UNIVERSIDADE ESTADUAL PAULISTA

"Júlio de Mesquita Filho"

Pós-Graduação em Ciência da Computação

Vanessa dos Anjos Borges

Apoio à Gerência de Requisitos de Software utilizando *Design Rationale*

UNESP

2014

Borges, Vanessa dos Anjos.
Apoio à gerência de requisitos de software utilizando Design Rationale / Vanessa dos Anjos Borges. -- São José do Rio Preto, 2014
99 f. : il., tabs.

Orientador: Rogéria Cristiane Gratão de Souza
Dissertação (mestrado) – Universidade Estadual Paulista "Júlio de Mesquita Filho", Instituto de Biociências, Letras e Ciências Exatas

1. Computação. 2. Engenharia de software. 3. Engenharia de requisitos. 4. Software – Documentação. I. Souza, Rogéria Cristiane Gratão de. II. Universidade Estadual Paulista "Júlio de Mesquita Filho". Instituto de Biociências, Letras e Ciências Exatas. III. Título.

CDU – 518.72

Ficha catalográfica elaborada pela Biblioteca do IBILCE
UNESP - Câmpus de São José do Rio Preto

Vanessa dos Anjos Borges

Apoio à Gerência de Requisitos de Software utilizando *Design Rationale*

Orientadora: Prof[a]. Dr[a]. Rogéria Cristiane Gratão de Souza

Dissertação de Mestrado elaborada junto ao Programa de Pós-Graduação em Ciência da Computação – Área de Concentração em Sistemas de Computação, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

UNESP
2014

Vanessa dos Anjos Borges

Apoio à Gerência de Requisitos de Software utilizando *Design Rationale*

Dissertação de Mestrado elaborada junto ao Programa de Pós-Graduação em Ciência da Computação – Área de Concentração em Sistemas de Computação, como parte dos requisitos para a obtenção do título de Mestre em Ciência da Computação.

Banca Examinadora

Profª. Drª. Rogéria Cristiane Gratão de Souza
UNESP – São José do Rio Preto
Orientadora

Profª. Maria Istela Cagnin Machado
UFMS - Universidade Federal do Mato Grosso do Sul

Prof. Leandro Alves Neves
UNESP – São José do Rio Preto

São José do Rio Preto
27/Fevereiro/2014

Dedico esse trabalho

Aos meus familiares, Gilson e Sandra, meus pais, Patrícia e Priscila, minhas irmãs, e ao meu esposo Leonardo, pelo amor, carinho e compreensão.

AGRADECIMENTOS

Agradeço primeiramente a Deus. Por sua providência, foi me dada força e perseverança para alcançar meus objetivos e não perder o fôlego diante das dificuldades.

Agradeço a minha orientadora Profª. Drª. Rogéria Cristiane Gratão de Souza pelo companheirismo e orientação, fundamentais na realização desse trabalho.

Aos meus queridos pais Gilson e Sandra, e minhas irmãs Patrícia e Priscila, que me acompanharam pacientemente durante essa caminhada, e que com amor acolheram minhas ansiedades e preocupações, trazendo conforto e carinho para que eu não desanimasse.

Ao Leonardo, meu marido, meu companheiro e eterno amor. Desde o princípio foi meu principal motivador, incentivou-me a continuar meus estudos para que realizasse meus sonhos.

À Universidade Estadual Paulista “Júlio de Mesquita Filho”, Instituto de Biociências, Letras e Ciências Exatas, de São José do Rio Preto e aos responsáveis pelo Programa de Pós Graduação Interunidades de Ciência da Computação, agradeço pela oportunidade e seriedade no trabalho realizado, e à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo fomento à essa pesquisa, que foi de fundamental auxílio.

SUMÁRIO

Capítulo 1 – Introdução 1
1.1 Contextualização 1
1.2 Motivação 2
1.3 Objetivos 3
1.4 Justificativa 4
1.5 Metodologia 6
1.6 Organização do Trabalho 7
Capítulo 2 – Fundamentação Teórica 8
2.1 Considerações Iniciais 8
2.2 Gerência de Requisitos 9
2.3 *Design Rationale* 14
2.3.1 Abordagens do *Design Rationale* 17
2.3.2 Captura de *Design Rationale* 18
2.3.3 Esquemas de Representação de *Design Rationale* 20
2.3.4 Recuperação de *Design Rationale* 26
2.4 Cenário e Tendências 26
2.5 Considerações Finais 30
Capítulo 3 – Expansão da Ferramenta de Suporte à Elicitação de Requisitos 32
3.1 Considerações Iniciais 32
3.2 Funções da Ferramenta de Suporte à Elicitação de Requisitos 33
3.3 Expansão da FSER 36
3.4 Ferramenta de Suporte à Engenharia de Requisitos 39
3.4.1 Reestruturação da Interface 39
3.4.2 Diagramas de Casos de Uso 41
3.4.3 Diagrama do Modelo de Design Reflexivo 44
3.4 Considerações Finais 45
Capítulo 4 –Avaliação da Ferramenta de Suporte à Engenharia de Requisitos 47
4.1 Considerações Iniciais 47
4.2 Definição e Planejamento do Experimento 48
4.2.1 Definição do Experimento 48
4.2.2 Planejamento do Experimento 49
4.3 Execução do Experimento 54
4.4 Apresentação e Análise de Resultados do Experimento 55
4.4.1 Turma A – São José do Rio Preto 56
4.4.2 Turma B – Presidente Prudente 59
4.5 Avaliação do Treinamento 62

4.6 Considerações Finais 63
Capítulo 5 – Conclusão 65
5.1 Considerações Finais 65
5.2 Contribuições 65
5.3 Trabalhos Futuros 66
Referências Bibliográficas 67
Apêndice A: Documentos de Requisitos de Software 72
Apêndice B: Atas de Reuniões 75
Apêndice C: Formulários 80

LISTA DE FIGURAS

Figura 1. A atividade de Gerência de Requisitos (Adaptado de Sommerville, 2003)..... 9
Figura 2. O vocabulário IBIS (Adaptado de Lee e Lai, 1991)..... 20
Figura 3. Exemplo de utilização da notação IBIS (Adaptado de Lee e Lai, 1991)..... 21
Figura 4. O vocabulário Potts & Bruns (Adaptado de Potts e Bruns, 1988).. 21
Figura 5. Exemplo utilizando a notação Potts & Bruns..... 21
Figura 6. O vocabulário DRL (Adaptado de Lee e Lai, 1991)..... 22
Figura 7. Exemplo utilizando a notação DRL..... 22
Figura 8. O vocabulário QOC (MCLEAN et al., 1991)..... 23
Figura 9. Exemplo utilizando a notação QOC..... 23
Figura 10. Objetos do Modelo de Design Reflexivo e seus respectivos elementos de origem..... 24
Figura 11. O vocabulário do Modelo de Design Reflexivo (Adaptado de Louridas e Loucopulos, 2000)..... 25
Figura 12. Exemplo utilizando o Modelo de Design Reflexivo (Adaptado de Louridas e Loucopulos, 2000)..... 25
Figura 13. Diagramas de Casos de Uso da FSER..... 34
Figura 14. Diagrama de Casos de Uso da expansão da FSER..... 38
Figura 15. Página Inicial da FSER..... 40
Figura 16. Nova interface padrão da FSER..... 41
Figura 17. Opção para geração de Diagramas de Casos de Uso..... 42
Figura 18. Funcionalidade de geração de Diagramas de Casos de Uso na FSER..... 42
Figura 19. Consulta às versões do Diagramas de Casos de Uso na FSER.. 43
Figura 20. Relação entre Diagramas de Casos de Uso e narrativas de Casos de Uso..... 43

Figura 21. Opção para geração de Diagramas do Modelo de Design Reflexivo na FSER..... 44
Figura 22. Funcionalidade de geração de Diagramas de Modelo de Design Reflexivo na FSER (Adaptado de Louridas e Loucopulos, 2000)..... 45
Figura 23. Consulta às versões do Diagramas do Modelo de Design Reflexivo na FSER..... 45
Figura 24. Avaliação da Complexidade das abordagens do experimento da Turma A..... 57
Figura 25. Avaliação da Eficiência das abordagens do experimento da Turma A..... 58
Figura 26. Avaliação da Motivação das abordagens do experimento da Turma A..... 59
Figura 27. Avaliação da Complexidade das abordagens do experimento da Turma B..... 61
Figura 28. Avaliação da Eficiência das abordagens do experimento da Turma B..... 61
Figura 29. Avaliação da Motivação das abordagens do experimento da Turma B..... 62
Figura 30. Avaliação do nível de satisfação dos participantes quanto ao curso e o treinamento sobre a FSER..... 63

