificação dos requisitos e rastreabilidade dos mesmos. Diferente do guia [4], essa abordagem é mais flexível e pode proporcionar melhorias aos produtos entregues por micro e pequenas empresas e por *freelancers*.

Para que uma empresa possa implantar o gerenciamento de requisitos, independente do guia ou metodologia adotada, é necessário seguir um conjunto de atividades prescritos pela literatura. Essas atividades garantem controle, monitoramento, uniformidade e reuso dos requisitos.

2.4 Atividades do Gerenciamento dos Requisitos

O processo de Gerenciamento de Requisitos ocorre em paralelo com as fases da Engenharia de Requisitas elencadas anteriormente [6]. As atividades do Gerenciamento por Requisitos incluem identificação dos requisitos e suas ramificações, controle de mudanças dos requisitos e a rastreabilidade dos requisitos.

1. **Identificação**: Essa atividade tem como objetivo identificar e registrar os requisitos. É essencial que para cada projeto os requisitos tenham um identificador único. Essa identificação única permite que o requisito seja referenciado, versionado e gerenciável [3].

2. **Controle dos Requisitos**: consiste no acompanhamento de entrada de novos requisitos a serem integrados ao sistema bem como a alteração de requisitos já existentes [2, 3]. Os responsáveis pelo desenvolvimento do *software* conseguem analisar o impacto da inclusão ou alteração de requisitos no sistema e estimar possíveis aumentos ou redução em variáveis como custo, tempo e qualidade do *software* [6].

3. **Rastreabilidade**: Atividade mais importante do Gerenciamento de Requisitos, a rastreabilidade proporciona aos envolvidos responsáveis pela concepção do *software* um acompanhamento e mapeamento de todos os elos entre requisitos bem como dos produtos gerados a partir de um requisito ou um conjunto de requisitos [7, 2, 3].

A atividade de Rastreabilidade, destacada na figura 2, é a atividade central do gerenciamento de requisitos [6] e tal atividade é essencial para a construção de *softwares* com qualidade pois os *stakeholders* conseguem visualizar quais requisitos já estão implementados, conflitos entre requisitos, graus de dependência e quais impactos podem ser gerados com a modificação de um ou mais requisitos.

Por conta da sua importância, a rastreabilidade de requisitos será melhor discutida nas próximas subseções. Serão apresentados os principais conceitos, classificações, impactos positivos e negativos da aplicação da rastreabilidade em um projeto de *software*, os metamodelos propostos por alguns autores e o metamodelo proposto por esse artigo e que foi implementado na ferramenta.

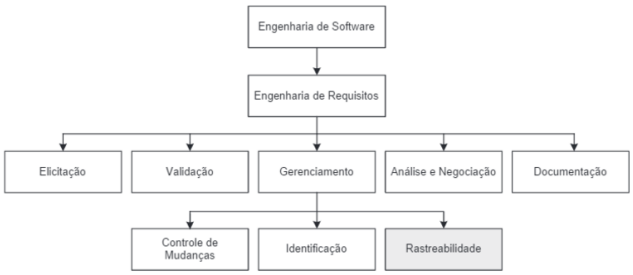


Figura 2: Área da Rastreabilidade dentro da Engenharia de Software. Extraído de [6]

2.5 Conceito da Rastreabilidade

A rastreabilidade de requisitos é utilizada para localizar todas as dependências e relacionamentos que deram origem ou foram originados a partir dos requisitos durante todo o ciclo de vida do *software* [7, 10, 2]. Os requisitos podem ser encontrados antes da sua implementação no *software* em atas de reuniões, entrevistas e outros métodos para elicitação de requisitos e posteriormente em artefatos documentais, componentes e código-fonte do *software*.

De acordo com o guia MPS.BR [4], a rastreabilidade é o relacionamento que pode ser estabelecido entre dois ou mais produtos do *software*, principalmente por produtos que tenham um vínculo do tipo mestre e subordinado. Além disso o guia destaca que a rastreabilidade de requisitos é fundamental para apoiar o processo de mudanças de requisitos.

Segundo Sayão [2], a rastreabilidade de requisitos é utilizada para promover o relacionamento entre os requisitos, a arquitetura do *software* e implementação final do sistema, permitir uma melhor compreensão dos relacionamentos existentes entre os requisitos do *software* e negociar novos prazos e custos a partir de mudanças com cliente. Sua implementação pode ser feita através de elos (ligações) entre os requisitos, dos requisitos a suas fontes e dos requisitos a sua implementação [2].

Em sua dissertação, Genvigir[6] diz que a rastreabilidade está diretamente associada ao processo de construção do *software* com a capacidade de estabelecer vínculos entre requisitos e outros artefatos de *softwares* como modelos, documentos, código-fonte e testes. Outras características da rastreabilidade ressaltadas por Genvigir [6] são: assistir o processo de verificação dos requisitos em um sistema, medir os impactos que a mudança em um requisito pode trazer ao projeto, compreender aspectos do projeto bem como a evolução de um componente do sistema e as motivações que levaram a modificações nos requisitos.

Para melhor compreender a rastreabilidade é necessário classifica-lá. A classificação da rastreabilidade dá-se com base nas informações em que deseja-se rastrear e o sentido/direção da rastreabilidade [2]. As informações que podem ser rastreadas são os requisitos, artefatos, vínculos entre os requisitos e entre requisitos e artefatos, origem dos requisitos (*stakeholders* ou documentos). Já a direção da rastreabilidade está diretamente ligada as fases do desenvolvimento de *software* em que um requisito está presente e as ligações entre requisitos e artefatos.

2.6 Classificação da Rastreabilidade

A rastreabilidade de requisitos pode ser classificada em dois tipos: quanto às fases que antecedem e sucedem a especificação dos requisitos representadas pela pré-rastreabilidade e pós-rastreabilidade [2] e quanto ao seu sentido/direção, representada pelas rastreabilidade horizontal (RH) e rastreabilidade vertical (RV) [4, 2].

2.6.1 Pré e Pós Rastreabilidade

A pré-rastreabilidade e pós-rastreabilidade, conforme ilustradas na figura 3, estão relacionadas ao ciclo de vida do requisitos antes de depois da fase de especificação dos requisitos [6, 1]. A pré-rastreabilidade registra o contexto em que os requisitos surgiram (reuniões, atas, *workshops*, entrevistas, questionários e outros) já a pós-rastreabilidade relaciona os requisitos a artefatos do sistema e código-fonte [2, 6].

De acordo com Genvigir [6], a principal diferença entre as duas são as informações que ambas são responsáveis. A pós rastreia um requisito a partir de um ponto de referência (especificação dos requisitos) aos artefatos oriundos dos requisitos contidos na especificação. Enquanto a pré cria um rastreamento entre as fontes dos requisitos (cliente, usuários, normas e padrões) aos artefatos gerados pelas primeiras fases da Engenharia de Requisito como a Elicitação, Análise, Validação e Documentação dos Requisitos [1].

2.6.2 Rastreabilidade Horizontal e Vertical

A aplicação da RH e RV, conforme apresentadas na ima-

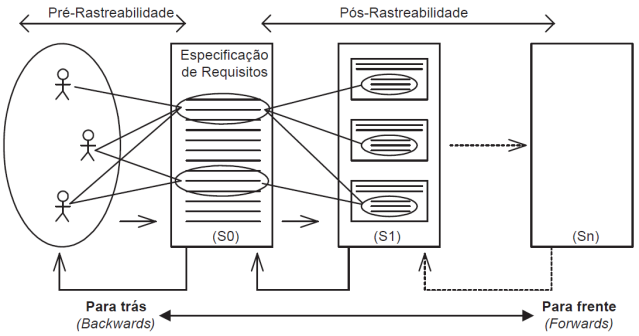


Figura 3: Pré e Pós Rastreabilidade. Extraído de [6]

gem 4, auxilia a determinar se os requisitos foram implementados e se os mesmos possuem uma fonte válida, além de fornecer ao responsável pelo gerenciamento do projeto informações precisas para negociar com o cliente mudanças no plano do projeto para atender a mudanças no requisitos de forma que desvios no cronograma e em custo sejam previamente estimados [4, 2].

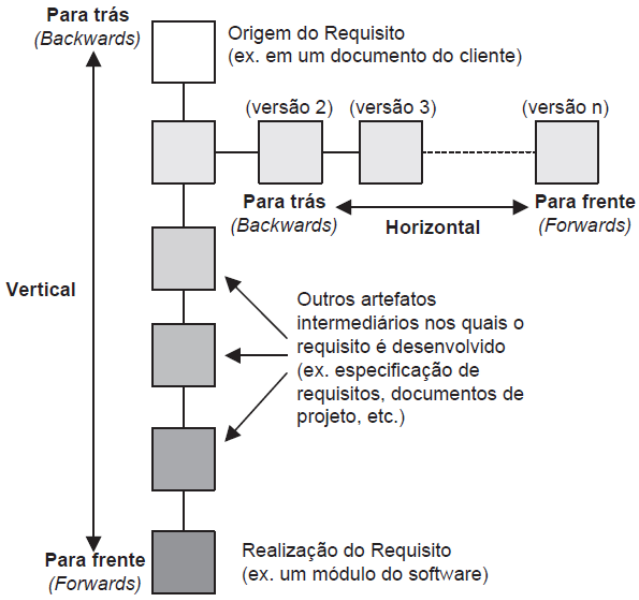


Figura 4: Rastreabilidade Vertical e Horizontal. Extraído de [6]

A RH, também conhecida como inter-rastreabilidade, é a forma de rastreabilidade entre diferentes versões do requisito e permite uma visualização de como os requisitos relacionam-se com outros requisitos. É também utilizada para obter conhecimento dos versionamentos de um requisito e verificar as dependências de um requisito em um mesmo nível ou fase do ciclo de vida do produto [6, 4]

Já a RV, também conhecida como extra-rastreabilidade, é realizada entre requisitos e artefatos produzidos pelo processo de desenvolvimento ao longo do ciclo de vida do projeto, ou seja, permite ver como os requisitos relacionam-se com os artefatos que são gerados no decorrer do projeto [6]. Essa forma de rastreabilidade proporciona o conhecimento de todos os refinamentos dos requisitos que foram produzidos ao longo do ciclo de vida do projeto [6]. De acordo com o guia MPS.BR [4], a RV estabelece um rastreio desde um requisito fonte aos níveis mais baixos de decomposição do produto como código-fonte e testes unitários e vice-versa.

2.7 Aplicabilidade da Rastreabilidade de Requisitos

Como citado anteriormente, os requisitos de um *software* possuem uma natureza volátil e podem ser modificados durante o ciclo de vida de *software* por diversos fatores, além disso, os responsáveis pela criação do *software* tem interesse em saber o estado atual dos requisitos e encontrar artefatos, estruturas, documentos e componentes que estão vinculados aos requisitos [4, 1]. Nesse contexto, a rastreabilidade de requisitos pode auxiliar no processo da gerência de projetos bem como no processo de desenvolvimento de *software*, contudo, a manutenção da rastreabilidade e a atualização da documentação vinculada a ela pode ser custosa em alguns cenários de desenvolvimento de *software*.

Conforme destacado pelo guia MPS.BR [4], a rastreabilidade tem o papel de determinar se todos os requisitos previstos para um *software* foram implementados e, além disso, permite que o gerente de projetos negociem com os clientes as mudanças nos requisitos de forma que os riscos no projeto, custo, cronograma e qualidade, não sejam comprometidos.

Diversos autores destacam os benefícios da rastreabilidade tais como:

1. Estimar variações em cronogramas e alterações de custos no desenvolvimento [2];
2. Verificar a alocação de requisitos a componentes de *software* [2];
3. Compreender o relacionamento entre requisitos, requisitos e artefatos, projeto e implementação[6];
4. Suportar a verificação de requisitos de um sistema [6]
5. Validar o *software* junto ao cliente [2];
6. Resolver de conflitos entre requisitos [2];
7. Compreender a evolução de um artefato [6];
8. Compreender aspectos do projeto [6];
9. Entendimento das razões por trás das decisões de projeto (*Design Rationale*) [6];
10. Analisar o impacto na evolução do sistema [2];
11. Reusar componentes [2];
12. Realizar a cobertura de testes [2];
13. Corrigir defeitos [2];
14. Garantir uma contínua concordância entre os requisitos dos interessados no sistema [6];

Contudo, em sua dissertação Sayão [2] destaca que a pouca flexibilidade de ferramentas disponíveis no mercado obriga os desenvolvedores a catalogar a rastreabilidade manualmente o que pode levar ao desuso da mesma.

Para Lopes [1] a rastreabilidade é um trabalho extenso, caro e vagaroso e que o principal problema da rastreabilidade está vinculado a grande quantidade informações que ela

pode gerar. E, assim como Sayão [2], ele enfatiza a importância do uso ferramentas para catalogar e associar elementos relacionados ao requisitos. Além disso, Lopes [1] destaca que o comprometimento da equipe técnica com atividade de rastreabilidade é fundamental para manter a rastreabilidade dos requisitos do *software* e que jamais deve ser vista como um processo impeditivo pois resulta em benefícios a longo prazo.