LISTA DE TABELAS

Tabela 1. Critérios de Avaliação e Conceitos para avaliação de Ferramentas..... 28
Tabela 2. Avaliação de ferramentas que automatizam atividades de Engenharia de Software..... 30
Tabela 3. Hipóteses Nula H_0, Hipótese Alternativa H_{a0} e sua respectiva métrica..... 49
Tabela 4. Hipóteses Nula H_1, Hipótese Alternativa H_{a1} e sua respectiva métrica..... 50
Tabela 5. Hipóteses Nula H_2, Hipótese Alternativa H_{a2} e sua respectiva métrica..... 50
Tabela 6. Descrição da métrica Complexidade para a realização do experimento..... 50
Tabela 7. Descrição da métrica Eficiência para a realização do experimento..... 51
Tabela 8. Descrição da métrica Motivação para a realização do experimento..... 51
Tabela 9. Validades do experimento..... 54
Tabela 10. Cronograma de Execução..... 54
Tabela 11. Perfil dos Participantes da Turma A - São José do Rio Preto..... 56
Tabela 12. Perfil dos Participantes da Turma B - Presidente Prudente 59

LISTA DE ABREVIATURAS E SIGAS

CSS	*Cascading Style Sheet*
DRL	*Decision Representation Language*
ERS	Especificação de Requisitos de Software
FSER	Ferramenta de Suporte à Engenharia de Requisitos
GEPES	Grupo de Estudos e Pesquisas em Engenharia de Software
HTML	*Hyper Text Markup Language*
IBIS	*Issue Based Information System*
IEEE	*Institute of Electrical and Electronics Engineers*
KDD	*Knowledge Discovery in Databases*
MVC	*Model – View – Controller*
QOC	*Question, Option and Criteria*
UML	*Unified Modeling Language*

RESUMO

As necessidades dos usuários e partes interessadas no desenvolvimento de um software, denominadas requisitos, mudam constantemente, seja por falta de informações ou por mudanças organizacionais ou ambientais. A gerência desses requisitos gera um grande volume de informações que podem ser reutilizadas para consolidar o conhecimento dos envolvidos no projeto. Diversas são as técnicas para formalização desse conhecimento, destacando-se a técnica de *Design Rationale* que trata da organização da lógica de raciocínio utilizada para projetar artefatos. Neste contexto, este trabalho apresenta um estudo sobre a utilização de *Design Rationale* na Gerência de Requisitos de software, integrando essa técnica a um sistema web de apoio à Engenharia de Requisitos de Software, denominado Ferramenta de Suporte à Engenharia de Requisitos - FSER. Para avaliação dessa integração, foi realizado um Experimento Controlado, com a participação de alunos de graduação em Ciência da Computação e profissionais da área de Tecnologia da Informação, constatando sua contribuição no que concerne a disseminação do conhecimento e com a formação de uma base de dados que apóie o desenvolvimento de projetos futuros.

Palavras-Chave: Engenharia de Requisitos, Gerência de Requisitos, *Design Rationale*.

ABSTRACT

The users and stakeholders needs identified during software development, called requirements often change, either for lack of information or due to organizational or environmental changes. The management of these requirements generates a large volume of information that can be re-used to consolidate the knowledge between users and stakeholders of project. There are several techniques to formalize this knowledge, emphasizing Design Rationale technique which dealing with the logical reasoning organization used to design artifacts. In this context, this study presents the application of Design Rationale in Software Requirements Management, integrating this technique to a web support system for Software Requirements Engineering, called Requirements Engineering Support Tool - REST. For assessment this integration, it was performed a Controlled Experiment, involving Computer Science undergraduate students and Information Technology professionals, recognizing its contribution regarding knowledge dissemination and a database creation supporting the development of future projects..

Keywords: Requirements Engineering, Requirements Management, Design Rationale.

Capítulo 1 – Introdução

1.1 Contextualização

Dado o crescimento da demanda por sistemas de informações nas organizações, existe a preocupação de que a tecnologia empregada nos negócios esteja em conformidade com o que é necessário para o crescimento do mesmo.

Para desenvolver e entregar um software de qualidade é necessário olhar atentamente para o papel que o software irá desempenhar, entendendo os requisitos específicos que precisam ser satisfeitos (PRESSMAN, 2010).

A função de gerenciar requisitos enquanto são elicitados, analisados, negociados, validados e documentados é fundamental, pois os requisitos de um software estão suscetíveis a mudanças e evoluções frequentes. É de fundamental importância que tais mudanças e evoluções sejam comunicadas e documentadas a toda a equipe de desenvolvimento, evitando retrabalho.

Segundo Camponogara e Silveira (2009) ao considerar que projetos de software possuem uma grande quantidade de informação e um grande volume de decisões tomadas ao longo do seu desenvolvimento, é importante a existência de alguma técnica para documentar e manter estas decisões, especialmente, os motivos pelos quais estas foram tomadas.

Uma das técnicas que têm mostrado sucesso para a resolução desse problema é o *Design Rationale*. Diversas propostas envolvendo o armazenamento das razões por detrás das decisões em um projeto de software podem ser encontradas, tendo como finalidade fornecer documentação adicional a novos

participantes em um projeto, suporte à manutenção de software, entre outros (BURGE, 2008).

Essa técnica mantém de maneira sistêmica e organizada as informações relacionadas às tomadas de decisão realizadas no contexto de um projeto, mantendo uma base histórica que auxilie no desenvolvimento de projetos futuros (MORAN e CARROL, 1996).

Mesmo que o conhecimento sobre os produtos e processos possam ilustrar as soluções criadas e os cursos de ações envolvidas em sua criação, o processo de raciocínio, referente às soluções encontradas e ações realizadas, ainda é ausente a partir de uma representação integrada do conhecimento. O *Design Rationale* é uma técnica eficaz de captar esta falta de uma representação integrada do conhecimento que pode ser vista como um ativo valioso intelectual de uma empresa (WANG et al., 2012).

1.2 Motivação

O gerenciamento de requisitos é uma atividade da Engenharia de Requisitos que requer grande cuidado e atenção. Segundo Sommerville (2011) o relacionamento existente entre os requisitos e suas dependências são os principais aspectos ligados à Gerência de Requisitos. Também é necessário observar que durante o desenvolvimento de um projeto de um software, novos requisitos podem surgir e mudanças podem acontecer em requisitos existentes.

Nesse sentido, considerando a complexidade dos sistemas atuais, para evitar que a evolução e o surgimento desses requisitos possam causar impactos negativos no andamento do projeto é preciso utilizar ferramentas e métodos que apoiem as tomadas de decisão, fornecendo informações precisas e cruciais em tempo hábil.

No contexto da Gerência de Requisitos, é fundamental a retenção e disponibilidade de informações que sejam relevantes para o desenvolvimento de projetos de software. É possível encontrar na literatura diversos trabalhos e pesquisas direcionadas para a área de Gerência de Requisitos, mas que focam apenas partes desse processo, não as decisões e razões que permeiam esta atividade no projeto como um todo.

Assim, tem-se diferentes trabalhos envolvendo apoio computacional para a aplicação da técnica de *Design Rationale*. A ferramenta SEURAT (BURGE e BROWN, 2008) permite a utilização do *Design Rationale* para identificar a inconsistências durante o processo de manutenção de software. Já em Martinie et al. (2010) a ferramenta DREAMER utiliza o *Design Rationale* na definição da relação entre requisitos no domínio de Iteração Humano-Computador (IHC) tratando das modificações realizadas na interface dos sistemas e fazendo com que fatores como usabilidade e ergonomia de sistemas sejam avaliados. Em Santos e Medeiros (2011) é apresentado um software que automatiza a técnica de *Design Rationale* denominado KUABA, que foi aplicado mais especificamente na elicitação dos requisitos, contribuindo para um melhor raciocínio sobre requisitos e como modelá-los, já que os engenheiros de requisitos precisam avaliar cuidadosamente as justificativas para suas decisões.

Tais ferramentas definem a utilização do *Design Rationale* em parcelas segmentadas no desenvolvimento do software. Além disso, algumas delas utilizam-se de meios semiautomáticos de recuperação da informação, exigindo que a organização já possua determinado grau de maturidade para implantação de tais propostas.

Sendo assim, o mercado carece de ferramentas que executem a gestão do conhecimento e auxílio na tomada de decisões relevantes à Gerência de Requisitos de software, trazendo facilidade para a gestão dessas informações na organização. Pode-se então afirmar que ao aplicar o *Design Rationale* à atividade de Gerência de Requisitos, as atividades da Engenharia de Software tornam-se cada vez mais produtivas e aprimoradas.

1.3 Objetivos

Para realizar o gerenciamento das mudanças e evolução nos requisitos é necessário manter o rastreamento dos mesmos. Rastreabilidade é o grau em que cada elemento de um produto de desenvolvimento de software estabelece sua razão de existência, ou seja, o grau em que cada elemento do software referencia o requisito que o satisfaz (IEEE, 1996). Para Leffingwell e Wridrig (2003) a rastreabilidade de requisitos é uma técnica que desenvolve e melhora a qualidade e

confiabilidade do software, devendo ser aplicada durante as atividades de Engenharia de Requisitos.

Assim, este trabalho está inserido no contexto da Ferramenta de Suporte à Elicitação de Requisitos – FSER, previamente desenvolvida por membros do Grupo de Estudos e Pesquisas em Engenharia de Software – GEPES, do Instituto de Biociências, Letras e Ciências Exatas – IBILCE/UNESP. A FSER foi concebida inicialmente com o intuito de possibilitar ao analista a aplicação remota de diferentes técnicas de elicitação de requisitos, viabilizando a identificação dos requisitos do sistema por meio de uma ferramenta web (FUZII et al., 2009). Na sequência a FSER foi expandida com o intuito de apoiar as atividades de validação dos requisitos, utilizando a técnica de *checklist* (ITO et al., 2011). Em sua versão atual, a FSER contempla ainda suporte à especificação de requisitos (SIMONATO et al., 2012), tendo como base o padrão IEEE Std 830 (IEEE, 1998) e ao controle de rastreabilidade entre os requisitos e os artefatos de software (MIRANDA, 2013).

Portanto, observa-se que atualmente a ferramenta não dispõe de nenhum registro de experiências em projetos passados, para obtenção de informações sobre as modificações nesses requisitos e os motivos que levaram à ocorrência de tais modificações.

1.4 Justificativa

Diante desse cenário, o objetivo desse trabalho é, portanto, possibilitar o uso integrado do *Design Rationale* às demais atividades do processo de Engenharia de Requisitos, por meio da inserção à FSER de um módulo através do qual as organizações possam armazenar informações referentes às tomadas de decisões sobre as alterações e desenvolvimento dos requisitos dos projetos de desenvolvimento de software. Assim, a organização poderá utilizar um repositório de lições aprendidas vinculado aos projetos realizados, utilizando uma linguagem de captura formal de acordo com a técnica de *Design Rationale*, que permite o registro das experiências e tomadas de decisão em torno da gerência dos requisitos de um software, facilitando a compreensão e agilidade no desenvolvimento de novos projetos de desenvolvimento de software similares

A pesquisa relatada em Archibald e Prado (2012) realizada nos anos de 2005, 2006, 2008 e 2010 avaliou o grau de maturidade de diversas organizações, públicas e privadas, de pequeno, médio e grande porte, no âmbito de projetos de desenvolvimento de software. Na pesquisa realizada no ano de 2010, com a participação de 345 empresas brasileiras, foram levantados diversos pontos que demonstravam as causas no fracasso de projetos. O item mais pontuado foi relativo à mudança de escopo do produto, sendo que 76% dos projetos desenvolvidos as mudanças das necessidades dos clientes e alterações necessárias não foram previstas nem controladas, causando atrasos e aumento de custo no projeto.

Em sua obra, Patton (2001) afirma que 50% dos problemas e/ou defeitos de software são decorrentes da fase da especificação de requisitos, sendo que as principais causas de fracasso em projetos provém de requisitos incompletos.

O Standish Group, instituição formada em 1985 com o objetivo de coletar informações sobre fracasso de projetos da área da Tecnologia da Informação, relatou, em seu relatório CHAOS Manifesto de 2011, que dos projetos contemplados pela pesquisa, apenas 37% deles foram rotulados como projetos bem sucedidos. A taxa de projetos considerados fracassados foi de 21% (STANDISH GROUP, 2011). Dentre os motivos para o fracasso desses projetos, destacam-se entre requisitos incompletos, expectativas não realistas e mudanças de requisitos.

Nesse contexto, é possível constatar a carência de recursos que possibilitem às organizações de desenvolvimento de software um efetivo acompanhamento e monitoramento da evolução e motivação às quais os projetos estão sujeitos, devido a diversos fatores internos e externos ao ambiente do projeto. Salienta-se que, quando se trata da mudança do projeto, põe-se em pauta essencialmente a mudança de seus requisitos.

As mudanças nos requisitos de um software são constantes, tornando-se fundamental gerenciá-las (HAZAN e LEITE, 2003). Os objetivos e estruturas conceituais dos requisitos elicitados, avaliados, especificados e analisados podem mudar por inúmeras razões, tais como defeitos a serem corrigidos, mudanças de prioridades, melhor entendimento do cliente sobre determinada necessidade, mudanças ambientais e organizacionais, entre outros (LAMSWEERDE, 2009).

A habilidade de rastrear as relações dos requisitos com os artefatos do software e verificar o impacto da mudança dos requisitos constitui uma das chaves

de sucesso do processo de desenvolvimento de software, minimizando os fracassos (LEFFINGWELL e WRIDRIG, 2003).

Equipes de desenvolvimento frequentemente são compostas por indivíduos com diferentes graus de conhecimento sobre determinados domínios. Essa atividade de gestão de conhecimento é relevante, pois amplia a visão dos integrantes sobre o projeto, aumentando o grau de contribuição de cada um para o mesmo.

Com o passar dos anos, a Engenharia de Requisitos evoluiu, proporcionando o surgimento de novas técnicas e ferramentas para evitar que os projetos falhem por falta de concordância entre as necessidades dos clientes e a equipe de desenvolvimento do projeto. Porém, ainda existe um campo a ser explorado para a incorporação da gestão do conhecimento nessas ferramentas. Diversas são as técnicas e os métodos para a formalização da gestão do conhecimento como *Knowledge Discovery in Databases* (KDD) (LU et al., 1997) e Ontologias de Domínio (CIMIANO e REYLE, 2006), dentre as quais se encontra o *Design Rationale*.

Para Moran e Carrol (1996), *Design Rationale* refere-se ao raciocínio que justifica um projeto e a explicação sobre a escolha de determinadas estruturas frente a outras alternativas. *Design Rationale* abrange o conhecimento empregado durante a construção de um artefato, evidenciando decisões tomadas, razões, justificativas, outras alternativas e argumentos advindos da mesma (LEE, 1997).

Através da aplicação de técnicas de *Design Rationale*, é possível expressar as alternativas e principais questões levantadas para apoiar a tomada de decisão, reter o conhecimento produzido pelos integrantes de uma equipe de projeto, como forma de minimizar o impacto da perda de um dos mesmos e auxiliar o desenvolvimento de projetos futuros baseando-se nas experiências obtidas com o tempo (KLEIN, 1997).

1.5 Metodologia

Para o desenvolvimento deste trabalho, inicialmente foi realizado um levantamento bibliográfico a respeito das duas principais temáticas consideradas:

Gerência de Requisitos e *Design Rationale*. Além disso, também foi efetuado um levantamento e estudo das principais ferramentas que abordam o mesmo propósito.

Após essa etapa, esforços foram concentrados na Ferramenta de Suporte à Elicitação de Requisitos - FSER, a fim de aprofundar conhecimentos sobre as tecnologias e artefatos utilizados como o documento de requisitos, projeto de interface, estrutura de armazenamento de informações, linguagem de programação, entre outros. Ao finalizar essa etapa, foi possível definir os requisitos necessários para a expansão da FSER, visando a inserção de um novo módulo.

Como etapa final deste projeto, tem-se a integração da nova funcionalidade à FSER, e sua avaliação por pessoas com experiência no desenvolvimento de software da área de Ciência da Computação, a fim de detectar os pontos fracos da ferramenta a serem corrigidos, bem como suas reais contribuições.

1.6 Organização do Trabalho

Nesta seção é apresentada uma breve descrição dos capítulos subsequentes deste trabalho:

Capítulo 2 – Fundamentação Teórica: neste capítulo são abordados os principais aspectos da Engenharia de Requisitos, pondo em pauta a atividade de Gerência de Requisitos além de apresentar um estudo sobre a técnica de *Design Rationale*, relacionando suas principais características, abordagens, atividades de captura, formalização e pesquisa;

Capítulo 3 – Expansão da Ferramenta de Suporte à Elicitação de Requisitos: neste capítulo são resumidamente apresentadas as funções atuais da FSER, bem como as funções propostas para sua expansão;

Capítulo 4 – Descrição dos Resultados: neste capítulo são apresentados os resultados obtidos com a integração das novas funções à FSER e os resultados de um experimento realizado com a utilização da ferramenta;

Capítulo 5 – Discussões Finais: neste capítulo são apresentadas as considerações finais do trabalho, descrevendo as perspectivas de contribuições alcançadas e os trabalhos futuros estabelecidos.

Capítulo 2 – Fundamentação Teórica

2.1 Considerações Iniciais

A tradução da necessidade de um negócio ou da resolução de um determinado problema em uma coleção de métodos e sequências a serem processadas por um computador não é uma tarefa trivial. Ao desenvolver um projeto de software, que por sua própria definição e natureza é único, definir com precisão e corretude tudo aquilo que ele deve realizar em seu ambiente de produção é altamente complexo (SOMMERVILLE, 2011).

A Engenharia de Requisitos de software é o ramo da Engenharia de Software que se preocupa com os objetivos do mundo real diante das funções a serem desenvolvidas e restrições a serem levadas em conta no desenvolvimento de um software.

Neste contexto, a atividade de Gerência de Requisitos tem por finalidade geral organizar, verificar e assegurar que os requisitos elicitados serão contemplados no desenvolvimento de um software correto. Porém, muitas vezes o conhecimento adquirido em projetos anteriores e, que pode ser visto como um ativo valioso intelectual de uma empresa (WANG et al., 2012), nem sempre está disponível. Assim, considera-se de grande importância o estabelecimento de uma forma eficaz de captar, de maneira integrada, o conhecimento adquirido durante o projeto de artefatos[1], como o *Design Rationale*, de maneira a permitir que o aprendizado obtido

[1] Um artefato pode ser definido como a representação de qualquer informação gerada, alterada ou usada durante o processo de desenvolvimento de software (JACOBSON et al., 1999).

durante o desenvolvimento de um projeto possa ser disseminado a todos os interessados, como forma de contribuir para a qualidade do produto resultante.