Em suas pesquisas, Ramesh [7] relata que os problemas relacionados a rastreabilidade estão nas maneiras como ela é aplicada, o mau uso de ferramentas e na escolha dos requisitos a serem rastreados. Devido a grande quantidade de informação que a rastreabilidade pode gerar [1, 2], os requisitos a serem rastreados devem ser escolhidos de forma que não comprometam prazos e custos do projeto. Além disso, Sayão [2] afirma que a prática incorreta ou incompleta da rastreabilidade na pré-rastreabilidade pode causar impactos posteriores ao projeto.

Com relação ao uso da rastreabilidade por parte de usuários (desenvolvedores), Ramesh [7] conduziu uma pesquisa sobre as influências organização na rastreabilidade e classificou os usuários de rastreabilidade em dois grupos, usuários avançados e mais sofisticados denominados de *high-end* e os usuários comuns e básicos denominados *low-end*.

O primeiro grupo, *high-end*, utiliza técnicas de rastreabilidade mais avançadas e veem a rastreabilidade como uma atividade fundamental para melhoria no processo de *software* e que trará benefício a longo prazo à todos envolvidos no projeto. Usuários desse primeiro grupo utilizam esquemas de rastreabilidade mais ricos, registram as motivações por trás das mudanças nos requisitos e preocupam com a evolução do *software*. Já o segundo grupo, *low-end*, utiliza técnicas de rastreabilidade mais simples, encaram a rastreabilidade como uma imposição dos gerente do projeto e registram de forma incompleta ou insuficiente a motivação por trás da criação ou mudança nos requisitos [7].

Apesar de apresentar uma implementação cara e vagarosa [1], a rastreabilidade pode trazer ótimos benefícios ao *software* produzido, a empresa e aos clientes mas deve ser bem aplicada. Escolher adequadamente quais requisitos devem ser rastreados e difundir a prática de rastreabilidade entre os envolvidos pode facilitar o gerenciamento dos requisitos [2]. Lopes [1] cita que a rastreabilidade é um fator cultural e que cada empresa deve incentivar que a equipe técnica empenhe-se na implementação da atividade vislumbrando benefícios futuros.

Tendo em vista os problemas da rastreabilidade, diversos autores desenvolveram metamodelos para auxiliar na atividade. Alguns metamodelos classificam as atividades processuais, informações a serem rastreadas e os tipos de vínculos entre requisitos e artefatos.

2.8 The Next Release Problem

Um projeto de *software* pode ter a participação de diversos *stakeholders* de diferentes setores e/ou áreas de conhecimento com diferentes níveis de interesse sobre o *software* em questão [2, 3]. Cada um desses *stakeholders* possui maior ou menor grau de importância para o bom desempenho e sucesso do projeto, critério esse determinado pela organização solicitante do *software* [1, 8]. Tais *stakeholders* podem solicitar funcionalidades à serem desenvolvidas para atender suas expectativas e necessidades do mundo real, logo, os requisitos de um projeto de *software* possuem diferentes níveis de interesse para cada *stakeholder* do projeto. Além disso, os requisitos solicitados pelos *stakeholder* do projeto devem possuir um custo financeiro que esteja dentro do orçamento total disponível para o projeto [3, 7].

Um dos problemas computacionais vinculados aos requisitos de *software* está atrelado a seleção de requisitos que devem ser implementados em um período de desenvolvimento de forma que a escolha desses atenda as necessidades dos clientes e ao atual custo do projeto [8]. Objetivando resolver esse impasse, o *The Next Release Problem* (NRP), busca estabelecer um balanceamento na seleção dos requisitos que serão contemplados em uma *release*[6] levando em consideração diversos fatores como custo, prazo, esforços, riscos, satisfação dos clientes e interdependência entre os requisitos através da aplicação de algoritmos genéticos multiobjetivos [9].

O **NRP** é um dos problemas abordados pela área de Otimização em Engenharia de Software, também conhecida como *Search-Based Software Engineering*, tem como objetivo aplicar um conjunto de técnicas de seleção e otimização para problemas recorrentes da Engenharia de *Software* [8].

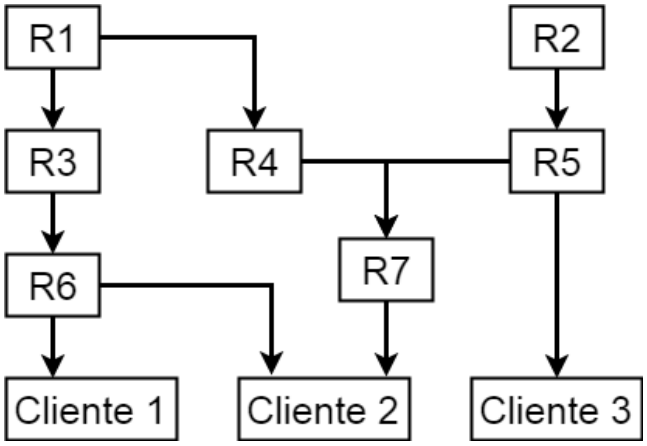


Figura 5: Exemplo do NRP. Adaptado de [8]

Para a aplicação dos algoritmos que objetivam a resolução do problema *Next Release Problem*, um conjunto de entradas parametrizadas e preestabelecidos deve ser disponibilizadas. Essas entradas seguem os critérios abaixo:

1. **Requisitos**: representam o conjunto de funcionalidades que devem ser atendidas pelo sistema em questão e que podem ser alocadas em uma nova *release*. O desenvolvimento do requisito exige a estimativa de um custo operacional. Para atender essa condição, esse trabalho considerou a criação de um atributo na classe *Requirement* denominado *cost*, conforme ilustrado na imagem 10 que representa o valor financeiro do requisito [8, 9];

2. **Clientes**: representam o conjunto de pessoas que estão interessadas no sucesso do projeto. Cada cliente possui um **grau de importância** para a organização ou para o projeto que deve variar de 1 (menor importância) à 10 (maior importância). Para atender essa condição, esse projeto representa os **clientes** como ***stakeholders*** e o grau de importância deles é

[6]Termo em inglês que refere-se a uma liberação de *software* para uso tal como uma nova versão oficial do mesmo

definido na classe ***StakeholderProject*** com o atributo ***importanceValue*** conforme ilustrado na figura 10 [8, 9];

3. **Clientes** *versus* **Requisitos**: Os clientes de uma organização ou projeto possuem diferentes interesses para cada requisito. Cada cliente pode atribuir uma importância para um requisito no qual ele possua interesse, sendo que, esse grau de importância, deve variar entre 0 (o cliente não tem interesse no requisito) à 9 (alta importância). Para atender essa condição, esse trabalho modelou a classe ***StakeholderRequirement*** com o atributo ***importanceValue*** conforme ilustrado na figura 10 [8, 9];

4. **Dependência e/ou Precedência entre Requisitos**: A criação dos elos entre requisitos permite que os clientes possam estabelecer quais requisitos são prioritários para o desenvolvimento bem como a dependência entre requisitos. Ao estabelecer quais requisitos devem ter prioridade em uma possível entrega, a satisfação do cliente é garantida. Para atender essa condição o trabalho propõe a criação de uma classe chamada *LinkBetweenRequirements* que possui como atributos o código do requisito de origem, o código de requisito de destino e o tipo do elo entre eles que pode ser de dependência e/ou precedência [8, 9];

A **SBSE** é uma das áreas de pesquisa da Engenharia de *Software* que está em crescimento e que apresenta algumas carências tais como a falta de dados parametrizados extraídos de fontes reais para aplicação de algoritmos para resolução de problemas, como por exemplo o já citado **NRP**, que é dos problemas do gerenciamento de requisitos. Visando auxiliar o crescimento dessa área, a ferramenta disponibiliza conjunto de *DataSets* gerados com base nos requisitos, *stakeholders* (clientes) e elos cadastrados na ferramenta nos formatos **CSV** e **JSON**.

2.9 Metamodelos para Rastreabilidade de Requisitos

De acordo com Genvigir [6], vários autores fazem o uso de modelos com base nas informações de um determinado domínio que desejam representar de forma gráfica ou textual. Para a rastreabilidade dos requisitos os modelos são utilizados para representar os diferentes tipos de elos que podem ocorrer entre requisitos e artefatos de *software* [7]. Para Sayão [2] esses elos possibilitam a identificação da origem de cada funcionalidade presente no sistema de acordo com a necessidade dos clientes, verificação e validação da alocação dos requisitos solicitados no *software* desenvolvido e conformidade dos testes com os requisitos.

Figura 6: Artefatos conectados através de um elo. Extraído de [6]

Conforme ilustrado na figura 6, o elo é o estabelecimento de um relacionamento direto entre um artefato de origem e um artefato de destino [6] e eles são os principais recursos para manter e representar os relacionamentos da rastreabilidade além de serem utilizados em diversas áreas da engenharia de *software* tais como a engenharia de requisitos nas atividades de validação, análise, evolução e referência cruzada entre requisitos e artefatos, na gerência de projetos com a análise de custos, restrições e impactos, na evolução do *software*, nos testes e na gestão de qualidade [2].

2.9.1 Metamodelo proposto por Ramesh

Ramesh [7] desenvolveu um metamodelo de rastreabilidade que pode ser visualizado na figura 7 baseado em uma longa pesquisa de forma que esse modelo pudesse representar a rastreabilidade de forma generalista. De acordo com Sayão [2], o metamodelo proposto por Ramesh [7] permite a captura de informações relacionadas a fontes (*source*), interessados ou envolvidos (*stakeholders*), objetos ou artefatos (*objects*) [7, 6].

As fontes podem ser classificadas como os documentos que remetem a origem dos requisitos tais como normas, padrões, editais, atas de reunião, regras institucionais, leis e medidas governamentais. Já os interessados podem ser os solicitantes do *software* que propõe melhorias e novas mudanças bem como os engenheiros e desenvolvedores interessados na rastreabilidade dos requisitos e por fim os objetos são todos os artefatos gerados durante o processo de desenvolvimento tais como código-fonte, arquitetura, casos de testes, documentos de requisitos e outros [7, 6, 2].

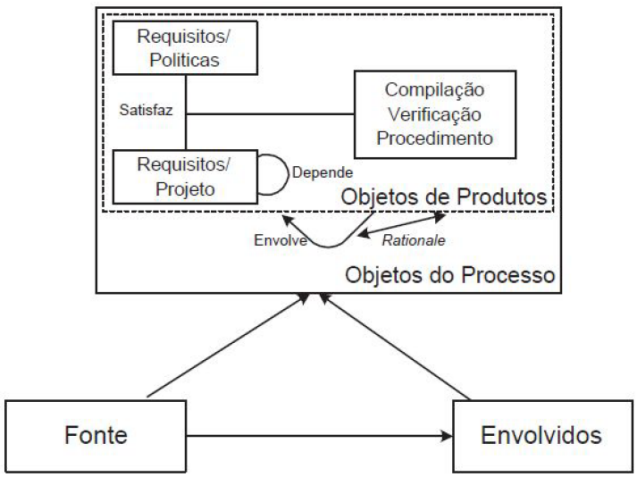


Figura 7: Metamodelo proposto por [7] adaptado por [6].

Em sua pesquisa, Ramesh [7] destacou que mesmo havendo uma grande quantidade de elos de rastreabilidade, eles podem ser agrupados em dois grupos básicos: produtos e processo. Os elos do primeiro grupo são divididos em satisfação e dependência e descrevem as propriedades e os relacionamento entre os objetos [6]. Já o segundo grupo que é dividido em elos de evolução e *rationale* tem o objetivo de registrar o histórico de ações e mudanças no processo [2].

2.9.2 Metamodelo proposto por Toranzo

Em sua proposta para melhoria da rastreabilidade entre requisitos, Toranzo [10] propõe a classificação das informações a serem rastreadas e um metamodelo que pode ser visualizado na figura 8 para auxiliar no rastreio do requisito, além de um metamodelo intermediário e um processo de

rastreabilidade que não serão abordados neste artigo.

Através deste metamodelo, Toranzo [10] propõe seis tipos de elos: satisfação, recurso, responsabilidade, representação, alocação e agregação.

1. **Satisfação**: a classe de origem tem dependência de satisfação com a classe de destino [10];
2. **Recurso**: indica que uma classe de origem têm dependência de recurso com a classe destino [2, 6]. Para Toranzo [10] recurso é um elemento que representa uma informação tais como motivações por trás de mudanças em requisitos;
3. **Responsabilidade**: representa a participação, responsáveis e conjuntos de ações das pessoas sobre os artefatos de *software* gerados [2, 6, 10];
4. **Representação**: Capturar a representação dos requisitos em linguagens sejam elas de programação ou modelagem de dados;
5. **Alocação**: similar ao elo de satisfação, esse elo representa a dependência de uma classe de origem com uma classe de destino que representa um subsistema [2];
6. **Agregação**: visualizar se um elemento é composto por outros elementos [10];

Outra contribuição importante na proposta de Toranzo [10] é a classificação do conjunto de informações que devem ser rastreadas para melhor entender a atividade de rastreamento dos requisitos. Ele propôs a classificação dessas informações em quatro níveis: ambiental, organizacional, gerencial e desenvolvimento.