Assim, na seção 2.2 é abordada a temática de Gerência dos Requisitos, visando apresentar as principais atividades desse processo. A seguir, na seção 2.3 é apresentada uma visão geral da estrutura estabelecida para o *Design Rationale*, evidenciando os benefícios com sua utilização na gestão do conhecimento em projetos de artefatos de software. Já na seção 2.4 são apresentados os trabalhos e ferramentas relacionadas ao tema proposto, visando destacar as contribuições do presente trabalho. Por fim, na seção 2.5 são apresentadas as considerações finais.

2.2 Gerência de Requisitos

Os requisitos de um software estão em constante mudança e evolução, variando conforme os recursos, ambiente e tecnologias adicionadas. Essas mudanças e evoluções devem ser gerenciadas e tratadas para que problemas de custo e tempo não comprometam o projeto.

A Engenharia de Requisitos é um processo de abordagem contínua que acompanha todo o ciclo de vida do desenvolvimento de um produto. A Gerência de Requisitos, atividade constituinte desse processo, é composta pela aplicação de práticas de gerência de mudanças, medições, qualidade e rastreabilidade (BERENBACH et al., 2009). Na Figura 1 é exemplificado como essa atividade de envolve e suporta as atividades do processo de Engenharia de Requisitos.

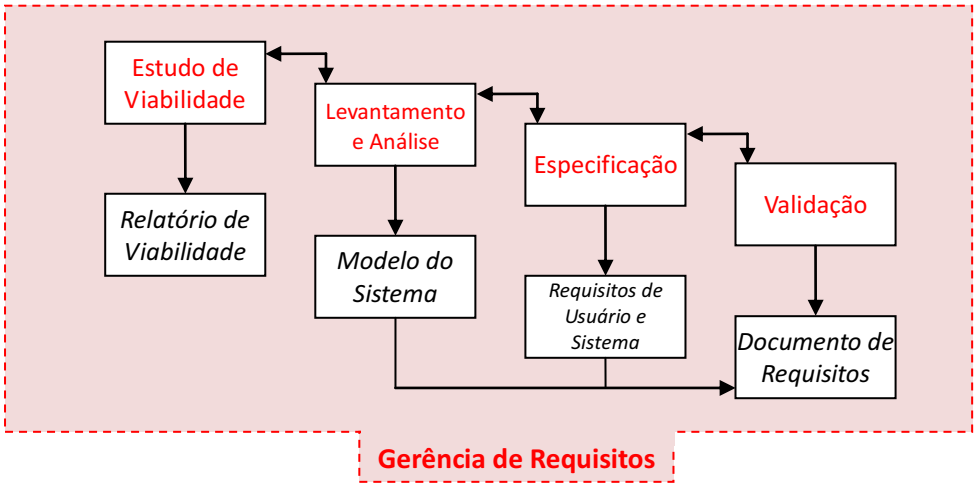


Figura 1. A atividade de Gerência de Requisitos (adaptado de Sommerville, 2011).

A gerência de mudanças no âmbito da atividade de Gerência de Requisitos de software envolve atividades relacionadas com o controle de versões dos artefatos relacionados ao processo de engenharia de requisitos, incluindo documentos fonte de requisitos, atas de reuniões, requisições de *stakeholders*, e documentos gerados pelos requisitos (especificação, modelagem e testes de requisitos). Além disso, durante a realização dessa atividade é importante a avaliação do impacto que tais mudanças terão sobre o desenvolvimento do software (BERENBACH et al., 2009).

As medições realizadas durante a gerência dos requisitos auxiliam efetivamente na execução das atividades de desenvolvimento e manutenção do software e no acompanhamento do progresso da execução das atividades de desenvolvimento do software. Os indicadores estabelecidos para o desenvolvimento do projeto podem ser utilizados também para guiar ou melhorar o desenvolvimento do processo de engenharia de requisitos (BERENBACH et al., 2009). Tais métricas e medições contribuem significativamente na qualidade do produto de software, garantindo que a base dos requisitos identificados seja composta por bons requisitos[2].

Para a garantia da qualidade, também se considera importante a documentação das decisões geradas durante a atividade de Gerência de Requisitos e também das decisões tomadas durante o projeto de desenvolvimento de software.

As decisões realizadas sobre requisitos muitas vezes são baseadas sobre pressupostos ocultos e/ou pessoais dos envolvidos. Fazendo-se suposições explícitas, e conectando essas suposições à raciocínios bem argumentados, é possível que as decisões possam ser revisadas e reaproveitadas, sem a necessidade de retrabalho. Ao explicar a razão da existência ou mudança de um requisito, ao menos quando se trata de requisitos controversos, economiza-se tempo, custo e mão de obra (ALEXANDER e BEUS-DUKIC, 2009).

A rastreabilidade dos requisitos é considerada uma atividade fundamental na atividade de Gerência de Requisitos, pois descreve e acompanha o ciclo de vida de um requisito durante o desenvolvimento do software (GOTEL e FILKELSTEIN,

[2] A norma IEEE 830 (IEEE, 1998) estabelece características para que um requisito tenha qualidade: correto, não ambíguo, completo, consistente, relevante, verificável, modificável e rastreável.

1994). Ela garante que os requisitos identificados e estabelecidos para a construção do produto de software sejam acompanhados durante seu desenvolvimento.

Certas habilidades gerenciais são sugeridas por Leffingwell e Widrig (2003), com o intuito de tornar mais produtivo o gerenciamento dos requisitos do software, a saber:

a) **Analisar o problema:** O tempo empregado em entender os problemas do negócio, as necessidades das partes interessadas e a natureza do ambiente na qual o software será utilizado é fundamental para que não haja retrabalho por falta de abstração do problema a ser resolvido;
b) **Entender as necessidades dos usuários e partes interessadas:** A habilidade de entender as necessidades das partes interessadas pode ser complementada pela habilidade de negociar os requisitos a serem implementados, ou seja, a cada nova iteração do projeto, as partes envolvidas podem considerar e discutir sobre novos requisitos a serem desenvolvidos no software. Negociação é um termo que pode ser definido como uma interação entre indivíduos a fim de se chegar a um senso comum, tendo início na expressão dos objetivos de cada um e fim em um acordo firmado entre as partes (ROBINSON e VOLKOV, 1998). O processo de negociação é dividido entre as seguintes atividades (AURUM e WOHLIN, 2005): Pré-Negociação - são conhecidos e analisados os objetivos das partes interessadas em cada iteração e, se necessário, reavaliados para garantir que se encontram de acordo com o produto que está em processo de construção; Negociação - são definidos os acordos em relação aos requisitos a serem implementados, reavaliados ou desconsiderados; Pós-Negociação - são enviados os *feedbacks* dos acordos estabelecidos e, se necessário, sugere-se renegociações;
c) **Definir o sistema:** Habilidade que busca empregar técnicas a fim de transformar a compreensão do problema a ser resolvido

e as necessidades dos usuários em uma definição inicial do software que irá suprir essas necessidades, sistematizando a especificação do comportamento do sistema e aumentando a quantidade de informações a serem gerenciadas. Segundo Aurum e Wohlin (2005) a maior parte de projetos de desenvolvimento de software possuem mais requisitos candidatos à implementação do que o custo e o tempo estabelecidos permitem implementar. Devido a esse problema, para definição do sistema como um todo, a capacidade de priorização de requisitos é de grande valia. Aspectos importantes a serem considerados na priorização dos requisitos são (AURUM e WOHLIN, 2005): Importância dos requisitos para o software que, por ser muito subjetiva, deve ser bem definida pelas partes interessadas do projeto; Custo da implementação dos requisitos, de forma que o orçamento possa ser administrado corretamente, evitando o fracasso do projeto devido à falta de recursos; Tempo de desenvolvimento do software, sendo ditado pela complexidade dos requisitos elicitados e riscos relacionados aos requisitos do projeto de software, identificando aqueles que precisam ser prevenidos, mitigados e até mesmo assumidos;

d) **Gerenciar o escopo:** a habilidade de gerenciar o escopo do produto a ser desenvolvido aumenta a probabilidade de sucesso do projeto. No escopo são estabelecidos todos os requisitos contemplados pelo sistema e todos os requisitos que não serão contemplados pelo sistema. As linhas de base de um projeto são estabelecidas a partir de seu escopo, para fornecer uma concepção inicial e, que permeia o projeto, a fim de estabelecer uma forma de controle, ou seja, não necessariamente todo o projeto seguirá a risca todos os valores estabelecidos como linha de base para o projeto. Em projetos de desenvolvimento de software, os requisitos estão sempre suscetíveis a mudanças e evoluções, seja por mudanças no ambiente em que irá operar ou, por adequações necessárias

devido a requisitos erroneamente compreendidos ou elicitados. Na atividade de Gerência de Requisitos, é fundamental a avaliação do impacto que a mudança desses requisitos terá no escopo do projeto, a fim de que as estimativas de tempo, custo e qualidade não sejam comprometidas (LAMSWEERDE, 2009);

e) **Refinar a definição do sistema:** esta habilidade visa examinar métodos para a elaboração, organização e comunicação dos requisitos do software. Dentre as técnicas utilizadas para uma melhor definição dos requisitos do sistema, podem ser citados: pseudocódigos que representam a especificação dos requisitos utilizando recursos da linguagem natural somados à estrutura de uma linguagem de programação; diagramas da *Unified Modeling Language* (UML) que modelam a estrutura, comportamento, arquitetura, estrutura de dados e processos de negócio de um software orientado a objeto; árvores e tabelas de decisão que expressam um conjunto de condições que, ao ocorrerem, desencadeiam uma ou mais ações;

f) **Desenvolver o sistema certo:** O foco principal dessa habilidade é de construir corretamente o sistema certo. Para tanto, é necessário garantir que os requisitos elicitados, analisados e validados estejam presentes no software desenvolvido. A rastreabilidade de requisitos é uma atividade que comprovadamente aumenta a qualidade e a confiabilidade do software (LEFFINGWELL e WIDRIG, 2003). A série de normas IEEE 610 (IEEE, 1996) define como rastreabilidade o grau em que uma relação pode ser estabelecida entre dois ou mais produtos do processo de desenvolvimento, especialmente os produtos que possuem uma relação de antecessor/sucessor ou mestre/subordinado uns com os outros, por exemplo, o grau de correspondência entre o requisito e o projeto de um dado componente de software. Complementando essa informação, a rastreabilidade é o grau em que cada elemento de um produto de desenvolvimento de software estabelece sua razão de existência; por exemplo, o grau em que cada elemento de um

> gráfico bolha referencia o requisito que o satisfaz (IEEE, 1996). Técnicas de Verificação e Validação são muito eficientes quando trata-se da rastreabilidade dos requisitos de software. As técnicas de Validação rastreiam os requisitos de acordo com os objetivos estabelecidos por eles e vice-versa, como a técnica de teste caixa preta. As técnicas de Verificação rastreiam requisitos entre as fases do produto, como do projeto do software até a sua programação. O objetivo da Verificação é assegurar que mesmo com os incrementos que o software incorpora, os requisitos estão presentes em todas as fases do produto. Uma das técnicas de Verificação para rastreabilidade dos requisitos do software é a Matriz de Rastreabilidade (LAUESEN, 2002).

Como o intuito de possibilitar que as atividades relacionadas à Gerência de Requisitos sejam efetivamente realizadas e as habilidades gerenciais sejam aperfeiçoadas, é importante proporcionar a um gerente de projeto o acesso às informações obtidas em experiências anteriores, por meio do armazenamento do conhecimento obtido durante o processo de desenvolvimento de software.

2.3 *Design Rationale*

A palavra *Design* remete à língua inglesa o significado de projeto, desenho, modelo, esquema. Assume-se, portanto, que *Design* é o processo de criar artefatos tangíveis a fim de atender às necessidades humanas. Já a palavra *Rationale*, da mesma língua inglesa, é o substantivo de racional e, significa análise racional, base lógica. Assim, *Rationale* é a razão, explicação, que se baseia em uma linha de raciocínio (MICHAELLIS, 2013).

Os primeiros trabalhos que mostram o surgimento do *Design Rationale* remetem aos anos 50, onde Toulmin (1958) tinha por objetivo criar uma visão lógica no estudo da prática do raciocínio através da argumentação, criando uma forma gráfica para estruturar seu raciocínio. A partir de então, diversos trabalhos como em Kunz e Rittel (1970) iniciaram os esforços para o desenvolvimento do *Design Rationale*.

Para Wang et al. (2012) o desenvolvimento de métodos para a recuperação de *Design Rationale* é útil como parte de uma gestão eficaz dos conhecimentos do projeto. O projeto e o desenvolvimento de produtos complexos envolve um processo complexo em que os projetistas realizam um grande número de atividades relacionadas à criação e avaliação de soluções, resolução de problemas levantados e tomada de decisões. A maior parte destas atividades depende do conhecimento e da experiência obtida em projetos anteriores, e requer uma eficiente reutilização do conhecimento.

Lee (1997) destaca diversos benefícios advindos da utilização de *Design Rationale* como:

- **Suporte para melhoria do projeto:** significativa contribuição ao clarificar a estrutura geral do processo de raciocínio e suporte à tomada de decisão;
- **Suporte para a melhoria da manutenção:** além de dar suporte ao projeto de artefatos, o *Design Rationale* fornece suporte à manutenção dos mesmos, extraindo linhas de raciocínio mais completas e sofisticadas do que as documentações estruturadas convencionais;
- **Suporte para a transferência de conhecimento:** o armazenamento e distribuição das informações capturadas e representadas pelo *Design Rationale* podem ser utilizadas para transferência de conhecimento do domínio para novos indivíduos que ainda não fazem parte da equipe envolvida no projeto;
- **Suporte para a documentação:** toda a informação armazenada em uma base de conhecimento utilizando *Design Rationale* serve como documentação dos processos de raciocínio lógico e apoia documentações e ativos da organização como documentação suplementar.

Moran e Carrol (1996) atribuem ao *Design Rationale* algumas dimensões que demonstram sua amplitude e abrangência:

- **Uma expressão da relação entre um artefato projetado, sua finalidade, a conceituação de seu projetista e as restrições contextuais para a realização do que é proposto:** *Design Rationale* possibilita definir a explicação entre estrutura e função de um artefato, além das exigências do domínio que permeiam o processo de tomada de decisões durante o projeto do artefato;
- **As razões lógicas para justificar um artefato projetado:** as razões lógicas traçam uma linha de raciocínio baseadas em argumentações, considerações e questionamentos a fim de justificar a existência do artefato;
- **Uma notação para as razões lógicas de um artefato projetado:** essa notação, seja ela informal, semi-formal ou formal, é a representação ou o formato para armazenar as razões, articulando as relações lógicas entre os artefatos e o contexto;
- **Um método de projetar um artefato em que a razão para isso é explícita:** refere-se ao método para projetar o artefato, em que uma notação é usada como uma ferramenta de processo de facilitação para orientar o processo de analisar cuidadosamente as razões para as escolhas certas ou passos no desenvolvimento do processo;
- **Documentação das razões da concepção de um artefato, das fases ou etapas do processo de *Design*, da história do projeto e de seus conceitos:** nesse sentido, a documentação pode ocorrer em vários níveis e por diversas razões, como para documentar todos os níveis do processo de projeção de um artefato ou para armazenar um conhecimento que possivelmente pode ser reutilizado no processo de criação ou manutenção de outros artefatos;
- **Uma explicação do porquê de um artefato ter sido projetado de determinada maneira:** essa explicação deve ser

> gerada, pois mesmo com a documentação em mãos, que tenta mostrar toda a lógica por detrás de um artefato, a antecipação de todas as questões que um novo artefato pode gerar podem não estar contidas ali.

Nas subseções a seguir, são apresentadas as abordagens, maneiras de captura, esquemas de representação e recuperação do *Design Rationale*.

2.3.1 Abordagens do *Design Rationale*

Existem duas abordagens para o desenvolvimento de sistemas baseados em *Design Rationale* sugeridas por Regli et al. (2000): abordagem orientada a processo e abordagem orientada a característica.

A primeira delas, a abordagem orientada a processo, é a mais usada comumente. Lara (2005) afirma que essa abordagem se aplica a fases do projeto em que os problemas são vagos, a solução tecnológica ainda não foi escolhida e existe pouca ou nenhuma padronização do artefato. São feitas diversas discussões a respeito dos requisitos dos artefatos e opções a serem investigadas, sendo uma fase mais dinâmica, onde o conhecimento se organiza de acordo com a evolução do projeto.

Em suma, a abordagem orientada a processo busca oferecer suporte ao processo de desenvolvimento através da análise reflexiva de problemas complexos que envolvem a obtenção e o armazenamento do raciocínio a fim de torná-lo um recurso reutilizável (SHUM et al., 1997).

A segunda abordagem, a abordagem orientada a característica, aplica-se a fases do projeto onde os artefatos são ricos em características técnicas, onde as regras e o conhecimento sobre o domínio de informação específico são considerados em cada tomada de decisão do projeto. Esse tipo de abordagem é geralmente utilizado para um contexto particular, usando estudos empíricos (LARA, 2005).

No âmbito deste trabalho, a abordagem que mais se assemelha ao realizado no processo de Gerência de Requisitos é a abordagem orientada a processo, onde diversos questionamentos e possibilidades desconhecidas muitas

vezes precisam ser consideradas para o desenvolvimento dos requisitos de um software.

2.3.2 Captura de *Design Rationale*

Para a Captura de *Design Rationale*, alguns autores apresentam diferentes alternativas para o desenvolvimento dessa atividade que é fundamental para a utilização da técnica de *Design Rationale*.