As informações do nível ambiental pode contemplar as informações de onde o *software* será desenvolvido, contexto organizacional da empresa [2], contexto econômicos, políticos e padrões (externos) [10]. O nível organizacional agrega informações como missão, objetivos, metas, técnicas internas e padrões (internos) [2]. As informações do nível gerencial relaciona os requisitos com as tarefas realizadas para atender os mesmos e um acompanhamento e controle dos requisitos [10]. E o último nível, desenvolvimento, representa os artefatos tais como documentos de requisitos, arquitetura, diagramas, código-fonte, casos de teste e outros que são produzidos durante a construção do *software*.

2.9.3 Metamodelo proposto por Genvigir

Para solucionar os problemas acima citados, Genvigir [6] propõe um metamodelo que pode ser visualizado na figura 9 baseado na generalização de todos os tipos de artefatos e elos que possam participar do processo de rastreabilidade. Seu modelo busca possibilitar a definição dos diferentes tipos de artefatos que podem ser gerados durante o desenvolvimento do *software* com a inclusão de atributos aos elos.

1. **Tipo de Artefato**: Classificação de um grupo de artefatos que possuem mesmas características tais como casos de uso, arquitetura, classes, módulos e outros;
2. **Atributos do Tipo**: Para cada Tipo de Artefato podem ser criados diversos atributos que tem o objetivo de detalhar seu comportamento ou suas propriedades;
3. **Artefato**: Reflete o artefato criado para o *software* e que será rastreado;
4. **Instâncias**: São os valores reais atributos para um artefato;

Um aspecto similar entre o modelo proposto por Genvigir e Ramesh [6, 7] é a simplicidade e o fácil entendimento do modelo, contudo, o modelo de [6] é generalista e não define os elos, permitindo que os usuários da rastreabilidade possam adaptar o modelo ao seu processo de desenvolvimento podendo criar artefatos e elos durante todas as fases do projeto além disso, tal modelo possibilita a adição de novos atributos a um Tipo de Artefato durante qualquer fase do projeto.

Para a criação de elos entre os requisitos e artefatos do sistema, Genvigir [6] propõe a criação de um Tipo de Artefato denominado Elo que possui três atributos Código da Origem, Identificador e Código de Destino. Os códigos de origem e destino armazenam os identificadores de requisitos e/ou artefatos que deseja-se criar um rastro. E o atributo Identificador serve para classificar o tipo do elo criado.

2.9.4 Analise dos Metamodelos propostos por Ramesh, Toranzo e Genvigir

Sayão e Genvigir [2, 6] realizaram uma breve análise dos modelos de Ramesh e Toranzo [7, 10] citados anteriormente e concluíram que em certos momentos os metamodelos apresentam aspectos similares e contribuições diferentes para a literatura.

De acordo com Sayão [2], os elos de satisfação criados por Ramesh [7] podem ser equiparados aos elos de alocação criados por Toranzo [10]. Ambos tem o propósito de indicar que o requisito está alocado em um sistema. Outra similaridade entre os trabalhos encontrada por Sayão [2] está nos elos de dependência. Os elos de dependência criados por Ramesh [7] podem ser equiparados aos elos de recurso, satisfação ou os de agregação criados por Toranzo [10]. Todos esses elos tem como objetivo mostrar um relacionamento direto ou indireto entre recursos, classes/componentes e elementos do *software*.

Além disso, Sayão [2] destaca que a maior contribuição da proposta de Ramesh [7] é a simplicidade do modelo e a criação de elos voltados para a fase de evolução dos requisitos de forma que possam ser registradas as alterações do *software* e as motivações para tais mudanças (*Design Rationale*). E que a contribuição do trabalho de Toranzo [10] é a atenção empenhada no aspectos gerenciais do projeto com a criação de elos de responsabilidade e classificação do ambiente, já que o gerenciamento dos requisitos é uma atividade interligada a gerência do projeto e que pode auxiliar na redução impactos negativos aos custos, cronogramas e otimização das atividades.

No ponto de vista de Genvigir e Lopes [6, 1], Ramesh [7] realizou uma importante classificação dos usuários de rastreabilidade em *high-end* e *low-end users*, citados anteriormente neste artigo, e concordam com [2] no tocante a importância do aspecto gerencial abordado no trabalho de [10] com a criação do elo de responsabilidade para rastrear os *stakeholders* do projeto. Porém Genvigir [6] cita alguns pontos negativos em ambos trabalhos:

1. A predefinição dos elos que devem ser rastreados através de padrões predefinidos em grupos de elos;
2. Representação dos elos em matrizes de rastreabilidade torna-se incompleto uma vez que deve ser criada uma matriz para cada tipo de elo;

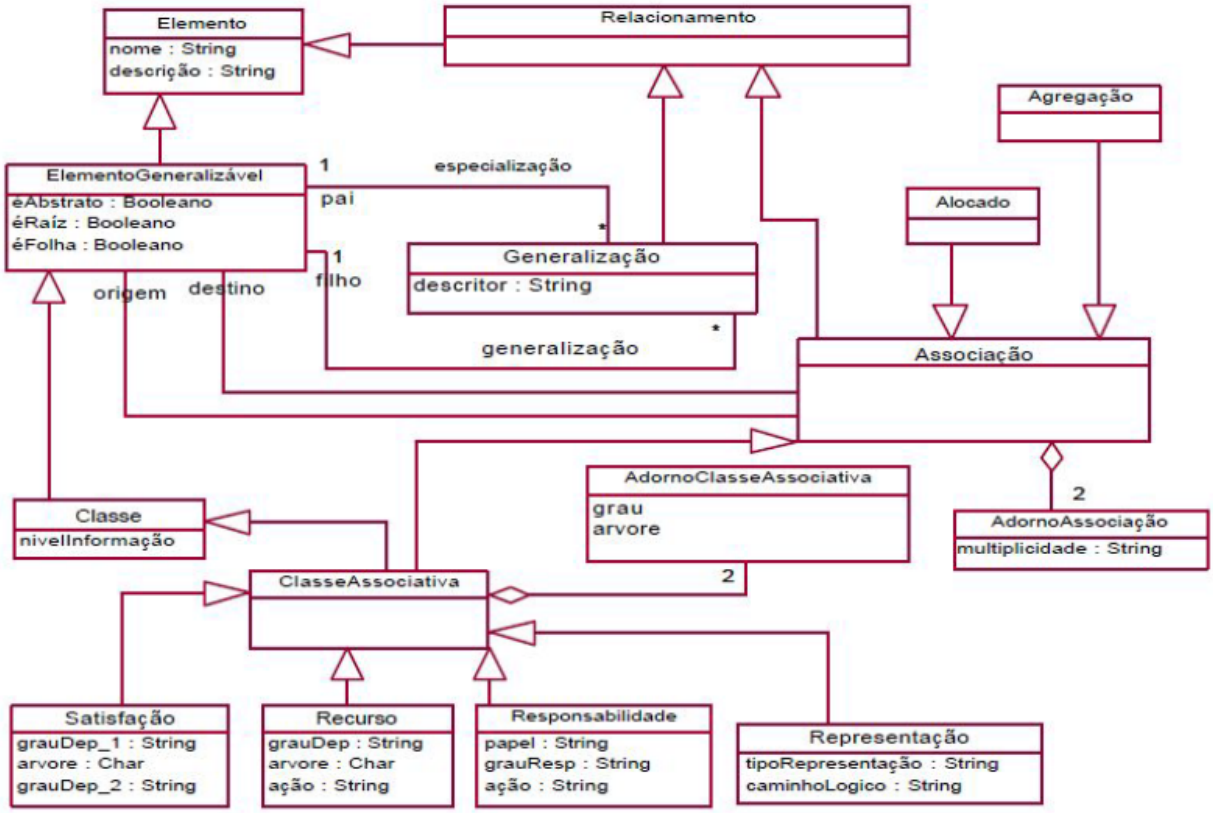


Figura 8: Metamodelo proposto por Toranzo. Extraído de [10]

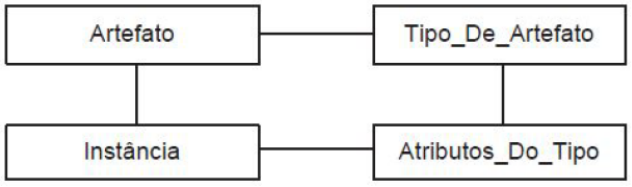


Figura 9: Metamodelo proposto por Genvigir. Extraído de [6]

3. Impossibilidade de adicionar atributos aos elos para enriquecimento da rastreabilidade;

O modelo proposto por Genvigir [6] apresenta uma alta flexibilidade para a criação de novos tipos elos e de artefatos de acordo com a necessidade da empresa diferente dos modelos de Ramesh [7] e Toranzo [10] no qual os elos são predefinidos. O autor justifica a que essa generalização dos elos e artefatos se da pela agregação de novos campos para enriquecer os dados de um elo ou de um artefato e a alocação de novos tipos de artefatos ao projeto.

Um dos problemas encontrados no modelo proposto por Genvigir [6] está relacionado a falta de flexibilidade para criação de atributos apenas para os artefatos sendo possível apenas a inclusão de atributos aos tipos dos artefatos. Consequentemente, todo artefato criado poderá possuir todos os atributos vinculados ao tipo. Como por exemplo, um artefato é criado e a ele é atribuído o Tipo "Requisito". Caso esse Tipo possua um atributo chamado Versionamento, todos os artefatos criados a partir desse tipo poderão ter instâncias do atributo Versionamento sendo que nem todos os Requisitos de um *software* são versionáveis.

Outro problema está na possibilidade de geração de inconsistências em bases de dados caso o modelo proposto por Genvigir [6] seja aplicado. De acordo com o modelo proposto pelo autor e por exemplos em sua dissertação, o elo entre requisitos e artefatos pode ser mantido através de um Tipo de Artefato denominado "Elo" que possui dois Atributos do Tipo denominados "Código de Origem" e "Código de Destino". A Instância desses Atributos do Tipo recebem no campo "Valor" a chave primária da tabela Artefatos (que nesse contexto está representando um requisito e um artefato) criando assim o elo de rastreabilidade. Contudo, o campo Valor não é uma *Foreign Key* da tabela Artefato, possibilitando assim que um artefato possa ser excluído e o banco não aplique regras básicas de integridade de dados providas por SGBDs.

2.10 Metamodelo Proposto

O metamodelo proposto, conforme figura 10 baseia-se nas principais contribuições dos metamodelos acima citados e em conceitos do Gerenciamento de Requisitos apresentados nas seções anteriores.

1. **Project**: Representa um projeto de *software* que possui as informações ambientais e organizacionais para auxiliar no desenvolvimento do projeto Toranzo [10]. Os projetos devem possuir um custo que será particionado entre os requisitos do *software*;

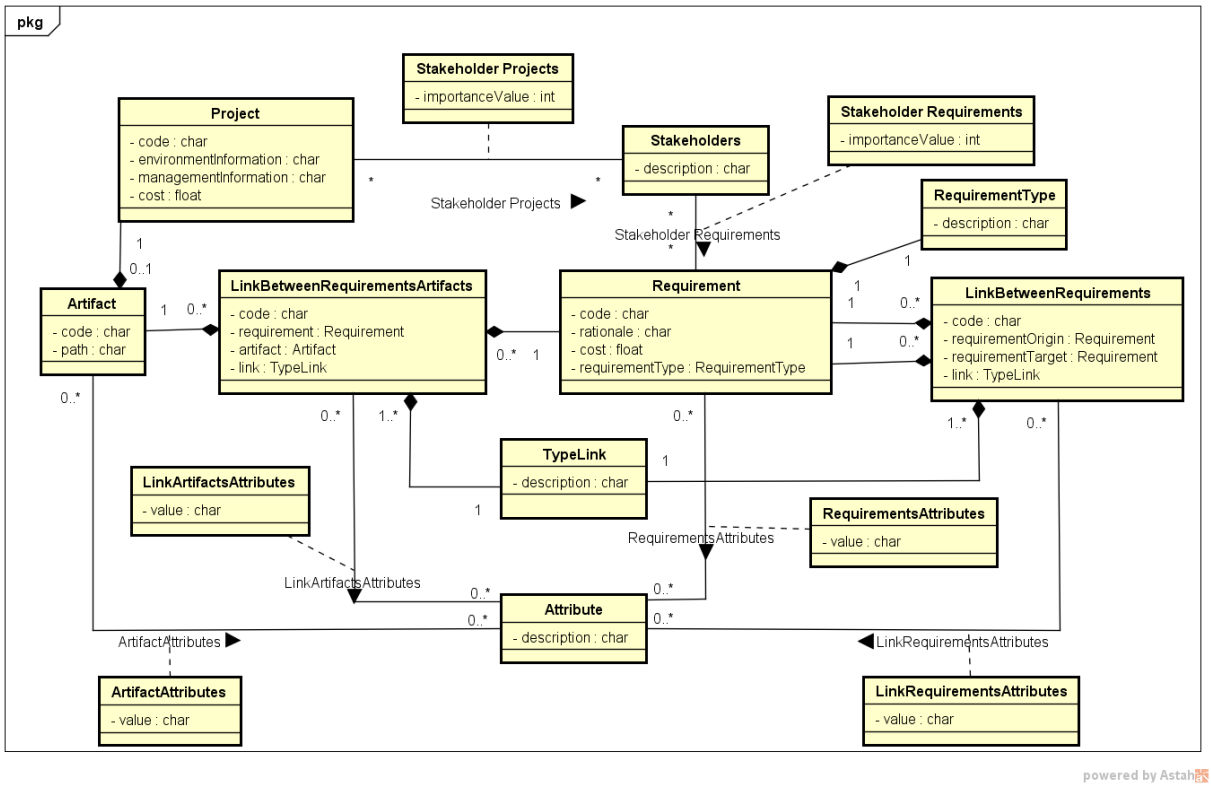


Figura 10: Metamodelo proposto para a ferramenta

2. **Requirement**: Representa um requisito do *software* que será passível de gerenciamento. Cada requisito deve possuir uma descrição do *rationale* por trás da criação do mesmo [2, 3, 7]. O custo de cada requisito deve estar dentro do valor estipulado pelo projeto;

3. **LinkBetweenRequirements**: Representa um elo entre dois requisitos. Esse elo deve possuir um tipo de elo representado pela classe ***TypeLink***.

4. **TypeLink**: Representa o conjunto de elos disponíveis e que foram elencados no trabalho de [1];

5. **RequirementType**: Classifica os tipos de requisitos (funcionais, não funcionais e outros);

6. **Stakeholders**: Classifica os *stakeholders* do projeto que podem estar vinculados a projetos e a requisitos;

7. **StakeholderProjects**: Vinculo entre um *Stakeholder* e um Projeto. Cada *stakeholder* tem um valor de importância para o projeto que pode variar de 1 a 9;

8. **StakeholderRequirements**: Vinculo entre um *Stakeholder* e um Requisito. Cada *stakeholder* tem um valor de importância para o requisito que pode variar de 0 a 9;

9. **Artifact**: Representa um artefato do *software* que possui um identificador único. O diretório lógico do artefato deve ser informado para que o mesmo possa ser rastreado.

10. **LinkBetweenRequirementArtifacts**: Representa um elo entre um requisito e um artefato do *software*. Assim como o elo entre requisitos, este deve possuir um tipo de elo representado pela classe ***TypeLink***;

11. **Attribute**: Representa um atributo que poderá ser utilizado pelas classes que a agregarem. Essa classe foi criada baseada no conceito de atributos para entidades proposto por [6];

As classes *Project, Requirement, LinkBetweenRequirements, Artifacts* e *LinkBetweenRequirementArtifacts* devem possuem um identificador único representados através do atributo de classe *code* para que aspectos como rastreabilidade, reuso de código, não ambiguidade e outros possam ser alcançados.

Requirement, LinkBetweenRequirements, Artifacts e *LinkBetweenRequirementArtifacts* podem possuir um conjunto de atributos específicos que ajudam no enriquecimento da rastreabilidade de informações [6]. Esses atributos são representados pelo campo *value* pertencente as classes-associativas que são geradas a partir da relação entre as quatro classes acima citadas com a classe *Attribute*.

3. TRABALHOS RELACIONADOS

Nesta seção são apresentadas e analisadas as funcionalidades de ferramentas confeccionadas para a área de Gerenciamento de Requisitos. Um grupo de pesquisa da Universidade Federal de Goías (UFG) [15] fez uma análise de ferramentas que apoiem a gerência de projetos e a gerência de requisitos que fossem adequadas às necessidades estabelecidas no processo de desenvolvimento e evolução do *software* da UFG.

Para tanto esse grupo [15] fez a análise de onze ferramentas sendo quatro delas de *software* livre e sete proprietárias conforme a representado na tabela 1. Essa tabela foi adaptada e a primeira coluna desta refere-se a ferramenta proposta por esse trabalho.

3.1 Ferramentas

Apesar do trabalho do grupo de pesquisa da UFG [15] fazer a analise de onze ferramentas, esse trabalho selecionou apenas cinco ferramentas para uso e experimentos.

1. **Enterprise Architect**[7]: Criada pela Sparx Systems, essa ferramenta auxilia na visualização e gerenciamento dos requisitos e na integração dos mesmos com o ambiente de desenvolvimento. A ferramenta também possui a modelagem de Casos de Uso da UML, importação de requisitos com base no formato CSV, a rastreabilidade de requisitos na implementação do *software*, analise de impacto em modificações, na comunicação entre os *stakeholders* e a criação de um dicionário de comunicação para ajudar na comunicação.

2. **Controla**[8]: Ferramenta de apoio ao gerenciamento de requisitos resultante de estudos comparativos entre as metodologias de gerenciamento de projetos. Essa ferramenta possibilita a elicitação dos requisitos junto aos *stakeholders*, detalhamento, gerenciamento de mudanças, controle de versões, matriz de rastreabilidade, avaliação de impacto e outras.

3. **AvenqoPEP**[9]: Voltada para o gerenciamento de requisitos, essa ferramenta permite o cadastro de requisitos com classificação de riscos, *status* do requisito, anexar arquivos ao requisito e a criação de relacionamento entre requisitos (elos). Um dos diferenciais dessa ferramenta é a sinalização de impactos quando um requisito ligado a outro é alterado e a associação de tarefas aos requisitos.

4. **Caliber**[10]: Criado pela Micro Focus[11], *Software* que fornece um gerenciamento dos requisitos com o uso de uma abordagem iterativa e colaborativa para definir e gerenciar requisitos em todas as fases da Engenharia de Requisitos de forma que os requisitos estejam em conformidade com as necessidades dos clientes. Assim como algumas outras ferramentas, o Caliber fornece visualização dos requisitos, rastreabilidade e análise de impacto em tempo real

5. **IBM Rational DOORS**[12]: Solução para gerenciamento de requisitos que ajuda a empresa a gerenciar o escopo e custo do projeto. Permite a captura, rastreio, gerenciamento de mudanças e a criação de casos de testes para os requisitos cadastrados na ferramenta.

[7]http://www.sparxsystems.com.au/products/ea/requirements.html
[8]http://www.periodicosibepes.org.br/index.php/reinfo/article/viewFile/156/48
[9]http://www.avenqo.com/
[10]https://www.microfocus.com/pt-br/products/requirements-management/caliber/
[11]https://www.microfocus.com/pt-br/
[12]https://www-03.ibm.com/software/products/pt/ratidoor

3.2 Comparativo entre as ferramentas

Como já citado anteriormente, a tabela 1 foi extraída e adaptada do trabalho de [15]. A análise das ferramentas foi fundamentada com base na documentação encontrada nos *sites* bem como em versões *trial*.

Conforme apresentado na tabela 1, diversas são as soluções criadas para o gerenciamento de requisitos e algumas delas são integradas com o gerenciamento de projetos. Este trabalho propõe a criação de uma ferramenta para gerenciamento de requisitos com base na abordagem AADSP que fornece um modelo alternativo para o gerenciamento de requisitos de forma que todos os critérios estabelecidos por tal gerência sejam atendidos.

Nessa ferramenta destacam-se as funcionalidades de criação de requisitos, criação de *templates* para acelerar o cadastro dos requisitos, matriz de rastreabilidade entre requisitos, matriz de rastreabilidade entre requisitos e artefatos, criação de atributos para requisitos, artefatos e para os elos, versionamento dos requisitos com implementação do *Design Rationale*, impressão do documentos de requisitos e outras.

4. SOLUÇÃO DESENVOLVIDA

A ferramenta T-AADSP-Requirements tem como objetivo auxiliar *stakeholders* de um projeto com o gerenciamento de requisitos. Inclusão de requisitos, controle de mudanças, criação de vínculos entre requisitos e entre requisitos e artefatos tais como casos de uso, documentos de teste e outros além do rastreio do requisito em código-fonte com base no seu identificador único gerado pela própria ferramenta.

4.1 Requisitos Funcionais

4.1.1 Entrar no Sistema

Para utilizar o sistema o usuário deve fornecer o *login* e a senha que foram previamente cadastrados por um administrador do sistema com as devidas permissões e enviados para seu *e-mail* pessoal. O primeiro usuário de acesso para o sistema já deve ser cadastrado durante a implantação da ferramenta na empresa solicitante. O *login* desse usuário deve ser "admin" e a senha deve ser "123456".

4.1.2 Sair do Sistema

O usuário pode encerrar suas atividades no sistema ao clicar no Botão *LogOut*. Esse botão deve estar disponível em todas as páginas do sistema e em um lugar visível e fácil acesso.

4.1.3 Cadastrar Usuários

Os gestores do sistema que possuem as devidas permissões de Administradores podem efetuar o cadastro de novos usuários para o sistema. Após efetuar o cadastro de um usuário um *e-mail* deve ser enviado para o usuário com as informações básicas para acesso ao sistema.

4.1.4 Cadastrar Perfis de Acesso

Os gestores do sistema que possuem as devidas permissões de Administradores podem efetuar o cadastro de novos permissões de acesso. Os perfis de acesso contém o conjunto de ações que os usuários do sistema podem efetuar.

4.1.5 Vincular Usuário ao Perfil de Acesso

Tabela 1: Analise de Ferramentas para Gerência de Requisitos (adaptado de [15])

Característica	1	2	3	4	5	6	7	8	9	10	11	12
Estabelece relacionamentos entre cada requisito e suas fontes (Fornecedores de Requisitos associados)	S	NA	P	P	N	NA	NA	NA	NA	NA	S	S
Estabelece dependências entre Requisitos Funcionais (incluindo Regras de Negócio), Casos de Uso, Requisitos Não-Funcionais, Requisitos de Dados e Telas	S	S	P	S	S	S	S	S	S	S	S	S
Estabelece dependências entre os requisitos e os artefatos de design	S	S	N	S	S	S	NA	S	S	S	S	S
Estabelece dependências entre os requisitos e os artefatos de implementação	S	S	S	S	S	NA	NA	S	S	S	S	S
Dado um requisito, ou qualquer outro artefato, mostra as dependências do artefato em questão com o restante dos artefatos	S	S	N	S	S	NA	NA	S	S	S	S	S
Permite definir requisitos e casos de uso na própria ferramenta (verificar se há *template* para definição destes artefatos)	S	S	S	S	P	S	S	NA	S	S	S	N
Estabelece e indica dependências entre produtos gerados (dado um sistema, mostra as suas dependências com os demais sistemas)	S	NA	N	S	S	NA	NA	NA	NA	NA	P	S
É fácil identificar na Matriz de Rastreabilidade o requisito em questão	S	NA	S	S	S	NA	S	NA	NA	S	S	N
Divulga a Matriz de Rastreabilidade para todos os interessados	S	S	N	S	N	NA	NA	S	S	NA	S	N
Evidencia requisitos que não estão sendo implementados em qualquer artefato de projeto ou de implementação	S	NA	N	S	N	NA	NA	NA	NA	NA	N	S
Evidencia artefatos que não estão associados a requisitos	S	NA	N	S	N	NA	NA	NA	NA	NA	S	S
Verifica a existência de integração entre ferramentas de design e de implementação	S	S	N	S	N	NA	NA	S	S	NA	N	S

Legenda:
Ferramentas – (1) T-AADSP-Requirements; (2) Caliber;(3) Controla; (4) Enterprise Architect; (5) ReqManager; (6) Doors; (7) RequisitePro; (8) Nant; (9) Star Team; (10) Team System; (11) Avenqo; (12) OpenShore.
Avaliação – (S) Satisfaz o requisito em questão; (N) Não atende o requisito; (P) Atende Parcialmente o requisito; (NA) O requisito em questão Não foi Avaliado; (NE) A informação para avaliação do requisito Não foi Encontrada.

Os gestores podem vincular um perfil de acesso para um usuário e a partir desse vinculo os usuários podem efetuar as ações que o perfil permite.

4.1.6 Vincular *Actions* aos Perfil de Acesso

Os gestores podem atribuir a um perfil de acesso o conjunto de ações que a ferramenta disponibiliza.

4.1.7 Geração Automática de Identificadores Únicos

Todos os Requisitos, Projetos, Artefatos, Elos entre Requisitos e Elos entre Requisitos e Artefatos gerados pela ferramenta devem possuir um código de identificação único. Esse identificador único deve ser gerado automaticamente pelo sistema e não deve ser passível de edições pelo usuário. Para os Requisitos o identificador deve começar com a abreviação "REQ", para os Artefatos o identificador deve começar com a abreviação "ART", para os projetos deve começar com "PRJ", para os Elos entre Requisitos com a abreviação "R-R" e para os Elos entre Requisitos e Artefatos com a abreviação "R-A". Todas os códigos devem vir seguidos de uma numeração única.

4.1.8 Listagem das *Controller* e *Actions*

É possível realizar a listagem de todas as *Controller* e suas respectivas *Actions* criadas para a Ferramenta;

4.1.9 Atualizar *Controller* e *Actions*

Caso o sistema seja evoluído e *Controller* e *Actions* sejam adicionadas ou removidas do projeto, essas podem ser atualizadas na base de dados. Não será possível excluir *Actions* que já estejam vinculadas a um perfil de acesso.