Regli et al. (2000) simplifica o processo de captura do *Design Rationale* dividindo-o basicamente em dois passos:

- **Registro do Conhecimento:** considera o armazenamento do máximo de informações possíveis durante o projeto;
- **Construção do *Design Rationale*:** refere-se à extração, organização e armazenamento do conhecimento produzido baseando-se no modelo de representação do *Design Rationale* e nas restrições do domínio.

Lee (1997), por sua vez, propõe quatro abordagens para a captura de *Design Rationale*, onde a captura pode emergir de diferentes ambientes e áreas:

- **Reconstrução:** pode ser realizada utilizando o raciocínio do indivíduo com um conhecimento do domínio já existente, com a realização de entrevistas com os envolvidos na concepção do artefato, ou capturar o raciocínio em forma bruta, a partir de documentações e ativos organizacionais, traduzindo em uma forma estruturada. A Engenharia Reversa também é uma maneira eficaz de capturar as informações do domínio. Na captura pela reconstrução é possível realizar uma reflexão mais cuidadosa na representação e estruturação das razões, sem interromper as atividades dos envolvidos no projeto do artefato;

- **Subproduto mercadológico:** as informações e raciocínios emergem naturalmente durante o projeto do artefato. Esta abordagem é atrativa porque o próprio indivíduo que realiza a captura das informações se beneficia do método e as justificativas do projeto são obtidas a um custo relativamente baixo (KAWAMOTO, 2007);
- **Aprendiz:** visa acompanhar as ações tomadas pelo responsável pelo projeto, questionando-o quando determinada ação gera discordâncias ou falta de entendimento, e estabelecendo uma linha de base (*baseline*) quando a ação estiver conforme o raciocínio estabelecido. A vantagem da captura Aprendiz é que essa interação e debate de informações evita a probabilidade de armazenamento de informações imprecisas ou incompletas. Geralmente, essa captura é realizada através de um sistema de informação;
- **Geração automática:** a captura das informações é realizada a partir da coleta de informações de um histórico de execuções, ou seja, informações já armazenadas sobre projetos e artefatos produzidos anteriormente pela organização, equipe de desenvolvimento e até mesmo em literatura específica, o qual deve ser armazenado com informações relevantes. A principal vantagem da captura pela geração automática é reutilizar informações, reduzindo tempo e custo. Porém, o custo inicial para realização dessa captura é alto, pois é necessário ter armazenado tal histórico durante um certo período para criar uma base de dados.

Para a implementação da funcionalidade de representação, armazenamento e recuperação de *Design Rationale* no contexto da Ferramenta de Suporte à Engenharia de Requisitos, a captura de *Design Rationale* se adéqua ao processo estabelecido por Regli et al. (2000). O objetivo deste modelo de captura é coletar o máximo de informações possíveis dos envolvidos, ativos e artefatos

durante o processo de engenharia e Gerência de Requisitos, realizando então a representação do *Design Rationale*.

2.3.3 Esquemas de Representação de *Design Rationale*

Ao realizar a captura do *Design Rationale*, é essencial que todo o conhecimento contido nele seja formalizado, utilizando-se de estruturas previamente estabelecidas. Diversos esquemas e modelos de representação de *Design Rationale* são apresentados pela literatura. Dentre as abordagens existentes, podemos destacar:

- **IBIS:** (*Issue Based Information System*) a notação foi desenvolvida em Kunz e Rittel (1970) com a proposta de representar a argumentação das atividades de projetistas. Desde então, algumas variações dessa representação foram criadas para representar *Design Rationale*. IBIS foi a primeira representação criada para o contexto de projeção e desenho de artefatos. Nas Figuras 2 e 3 são representados, respectivamente, o vocabulário da abordagem IBIS e um exemplo de sua utilização.

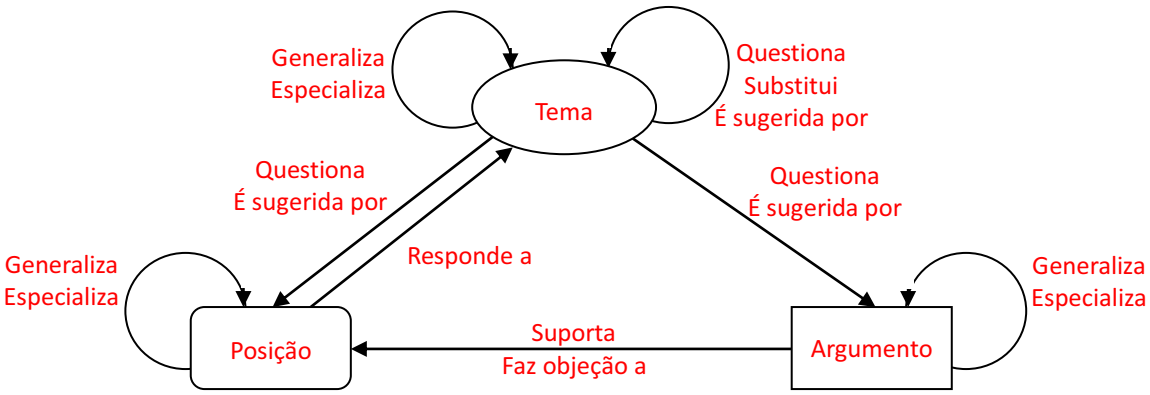


Figura 2. O vocabulário IBIS (adaptado de Lee e Lai, 1991).

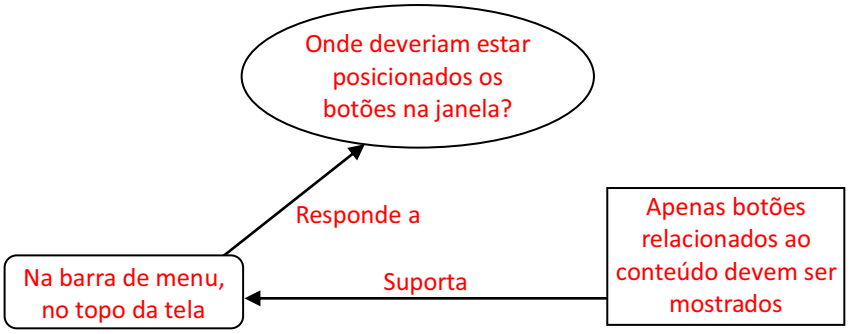


Figura 3. Exemplo de utilização da notação IBIS (adaptado de Lee e Lai, 1991).

- **Potts & Bruns:** essa notação foi desenvolvida baseando-se na notação IBIS, onde foi incluído a noção de artefatos intermediários, para representar a ligação entre as decisões e os artefatos de design (POTTS e BRUNS, 1988). Na Figura 4 é apresentado o vocabulário de Potts & Bruns e na Figura 5 um exemplo de sua utilização.

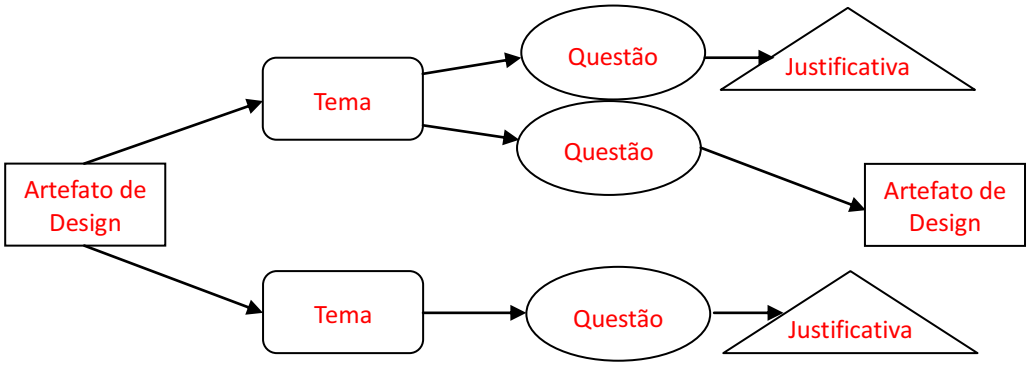


Figura 4. O vocabulário Potts & Bruns (adaptado de Potts e Bruns, 1988).

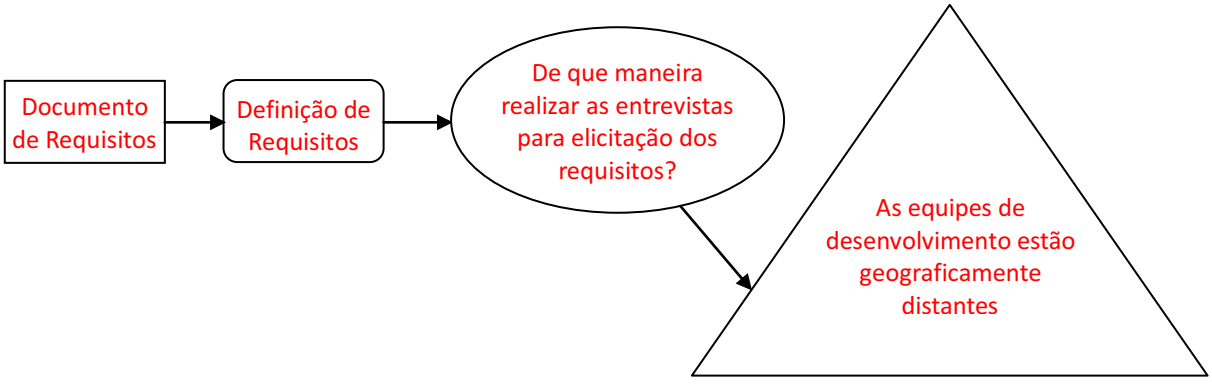


Figura 5. Exemplo utilizando a notação Potts & Bruns.