4.1.10 Renderizar Arquivos no formato PDF

Este requisito deve ser atendido para a geração de arquivos no formato PDF. O documento no formato PDF deve ser gerado e automaticamente baixado para a máquina do usuário em uma nova janela do navegador.

4.1.11 Listagem de Projetos

Todos os projetos cadastrados na ferramenta são listados em uma tabela dinâmica e interativa de forma que os usuários possam selecionar o projeto e ver os detalhes do mesmo. Além disso, essa tabela deve fornecer um filtro que consulte as informações dos projetos na base de dados.

4.1.12 Detalhes do Projeto

Os usuário do sistema conseguem visualizar todos os detalhes do projeto de software selecionado. A tela que representa esse requisito deve apresentar atalhos rápidos para outros requisitos como Adicionar Requisito, Adicionar *Stakeholder* ao Projeto, Editar o Projeto e Gerar PDF de todos os Requisitos do Projeto. Além disso, essa tela deve apre-

sentar tabelas dinâmica e interativa separadas por abas que apresentem quais os *stakeholders* estão vinculados ao projeto, quais os requisitos do projeto, quais os artefatos gerados para o projeto, a matriz de rastreabilidade para requisitos e a matriz de rastreabilidade para requisitos e artefatos.

4.1.13 Listagem de Requisitos

Todos os requisitos cadastrados na ferramenta são listados em uma tabela dinâmica e interativa de forma que os usuários possam selecionar o requisito e ver os detalhes do mesmo. Além disso, essa tabela deve fornecer um filtro que consulte as informações dos requisitos na base de dados.

4.1.14 Adicionar Requisito

Deve ser possível adicionar um requisito a um projeto. Para o cadastro do novo requisito deve ser possível visualizar os Tipos de Requisitos, Sub-Tipos de Requisitos, Status do Requisito, Importância e *Templates* para Requisitos baseados no Tipo de Requisito escolhido. A data de início e data de término do requisito devem obedecer o padrão **mm/dd/yyyy** e o custo do requisito deve seguir o padrão monetário internacional. Além disso, a descrição do requisito deve ser registrada com o auxilio de editores **WYSIWYG HTML**.

4.1.15 Adicionar *Stakeholder* ao Projeto

É possível adicionar um *stakeholder* ao Projeto de Software. Para esse novo *stakeholder* devem ser informadas o conjunto de atividades que o mesmo desempenhará no projeto e o grau de importância do mesmo que varia de 0 a 9.

4.1.16 Editar o Projeto

Deve ser possível editar as informações do projeto tais como descrição, título, data de início, data de fim, status do projeto, *path* e ao editar o projeto um histórico de mudanças deve ser salvo.

4.1.17 Gerar PDF de todos os Requisitos do Projeto

O usuário pode gerar um PDF de todos os Requisitos vinculados ao Projeto de Software que ele selecionar. Além disso, o usuário pode solicitar essa impressão de acordo com um Tipo de Requisito especifico.

4.1.18 Detalhes do Requisito

Os usuários do sistema conseguem visualizar todos os detalhes do requisito selecionado. A tela que representa esse requisito deve fornecer atalhos para os seguintes requisitos: Adicionar *Stakeholder* ao Requisito, Adicionar uma Característica ao Requisito, Criar Elo com outro Requisito, Criar Elo com Artefato, Apreciar o Requisito, Requisição de Mudança, Rastrear Requisito e Gerar PDF do Requisito. Além disso, essa tela deve exibir tabelas dinâmicas e interativas em diferentes abas que exibam os elos com outros requisitos, os elos com artefatos, as características implementadas pelo requisito, os *Stakeholder* interessados no requisito, a apreciação do requisito pelos *stakeholders*, as requisições de mudanças e o versionamento do requisito.

4.1.19 Adicionar *Stakeholder* ao Requisito

É possível adicionar um *Stakeholder* ao Requisito. Esse *stakeholder* deve estar vinculado ao mesmo projeto ao qual o requisito pertence além disso, deve ser especificado o grau de importância desse *Stakeholder* para o Requisito que pode variar entre 1 e 9.

4.1.20 Adicionar uma Característica ao Requisito

Uma característica de requisitos de software previstos pela Engenharia de Software pode ser vinculada ao requisito. O conjunto de características atribuídas aos requisitos formam um *checklist* gerencial de forma que gestores possam marcar os itens como atendidos ou não atendidos.

4.1.21 Criar Elo com outro Requisito

A ferramenta deve possibilitar a criação de um elo entre dois requisitos do sistema. Para isso devem ser selecionados o requisito de origem, requisito de destino e o tipo de elo.

4.1.22 Criar Elo com Artefato

A ferramenta deve possibilitar a criação de um elo entre um requisito e um artefato do sistema. Para isso devem ser selecionados o requisito, o artefato e o tipo de elo.

4.1.23 Apreciar o Requisito

Os *StakeholdersStakeholders* vinculados ao Requisito podem dar um parecer sobre o requisito informando se o mesmo está de acordo com as especificações e exigências;

4.1.24 Requisição de Mudança

Os *Stakeholders* vinculados ao Requisito podem solicitar a mudança do mesmo. Essa mudança pode ser avaliada pelo gestor do projeto e mudar de situação para “Em Analise”, “Rejeitada” ou “Aprovada”.

4.1.25 Avaliação de Solicitação de Mudança

Os gestores do projeto de software devem poder avaliar uma Requisição de Mudança e colocar elas sobre Análise, Rejeitar essa Requisição ou Aprovar a Requisição. Ao aprovar a requisição, o gestor deve poder editar o requisito e o mesmo deve ser versionado.

4.1.26 Editar Requisito

Os gestores do projeto podem editar os dados do requisito mediante uma Requisição de Mudança. Os dados a serem modificados devem estar em conformidade com a requisição de mudança feita pelo *stakeholder* solicitante. Após ser modificado, esse requisito deve ser versionado e o número da sua versão acrescentado em mais um (+1).

4.1.27 Gerar PDF do Requisito

O usuário pode gerar um arquivo no formato PDF com informações referentes ao requisito.

4.1.28 Rastreio em Código-Fonte

A ferramenta deve disponibilizar o rastreio em código-fonte Requisito, Artefato, Elos entre Requisitos e Elos entre Artefatos. Para isso, deve ser informado o caminho lógico do diretório em que encontra-se o código-fonte. Esse diretório deve estar no mesmo local em que a ferramenta encontra-se hospedada/publicada para que o rastreio possa ser efetuado.

4.1.29 Template para Requisitos

Para aumentar a velocidade e disseminar a padronização do cadastro de requisitos, os usuários devem poder cadastrar *templates* para requisitos no formato HTML com uso de editores WYSIWYG de forma que durante o cadastro dos

requisitos os usuários possam escolher um *template* e apenas completar o mesmo com as informações que restam.

4.1.30 Cadastrar Stakeholder

Os usuários cadastrados no sistema por um administrador podem tornar-se *stakeholders* e assim poderem participar de avaliação de projetos e requisitos. Para isso, deve ser selecionado o usuário e a classificação do mesmo como um *stakeholders*. Cada usuário pode ser cadastrado diversas vezes como um *stakeholder* contudo não deve possuir a mesma classificação.

4.1.31 Classificação de Stakeholders

A ferramenta deve disponibilizar o cadastro de classificações para os usuários que desejam tornar-se *stakeholders*. As classificações indicam as ocupações, profissões e conhecimentos que o *stakeholder* pode agregar ao projeto.

4.1.32 Cadastrar Tipos de Requisitos

Os requisitos podem ter diversos tipos tais como funcionais, não funcionais e histórias de usuário. O sistema deve fornecer esses três tipos já pré-cadastrados mas deve fornecer a opção de inserir novos tipos no sistema.

4.1.33 Cadastrar Subtipos para Requisitos

Para cada tipo de requisito uma família de Subtipos deve poder ser cadastrada. Os requisitos não funcionais possuem diversos de SubTipo tais como Desempenho, Segurança, Disponibilidade, Usabilidade, Confiabilidade e muitos outros.

4.1.34 Cadastrar Características

Os usuários do sistema podem cadastrar Características que os requisitos do *software* poderão implementar. Essas características podem ser ou não classificadas como obrigatórias para os requisitos. Quando um requisito for criado ele deve conter todas as características classificadas como obrigatórias.

4.1.35 Matriz de Rastreabilidade entre Requisitos

O sistema deve fornecer uma matriz de rastreabilidade entre requisitos de forma dinâmica e interativa. Na primeira coluna devem estar dispostos os requisitos de origem do rastreio e nas demais colunas os requisitos de destino. Ao clicar no código de um requisito deve ser exibida uma pequena janela contendo os detalhes do requisito. Ao clicar na no ponto de relação entre os dois requisitos (linha versus coluna), deve ser exibida uma pequena janela com os dados do elo entre os requisitos e os atributos do elo.

4.1.36 Matriz de Rastreabilidade entre Requisitos e Artefatos

O sistema deve fornecer uma matriz de rastreabilidade entre requisitos e os artefatos de forma dinâmica e interativa. Na primeira coluna devem estar dispostos os requisitos do rastreio e nas demais colunas os artefatos. Ao clicar no código de um requisito deve ser exibida uma pequena janela contendo os detalhes do requisito. Ao clicar no código de um artefato deve ser exibida uma pequena janela contendo os detalhes do artefato. Ao clicar no ponto de relação entre o requisito e o artefato (linha *versus* coluna), deve ser exibida uma pequena janela com os dados do elo entre os requisito e os artefatos bem como os atributos do elo.

4.1.37 DataSets

O sistema deve disponibilizar um conjunto de *DataSets* para a área de *Search-Based Software Engineering* nos formatos **JSON** e **CSV**. Tais *DataSets* devem fornecer os dados com os seguintes padrões:

1. **Custo dos Requisitos**: Listagem de todos os requisitos com seus respectivos custos;

2. *Dependências entre Requisitos*: Listagem de todos os requisitos que possuem dependência com outros requisitos;

3. *Precedência entre Requisitos*: Listagem de todos os requisitos que possuem precedência com outros requisitos;

4. *Média da Importância dos Stakeholder por Requisitos*: Listagem de todos os requisitos com a média aritmética do grau de importância que cada *stakeholder* atribuiu ao requisito.

5. ***Stakeholders* por Projeto e por Requisito**: Listagem de todos os *stakeholders* por projeto e requisito com o respectivo grau de importância atributo para o projeto e requisito.

4.2 Requisitos Não-Funcionais

4.2.1 Autenticação

Somente usuários previamente cadastrados no sistema por um administrador poderão acessar o sistema mediante o fornecimento de um *login* e senha. Toda requisição feita ao sistema deve verificar se o usuário está logado no sistema.

4.2.2 Implementação

O sistema deve ser desenvolvido com a linguagem CSharp com o uso do *framework* ASP.NET MVC 5. O código-fonte gerado a partir dos requisitos elicitados na seção 4 deve ser aberto, livre e disponibilizado em uma plataforma acessível.

5. DESENVOLVIMENTO DO SISTEMA

5.1 Arquitetura

A arquitetura desse projeto foi baseada no estilo arquitetural em camadas e em padrões de projetos difundidos entre os desenvolvedores da plataforma ASP.NET MVC. No apêndice C encontra-se a figura 18 que representa o diagrama arquitetural dessa proposto para a ferramenta.

1. **MVC**: As *Controllers* recebem requisições (*Request*) e envia respostas (*Response*) atráves de visualizações de dados (*Views*). Essa camada solicita dados da camada de *Service* camada esta responsável pelo tratamento das regras de negócio do sistema;

2. ***Service***: Representa as lógicas de negócio do *software*. Essa camada solicita os dados oriundos da camada *Repository* e manipula os dados fornecidos pela mesma bem como envia dados para que a camada *Repository* realize operações de persistência, atualização e remoção;

3. ***Repository***: Essa camada é responsável por fornecer uma abstração de como os dados são acessados na base de dados implementando ações de consulta, persistência, atualização e remoção de dados de forma que a camada superior não tenha conhecimento de como esse processo é realizado;

5.2 Tecnologias Utilizadas

Essa sub-seção apresentada as principais tecnologias utilizadas para o desenvolvimento do sistema.

1. **GIT**: A tecnologia é utilizada em Sistemas de Controle de Versões Distribuídos. De código livre, o GIT possibilita que diversos usuários possam trabalhar sobre um conjunto de arquivos de forma independente, criar fluxos alternativos de trabalho sem impactar a linha principal de produção e outras vantagens. Para auxiliar o versionamento do sistema desenvolvido foi escolhida a plataforma GitHub [13] que utiliza a tecnologia GIT;

2. **EntityFramework 6.1.7**[14]: É um framework utilizado para mapeamento objeto relacional (ORM – *Object Relational Management*) que proporciona uma abstração do banco de dados relacional e dos comandos de leitura e escrita (*SELECT*, *INSERT*, *UPDATE* e *DELETE*) permitindo que os desenvolvedores acessem os dados através de classes que mapeiam tabelas da base de dados;

3. **jQuery 3.2.1**[15]: Biblioteca de funções *JavaScript* que manipula diretamente os elementos HTML que foi desenvolvida com o objetivo de simplificar os *scripts* interpretados pelos *browsers*;

4. **Bootstrap 3.3.7**: *Framework* HTML, CSS e *JavaScript* para desenvolvimento de projetos *web* responsíveis;

5. **ASP.NET MVC 5.2.3**: *Framework* criado para a implementação da arquitetura MVC para a plataforma ASP.NET;

6. **Autofac.Mvc5 4.0.1**: *Framework* utilizado para a inversão de controle e injeção de dependência para a plataforma ASP.NET MVC;

7. **AutoMapper 5.0.2**: Biblioteca utilizada para mapeamento de atributos de uma classe para os atributos de outra classe;

8. **iTextSharp 5.5.12**: *Framework* para a geração de arquivos no formato PDF com bases em textos normais ou no formato HTML;

9. **Sistema de Gerenciamento de Banco de Dados (SGDB)**: *SQLServer* 2008;

[13]https://github.com/
[14]https://msdn.microsoft.com/pt-br/library/bb399567(v=vs.110).aspx
[15]https://jquery.com/

6. MODELAGEM

Nesta seção serão apresentados os dois grupos de diagramas confeccionados para a modelagem do sistema: casos de uso e de entidade e relacionamento. O primeiro grupo de diagramas foi produzido com base na linguagem UML com o uso da ferramenta **ASTAH Community**[16] e o segundo grupo de diagramas foi produzido com a ferramenta **SQL-POWER**[17].

Todos os diagramas dessa seção estão dispostos no *site* oficial do grupo de pesquisa **LabraSoft**[18].

7. PRINCIPAIS FUNCIONALIDADES

O conjunto das principais funcionalidades desenvolvidas para a ferramenta desenvolvidas com base nos requisitos elicitados e descritos na seção 4, estão disponibilizados em um vídeo demonstrativo hospedados na plataforma de vídeos **YouTube**[19].

8. PROJETO EM ESTUDO

A ferramenta proposta por essa monografia foi implementada na empresa Computação Brasil[20] que atua no desenvolvimento de *softwares* corporativos desde 2008. Essa empresa desenvolve soluções e aplicações voltadas ao gerenciamento de projetos, materiais e exames hospitalares, transações financeiras e de administração contábil, financeira e recursos humanos.

Durante os dias 01/01/2018 à 16/03/2018, a ferramenta foi utilizada por três analistas de sistemas para a implementação do gerenciamento dos requisitos em projetos reais e consultada por três desenvolvedores e quatro estagiários como forma de apoio ao desenvolvimento dos requisitos. Durante os dias 01/02/2018 à 23/03/2018 a mesma foi avaliada por dois analistas de sistemas com um maior grau de experiência e por dois professores e dois estudantes do Instituto Federal de Ciências e Tecnologia da Bahia. Cinco dias antes do término desses períodos foram aplicados cinco questionários dispostos no apêndice A.

Os questionários foram confeccionados com base nas metodologias de avaliação **GQM**. A avaliação GQM (*Goal*, *Questions*, *Metrics*) é bastante utilizada na Engenharia de *Software* na qual é definida uma Meta (*Goal*) e um conjunto de Questões (*Questions*) são estabelecidas para compreender a Meta, além disso, um conjunto de Métricas (*Metrics*) são estabelecidas para avaliar essas questões. Para esse trabalho, a avaliação das questões foi realizada com a aplicação da Escala Likert [16] que proporciona uma avaliação quantitativa para as questões. Todas as questões foram respondidas seguindo o formato tradicional da Escala Likert:

1. Discordo totalmente;
2. Discordo parcialmente;
3. Indiferente;
4. Concordo parcialmente;
5. Concordo totalmente;

[16]http://astah.net/editions/community
[17]http://software.sqlpower.ca/page/splash
[18]http://www.labrasoft.ifba.edu.br/t-aadsp
[19]https://www.youtube.com/channel/UC9bKqjDgmQegKBipbrjEiMg
[20]http://computacaobrasil.com.br/

9. RESULTADOS E DISCUSSÃO

O gerenciamento de requisito foi aplicado em quatro projetos de *software* sendo que os mesmos encontravam-se em diferentes fases. Desses os *softwares* de gerenciamento de transações financeiras com cartões e o de gestão de coordenadores estavam em codificação. Já o de gerenciamento de solicitações de compras encontrava-se em evolução e o de controle de pregão e licitações estava concluído. Desse modo, foram aplicados cinco questionários objetivando conhecer a aplicabilidade do gerenciamento de requisitos com o uso de uma ferramenta, em quais pontos a ferramenta atendeu as expectativas dos usuários e os auxiliou no projeto e desenvolvimento de *softwares* com qualidade.

O primeiro questionário foi aplicado com estagiários e recém-formados da área computacional e com base nas respostas coletadas observou-se que a metade dos entrevistados não teve contato com requisitos anteriormente e a outra metade já possuía contato com requisitos. Além disso, a ferramenta ajuda no esclarecimento das funcionalidades do *software* a ser desenvolvido e na familiarização com os requisitos do projeto e que a falta de documentação, ambiguidade e a não clareza dos mesmos pode afetar de forma negativa na participação dos mesmos no projeto. Requisitos bem documentados e constantemente atualizados ajudam no processo de desenvolvimento do *software* [1] e proporcionam qualidade ao mesmo pois todos os interessados no sucesso do projeto tem conhecimento adequado do que deve ser desenvolvido e em como atender as necessidades dos *stakeholders* [4, 7, 2].

O segundo questionário foi aplicado com analistas de sistemas para compreender se a ferramenta atende aos critérios da gerência de requisitos propostos pelo guia **MPS.BR** [4]. Com base nas respostas dos entrevistados, observou-se que entendimento dos requisitos junto com os *stakeholders* do projeto, revisões em planos e produtos de trabalho e as mudanças nos requisitos são gerenciadas ao longo do projeto são parcialmente atendidos pela ferramenta. A ferramenta disponibiliza mecanismos para aceitação e avaliação dos requisitos pelos *stakeholders* interessados no mesmo e versionamentos dos requisitos, contudo, a revisão em planos e produtos e entendimento dos requisitos junto com *stakeholders* - critério esse imposto pela gerência de requisitos do guia **MPS.BR** [4] - é um processo específico de cada empresa e que a ferramenta não mantém controle pois cada empresa tem um conjunto regras específicas para revisão de projetos e artefatos, além disso, os *stakeholders* dos projetos podem ou não estar disponíveis para consulta e validação de requisitos [7, 10, 1, 3]. Em contrapartida, o comprometimento da equipe técnica e a rastreabilidade bidirecional entre os requisitos e os artefatos solicitados pelo guia **MPS.BR** [4] são amplamente atendidos pela ferramenta.

O terceiro questionário foi aplicado com analistas de sistemas objetivando conhecer como uma ferramenta de gerenciamento de requisitos pode trazer benefícios ao desenvolvimento de *software* em um contexto onde requisitos mudam constantemente e se essas mudanças que podem afetar a entrega, custo, prazo e qualidade do projeto [3, 7, 2, 1, 6] podem ser amenizadas com o uso da ferramenta. Fora constatado que o uso da ferramenta para gerenciamento de requisitos proporciona agilidade no cadastro e riqueza nos detalhes dos requisitos além do esclarecimento das solicitações de mudanças. Outro ponto interessante observado nesse questionário e que também fora constatado no primeiro questionário está na utilização da ferramenta como forma de aprendizado para pessoas com pouca experiencia ou novos membros na equipe [7].

Outro resultado coletado no terceiro questionário diz respeito a participação dos membros da equipe de desenvolvimento no processo de gerenciamento de requisitos [7, 2]. Para os entrevistados toda equipe de desenvolvimento deve participar do processo de gerenciamento de requisitos, contudo, essa responsabilidade deve ser atribuída a um membro ou grupo especifico [1, 6]. O cenário ideal seria o(s) Engenheiro(s) de Requisitos ter(em) o controle dos requisitos com base na gerência de requisitos e a equipe de desenvolvimento participar com o comprometimento sobre o(s) requisito(s), informar inconsistências e conflitos e zelar pelo desenvolvimento conforme as especificações do requisito.

O quarto questionário foi aplicado visando a compreensão sobre a prática da rastreabilidade de requisitos. Os entrevistados concordam que o registro de elos ajudam a compreender melhor os requisitos [6] e que a ferramenta possibilita o registro da rastreabilidade durante todo ciclo de vida do *software* [6, 7, 2]. Entretanto, uma divergência foi constatada nas respostas relacionadas a escolha e manutenção da rastreabilidade para os requisitos. Metade dos dados coletados expressam que a rastreabilidade deve ser aplicada para todos os requisitos, contudo, a outra metade expressa que a grande quantidade de informações gerada pela rastreabilidade pode tornar-se custosa a longo prazo [7, 2].

O último questionário foi aplicado com um grupo de pesquisadores do Instituto Federal de Ciências e Tecnologia da Bahia (IFBA) objetivando o aceite dos *DataSets* gerados pela ferramenta com base nos parâmetros de entrada de dados estabelecidas pelo problema da área de **SBGE**, o **NRP** [9], abordado brevemente na seção 2.8, tais como no grau de importância dos *stakeholders* para os projetos, dos requisitos para os *stakeholders*, custo dos requisitos e precedência e dependência entre requisitos. Com base nos dados coletados, todos os *DataSets* gerados pela ferramenta atendem as expectativas.

10. CONCLUSÃO E TRABALHOS FUTUROS

Este trabalho apresentou a ferramenta T-AADSP *Requirements*, ferramenta esta voltada para o gerenciamento de requisitos de *software* e que segue os principais fundamentos para o gerenciamento de requisitos estabelecidos pela Engenharia de Requisitos, por guias de melhoria de processo e qualidade para *softwares* como o **MPS.BR** [4] e guia **AADSP** [14] e em alguns metamodelos de rastreabilidade de requisitos propostos pela literatura.

Ramesh [7] em sua longa pesquisa de campo, afirma que a atividade gerenciamento de requisitos e a rastreabilidade dos mesmos é uma atividade custosa e duradoura e que seus benefícios são visualizados apenas a longo prazo. Com base nessa premissa, essa foi ferramenta foi concebida visando proporcionar aos usuários finais uma melhoria em seus *softwares* através da otimização de recursos como tempo, qualidade e custo.

Através de todo levantamento bibliográfico, elicitação dos requisitos para a própria ferramenta, acompanhamento e entrevistas com os usuários através de questionários, pode-se perceber que o de gerenciamento de requisitos é de crucial importância para o sucesso dos projetos de *software* pois

tal gerência consegue difundir uma visão uniforme e linear a todos os interessados nos requisitos do projeto, desde os desenvolvedores aos clientes. Contudo, a aplicação do gerenciamento de requisito deve variar de empresa para empresa e de projeto para projeto de acordo com o contexto socioeconômico-cultural na qual será aplicada.

Como trabalhos futuros para a ferramenta, podem ser aplicados algoritmos de seleção genética de requisitos com base nos seus custos e o grau de importância dos *stakeholders* para cada requisito e projeto. Tais algoritmos conseguem avaliar e indicar qual requisito do projeto precisa ser implementado pela equipe de desenvolvimento de forma que atenda as necessidades dos clientes. Além disso, o gerenciamento de tarefas pode ser implementado para vincular um conjunto de atividades feitas pela equipe de desenvolvimento a um requisito. Outras melhorias que podem ser aplicadas são: otimização do algoritmo de rastreio em código-fonte e a criptografia de dados. ç

11. REFERÊNCIAS

[1] P. S. N. D. Lopes, "Uma taxonomia da pesquisa na Área de engenharia de requisitos," *Dissertação de Mestrado, IME/USP*, pp. 14–44, 2002.

[2] M. Sayão and J. C. S. P. Leite, "Rastreabilidade de requisitos," *Revista de Informática Teórica e Aplicada - RITA*, vol. XII, no. 1, pp. 1–22, 2005.

[3] I. Sommerville and G. Kotonya, *Requeriments Engineering: Process and Techniques*, ch. 5. Requeriments Engineering: Process and Technique, 1998.

[4] SOFTEX, "Mps.br: melhoria de processo do software brasileiro," SOFTEX, 2012.

[5] P. R. C. Malcher, D. A. L. Ferreira, S. R. B. Oliveira, and A. M. L. de Vasconcelos, "Um mapeamento sistemático sobre abordagens de apoio à rastreabilidade de requisitos no contexto de projetos de software," *Revista de Sistema de Informação da FSMA*, vol. 1, no. 16, pp. 3–15, 2015.

[6] E. C. Genvigir, "Um modelo para rastreabilidade de Requisitos de Software baseado em Generalização de Elos e Atributos.," *Instituto Nacional de Pesquisas Espaciais - INPE*, pp. 27–70, 2009.

[7] B. Ramesh, "Factory influencing requirements traceability practice," *ACM*, vol. 41, no. 12, pp. 37–44, 1998.