- **Potts:** essa notação é uma extensão da notação Potts & Bruns, onde é introduzido o conceito de Passos que se refere às ações a serem realizadas baseadas nas argumentações realizadas (Potts, 1989);
- **DRL** *(Decision Representation Language)***:** a notação DRL, proposta em Lee (1990) tem como objetivo compartilhar o conhecimento e dar suporte à decisão. É uma notação considerada de alta formalidade por possuir diversos objetos e um grande número de atributos que relacionam esses objetos. Nas Figuras 6 e 7, respectivamente, são mostrados o vocabulário da notação DRL e um exemplo utilizando essa notação.

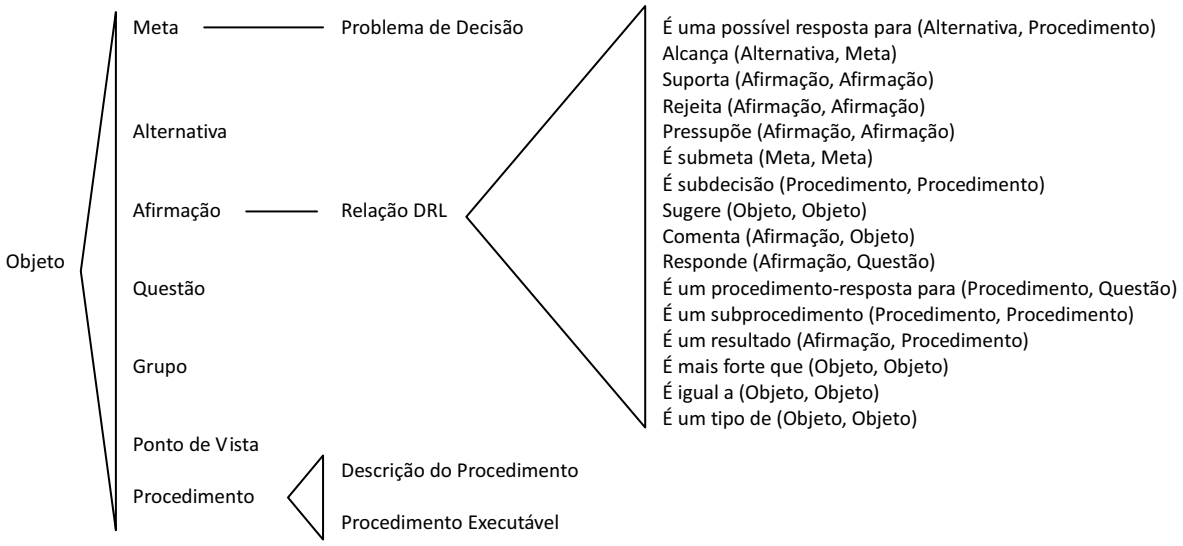


Figura 6. O vocabulário DRL (adaptado de Lee e Lai, 1991).

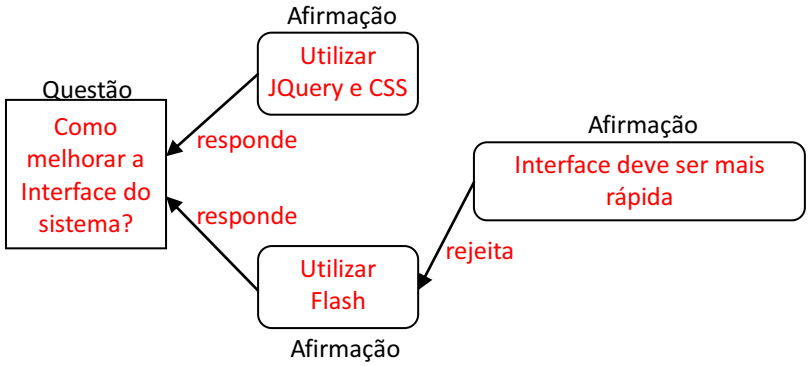


Figura 7. Exemplo utilizando a notação DRL.

- **QOC** (*Question, Option and Criteria*): é uma notação que coloca um artefato em um espaço de possibilidades e procura explicar os questionamentos em relação a esse artefato a partir de tais possibilidades (MCLEAN et al., 1991). Na Figura 8 são ilustrados o vocabulário e a estrutura utilizada pela representação QOC e na Figura 9 um exemplo de utilização.

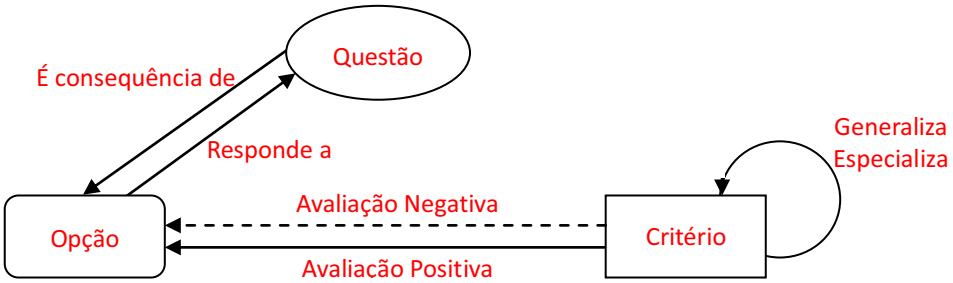


Figura 8. O vocabulário QOC (MCLEAN et al., 1991).

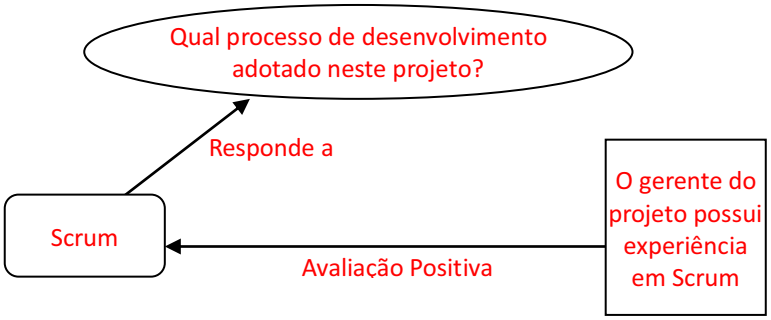


Figura 9. Exemplo utilizando a notação QOC.

- ***Reflective Design Model*** (Modelo de Design Reflexivo): a partir da análise de várias notações para representação de *Design Rationale*, Louridas e Loucopolos (2000) encontraram semelhanças inerentes a esses modelos e propuseram um novo modelo que utilizasse os conceitos dos elementos fundamentais para organizar as informações de uma maneira mais coerente. Na Figura 10 é mostrada a relação do Modelo de Design Reflexivo com as notações nas quais foi baseado. Observa-se que foram agrupados os principais objetos das representações mais difundidas de *Design Rationale*. O objeto

"Objetivos" relaciona-se a um questionamento a ser solucionado. O objeto "Hipóteses" define as alternativas ou opções que possam responder a um "Objetivo" (pode existir mais de uma "Hipótese" para um "Objetivo"). Os objetos "Ações de Design" mostram as decisões a serem tomadas sobre o problema proposto. Os objetos "Justificativas" reforçam as alegações feitas por meio das "Hipóteses". O ciclo de Design proposto é ilustrado na Figura 11. Na Figura 12 é mostrado um exemplo da utilização desse modelo.