[8] M. M. A. Brasil, F. G. de Freitas, T. G. N. da Silva, J. T. de Souza, and M. I. Cortés, "Uma nova abordagem de otimização multiobjetiva para o planejamento de releases em desenvolvimento iterativo e incremental de software," *I Workshop Brasileiro de Otimização em Engenharia de Software*, vol. 1, pp. 40–47, 2012.

[9] Y. Zhang, A. Finkelstein, and M. Harmn, "The multi-objective next release problem," *ACM. Proceedings of the 9th annual conference on Genetic and evolutionary computation*, vol. 1, pp. 1129–1137, 2007.

[10] M. Toranzo and J. Castro, "Uma proposta para melhorar o rastreamento de requisitos," *WER02 - Workshop em Engenharia de Requisitos, Valencia, Espanha*, pp. 194–209, 2002.

[11] IEEE, "Ieee std 830-1998 - ieee recommended practice for software requirements specififfications." http://www.math.uaa.alaska.edu/~afkjm/cs401/IEEE830.pdf, 1998 (acessado Janeiro 21, 2018).

[12] R. S. de Espindola, A. Majdenbaum, and J. L. N. Audy, "Uma análise crítica dos desafios para engenharia de requisitos em manutenção de software," *Anais do WER04 - Workshop em Engenharia de Requisitos*, pp. 226–238, 2004.

[13] S. de Rezende Alves and A. L. Alves, "Engenharia de requisitos em metodologias Ágeis," Pontifícia Universidade Católica do Rio Grande Sul, 2009.

[14] AADSP, "Adaptive approach for deployment of software process." http://www.labrasoft.ifba.edu.br/wp-content/uploads/2016/10/Guia-AADSP.pdf, 2016 (acessado Dezembro 14, 2017).

[15] F. F. Mendes, P. G. Fernandes, J. L. de Oliveira, C. da Cunha Mota, M. D. S. Martins, and R. da Silva Nunes, "Análise de ferramentas para apoio à gerência de projetos e gerência de requisitos de software," *WAMPS2010 - VI Workshop Anual do MPS*, pp. 148–157, 2010.

[16] T. C. Kinnear and J. R. Taylor, "Marketing research: an applied approach," McGraw Hill, 1991.

Apêndices

A. QUESTIONÁRIOS

A.1 Analisar o gerenciamento de requisitos com o uso da ferramenta T-AADSP Requirements com o propósito de verificar a importância do acompanhamento dos requisitos durante o ciclo de vida de um software no ponto de vista de desenvolvedores com pouca experiência (estagiários ou recém-contratados) no contexto onde o entendimento dos requisitos e a relação destes com os artefatos ajuda o profissional a se familiarizar com o objetivo do projeto e acelerar a participação deste no projeto.

A.1.1 O usuário teve contato com requisitos anteriormente através de documentos de requisitos.

A.1.2 Consultar os requisitos pode ajudar a esclarecer o conjunto de funcionalidades do software e no auxiliar no entendimento das mudanças solicitadas pelos stakeholders.

A.1.3 A ferramenta facilita o entendimento dos requisitos e a relação destes com os artefatos.

A.1.4 A falta de informação ou documentação dos requisitos do sistema pode impactar no atraso para o entendimento do projeto ou na sua contribuição para o projeto.

A.1.5 A ferramenta ajuda o profissional a se familiarizar com o objetivo do projeto e acelerar a participação deste no projeto.

A.2 Analisar a aplicabilidade da ferramenta T-AADSP-Requirements com o propósito de auxiliar no atendimento aos critérios estabelecidos no MPS-BR nível G - gerenciamento de requisitos - sob o ponto de vista da equipe de desenvolvimento no contexto onde as mudanças nos requisitos podem afetar o prazo de entrega, escopo, custo e qualidade do projeto.

A.2.1 O entendimento dos requisitos é obtido junto aos fornecedores de requisitos.

A.2.2 Os requisitos são avaliados com base em critérios objetivos e um comprometimento da equipe técnica com estes requisitos é obtido.

A.2.3 A rastreabilidade bidirecional entre os requisitos e os produtos de trabalho é estabelecida e mantida.

A.2.4 Revisões em planos e produtos de trabalho do projeto são realizadas visando a identificar e corrigir inconsistências em relação aos requisitos.

A.2.5 Mudanças nos requisitos são gerenciadas ao longo do projeto.

A.3 Analisar a aplicabilidade da ferramenta T-AADSP Requirements com o propósito avaliar o software com relação à melhoria no acompanhamento de requisitos sob o ponto de vista de analistas de sistemas ou gerentes de projeto no contexto onde as mudanças em requisitos podem afetar o prazo de entrega, escopo, custo e qualidade do projeto.

A.3.1 Possui experiência com registro e atualização de requisitos em editores de textos tradicionais.

A.3.2 O registro e atualização de requisito em ferramentas voltadas ao gerenciamento de requisitos proporciona uma maior agilidade no registro dos mesmos.

A.3.3 Consultar os requisitos para auxiliar nas mudanças solicitadas por stakeholders pode ajudar a esclarecer o conjunto de funcionalidades do software e evoluir o requisito garantindo a consistência do mesmo.

A.3.4 A ferramenta pode ser utilizada como um meio de aprendizado para pessoas com poucas experiência (estagiários e recém-formados) ou novos membros da equipe que não conhecem os requisitos da empresa.

A.3.5 Requisitos bem documentados e constantemente atualizados proporcionam qualidade ao software.

A.3.6 Toda equipe de desenvolvimento do software deve participar do processo de gerenciamento de requisitos.

A.3.7 Deve-se atribuir a responsabilidade gerenciamento de requisitos para um grupo ou membro específico da equipe de desenvolvimento.

A.3.8 O gerenciamento de requisitos possibilita a equipe e a gerência de desenvolvimento avaliar a viabilidade do projeto dado as constantes mudanças.

A.4 Analisar a rastreabilidade de requisitos proporcionada pela ferramenta T-AADSP Requeriments com o propósito rastrear requisitos com relação vantagens e desvantagens do uso da rastreabilidade sob o ponto de vista de gerentes de projeto, analistas e desenvolvedores de sistemas no contexto onde os requisitos constantemente modificados.

A.4.1 A prática de rastreabilidade deve ser aplicada para todos os requisitos.

A.4.2 A grande quantidade de informações gerada pela rastreabilidade de requisitos pode tornar-se custosa a longo prazo.

A.4.3 Toda equipe de desenvolvimento do software deve participar do processo de rastreabilidade de requisitos.

A.4.4 Deve-se atribuir a responsabilidade rastreabilidade para um grupo específico ou membro da equipe.

A.4.5 Os elos entre requisitos ajudam a compreender melhor os requisitos do projeto e as precedências entre eles.

A.4.6 A ferramenta apresentada possibilita o registro da rastreabilidade dos requisitos durante todo o ciclo de vida do projeto.

A.4.7 O gerenciamento e a rastreabilidade de requisitos na ferramenta apresentada (elos, versionamentos, artefatos de origem, stakeholders, rationale) ajudam a entender melhor o contexto em que os requisitos foram criados.

A.5 Analisar a geração de arquivos com informações dos requisitos, custos stakeholders e a importância destes para o requisito e o projeto pela ferramenta T-AADSP Requirements com o propósito de criar um DataSets para entrada nos algoritmos genéticos baseados em busca para engenharia de requisitos com relação a produção de dados para utilização em pesquisa sob o ponto de vista de pesquisadores de otimização em engenharia de software no contexto onde a escolha dos requisitos é crucial para continuidade e implantação do projeto de software.

A.5.1 Os datasets gerado pela ferramenta provê informações sobre dependência entre requisitos.

A.5.2 Os datasets gerado pela ferramenta provê informações sobre prioridades dos requisitos.

A.5.3 Os datasets gerado pela ferramenta provê informações sobre o grau de importância do stakeholder para o projeto.

A.5.4 Os datasets gerado pela ferramenta provê informações sobre a importância do requisito para o stakeholder

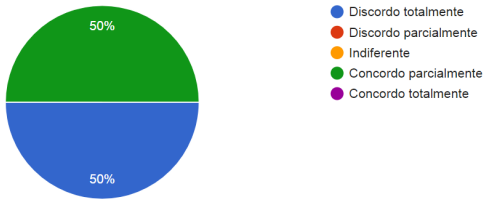
A.5.5 Os datasets gerado pela ferramenta provê informações sobre o custo para o desenvolvimento de cada requisito do projeto.

A.5.6 A ferramenta gera datasets compatível e útil para algoritmos genéticos de busca para seleção de requisitos.

A.5.7 O gerenciamento e a rastreabilidade de requisitos na ferramenta apresentada (elos, versionamentos, artefatos de origem, stakeholders, rationale) ajudam a entender melhor o contexto em que os requisitos foram criados.

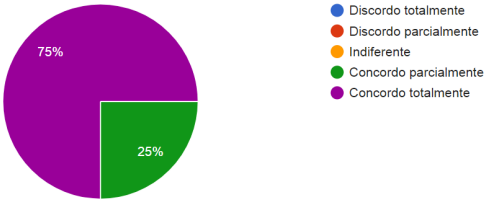
B. RESULTADOS

Contato com requisitos anteriormente através de documentos de requisitos.



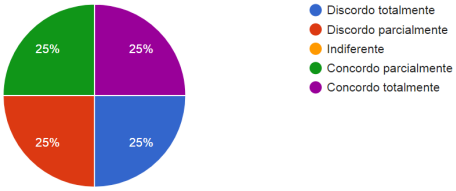
Questão A 1.1

Consultar os requisitos pode ajudar a esclarecer o conjunto de funcionalidades do software e no auxiliar no entendimento das mudanças solicitadas pelos stakeholders



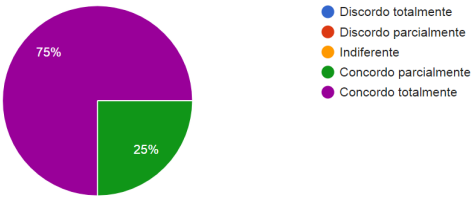
Questão A 1.2

A ferramenta facilita o entendimento dos requisitos e a relação destes com os artefatos.



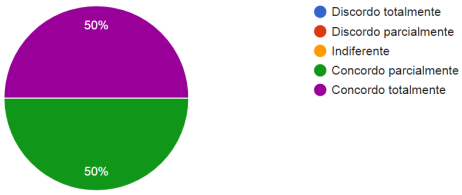
Questão A 1.3

A falta de informação ou documentação dos requisitos do sistema pode impactar no atraso para o entendimento do projeto ou na sua contribuição para o projeto.



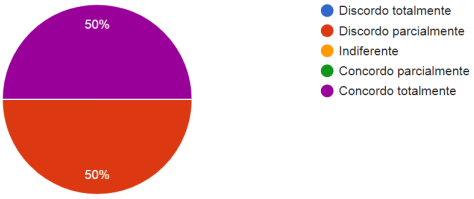
Questão A 1.4

A ferramenta ajuda o profissional a se familiarizar com o objetivo do projeto e acelerar a participação deste no projeto.



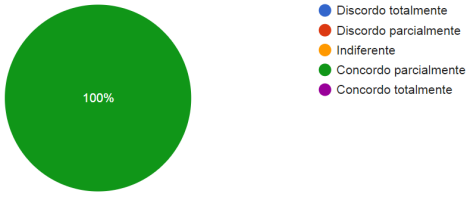
Questão A 1.5

O entendimento dos requisitos é obtido junto aos fornecedores de requisitos.



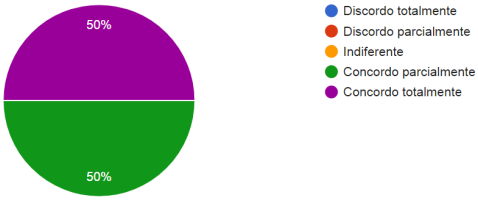
Questão B 2.1

Os requisitos são avaliados com base em critérios objetivos e um comprometimento da equipe técnica com estes requisitos é obtido.



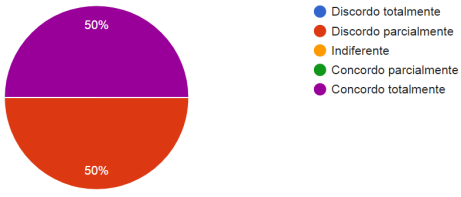
Questão B 2.2

A rastreabilidade bidirecional entre os requisitos e os produtos de trabalho é estabelecida e mantida.



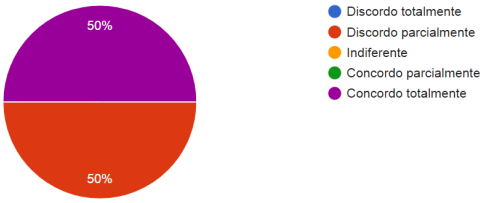
Questão B 2.3

Revisões em planos e produtos de trabalho do projeto são realizadas visando a identificar e corrigir inconsistências em relação aos requisitos.



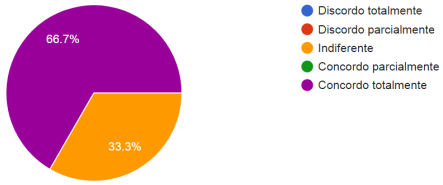
Questão B 2.4

Mudanças nos requisitos são gerenciadas ao longo do projeto.



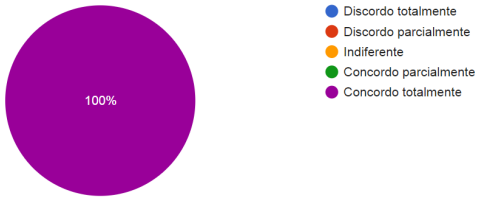
Questão B 2.5

Possui experiência com registro e atualização de requisitos em editores de textos tradicionais.



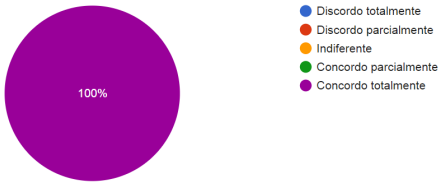
Questão C 3.1

O registro e atualização de requisito em ferramentas voltadas ao gerenciamento de requisitos proporciona uma maior agilidade no registro dos mesmos.



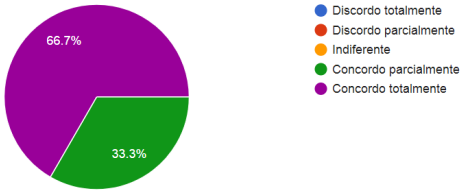
Questão C 3.2

Consultar os requisitos para auxiliar nas mudanças solicitadas por stakeholders pode ajudar a esclarecer o conjunto de funcionalidades do software e evoluir o requisito garantindo a consistência do mesmo.



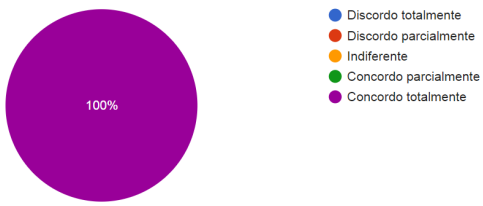
Questão C 3.3

A ferramenta pode ser utilizada como um meio de aprendizado para pessoas com poucas experiência (estagiários e recém-formados) ou novos membros da equipe que não conhecem os requisitos da empresa.



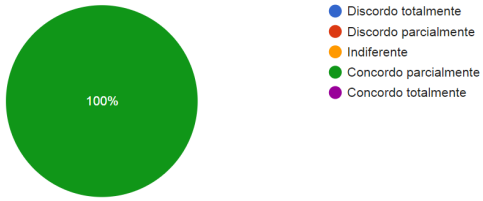
Questão C 3.4

Requisitos bem documentados e constantemente atualizados proporcionam qualidade ao software.



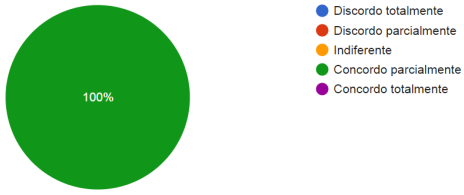
Questão C 3.5

Toda equipe de desenvolvimento do software deve participar do processo de gerenciamento de requisitos.



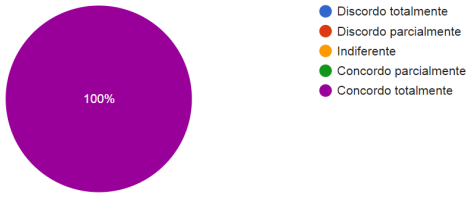
Questão C 3.6

Deve-se atribuir a responsabilidade gerenciamento de requisitos para um grupo ou membro específico da equipe de desenvolvimento.



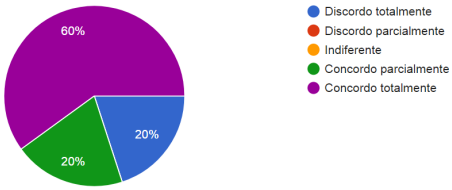
Questão C 3.7

O gerenciamento de requisitos possibilita a equipe e a gerência de desenvolvimento avaliar a viabilidade do projeto dado as constantes mudanças.



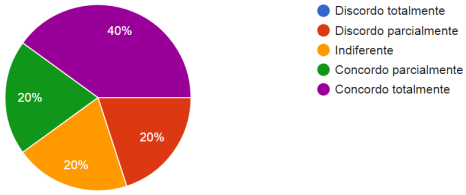
Questão C 3.8

A prática de rastreabilidade deve ser aplicada para todos os requisitos.



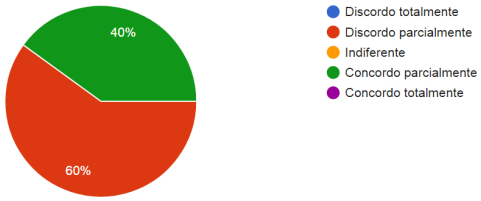
Questão D 4.1

A grande quantidade de informações gerada pela rastreabilidade de requisitos pode tornar-se custosa a longo prazo.



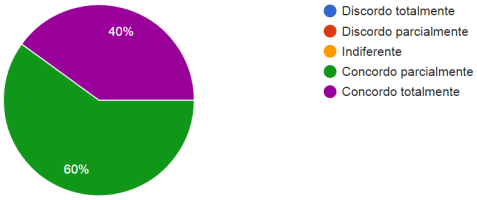
Questão D 4.2

Toda equipe de desenvolvimento do software deve participar do processo de rastreabilidade de requisitos.



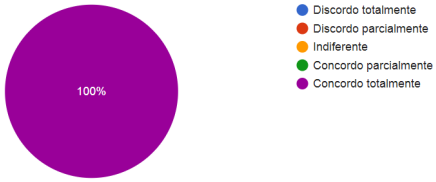
Questão D 4.3

Deve-se atribuir a responsabilidade rastreabilidade para um grupo específico ou membro da equipe.



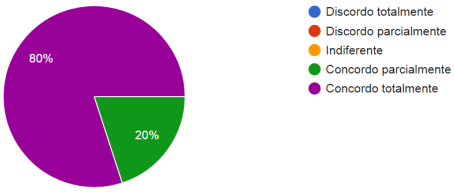
Questão D 4.4

Os elos entre requisitos ajudam a compreender melhor os requisitos do projeto e as precedências entre eles.



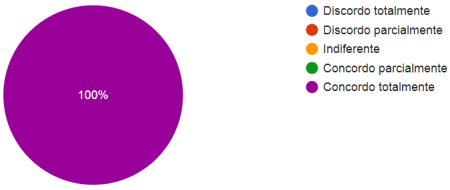
Questão D 4.5

A ferramenta apresentada possibilita o registro da rastreabilidade dos requisitos durante todo o ciclo de vida do projeto.



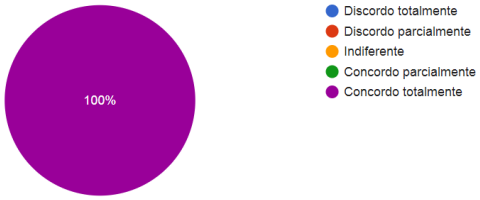
Questão D 4.6

O gerenciamento e a rastreabilidade de requisitos na ferramenta apresentada (elos, versionamentos, artefatos de origem, stakeholders, rationale) ajudam a entender melhor o contexto em que os requisitos foram criados.



Questão D 4.7

O dataset gerado pela ferramenta provê informações sobre dependência entre requisitos.

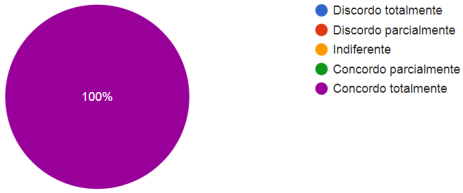


Questão E 5.1

O dataset gerado pela ferramenta provê informações sobre prioridades dos requisitos.

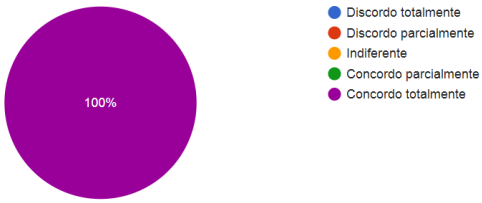
Questão E 5.2

O dataset gerado pela ferramenta provê informações sobre o grau de importância do stakeholder para o projeto.



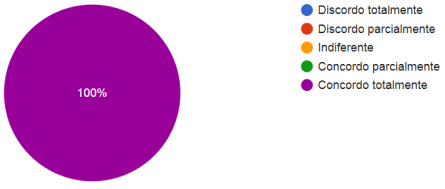
Questão E 5.3

O dataset gerado pela ferramenta provê informações sobre a importância do requisito para o stakeholder



Questão E 5.4

O dataset gerado pela ferramenta provê informações sobre o custo para o desenvolvimento de cada requisito do projeto.



Questão E 5.5

A ferramenta gera DataSet compatível e útil para algoritmos genéticos de busca para seleção de requisitos.

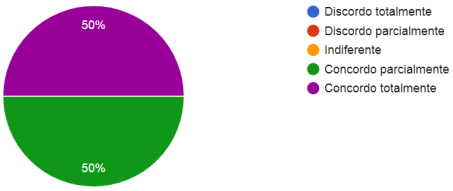


Figura 17: Questão E 5.6

C. MODELAGEM ARQUITETURAL

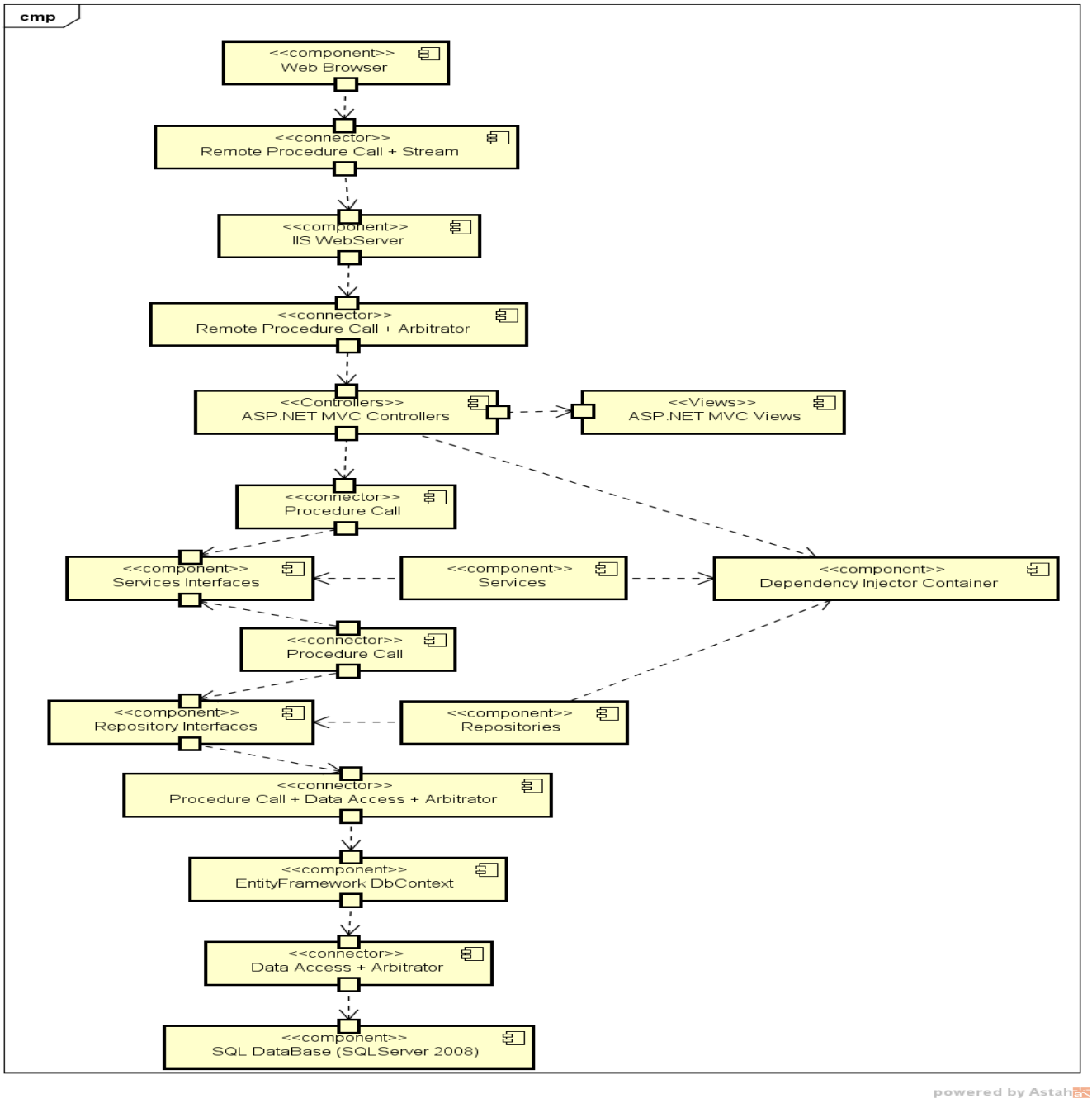


Figura 18: Diagrama Arquitetural Componente-Conector