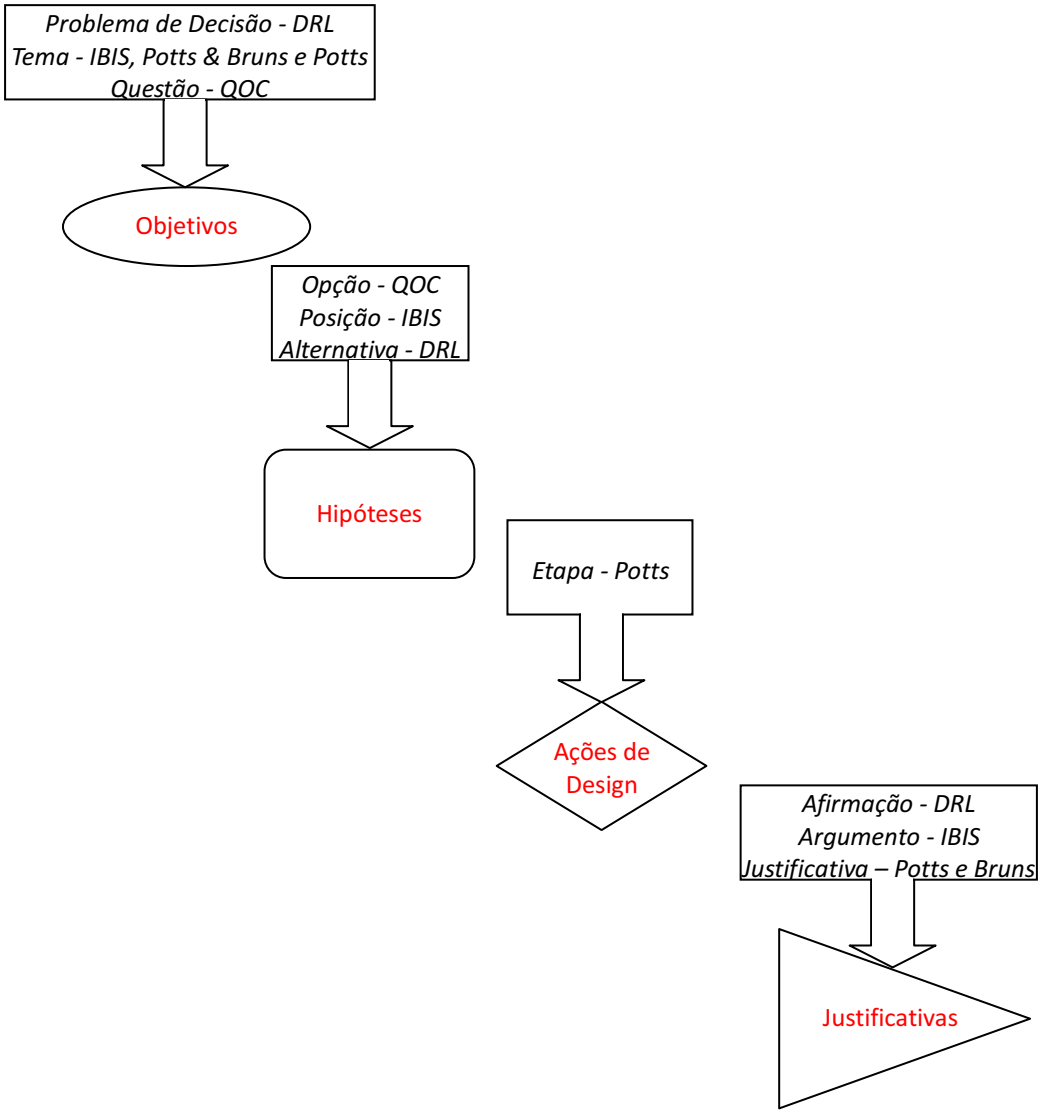


Figura 10. Objetos do Modelo de Design Reflexivo e seus respectivos elementos de origem.

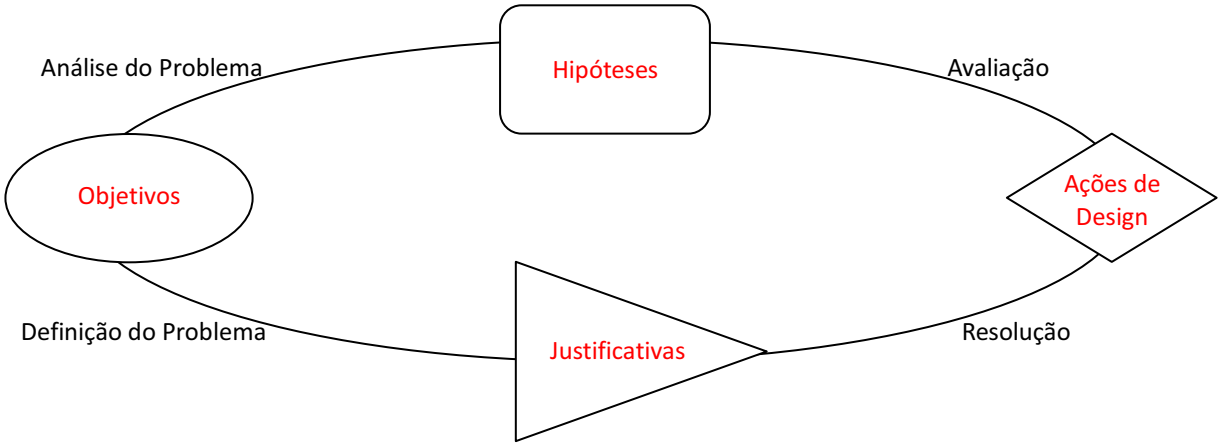


Figura 11. O vocabulário do Modelo de Design Reflexivo (Adaptado de Louridas e Loucopulos, 2000).

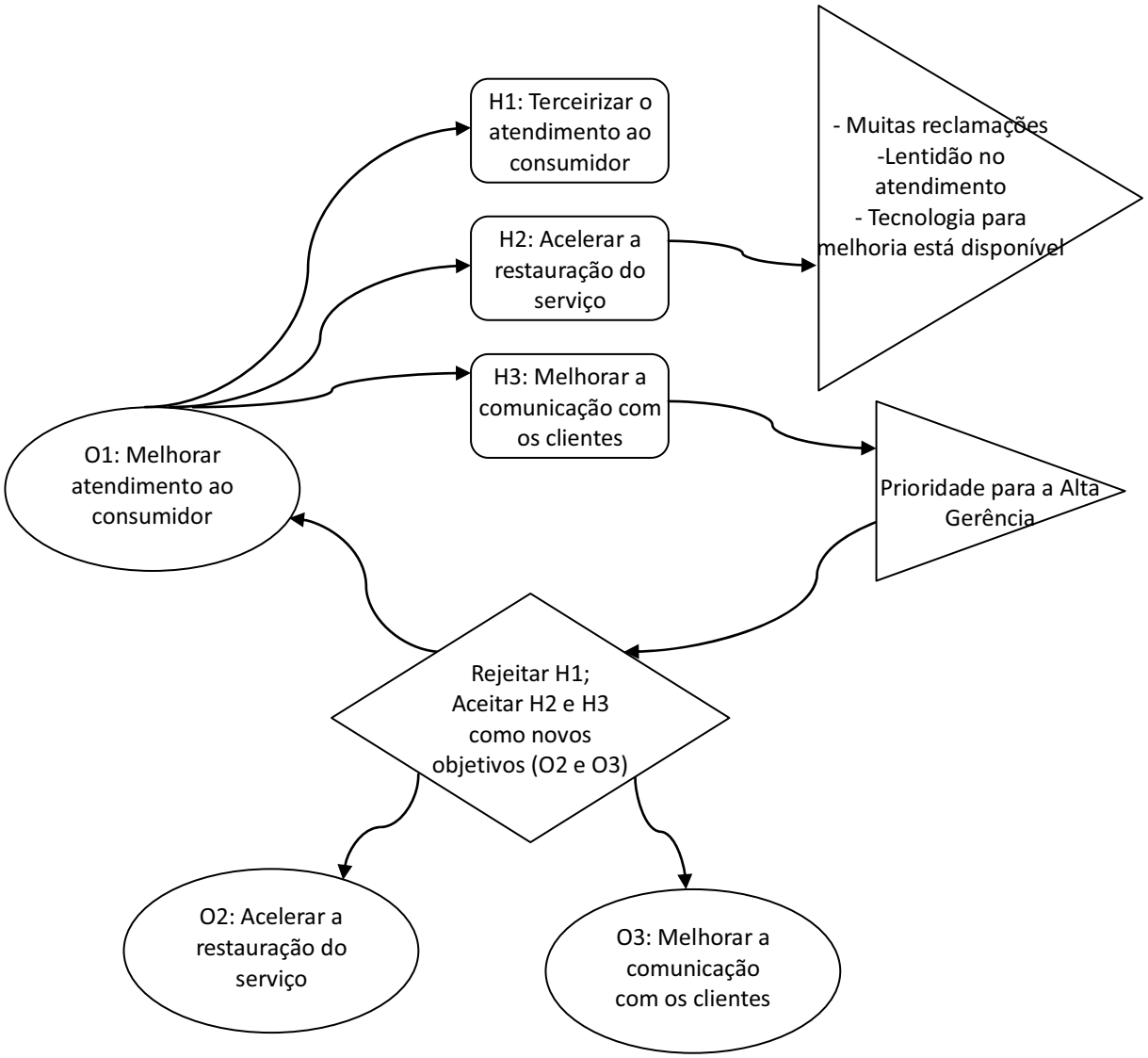


Figura 12. Exemplo utilizando o Modelo de Design Reflexivo (Adaptado de Louridas e Loucopulos, 2000).

No contexto de desenvolvimento deste trabalho, dada a abrangência proposta pelo Modelo de Design Reflexivo, tal representação foi implementada e incluída no contexto da Ferramenta de Suporte à Engenharia de Requisitos (FSER).

2.3.4 Recuperação de *Design Rationale*

Além de capturar e estruturar as informações para a formação de uma base de conhecimento, a recuperação do mesmo é uma atividade crucial para utilização e disseminação das informações processadas.

Regli et al. (2000) categorizam três diferentes abordagens para a recuperação das informações contidas nos modelos de representação de *Design Rationale*.

A primeira delas é a Navegação, que permite que os envolvidos no projeto dos artefatos explorem as informações transitando de um nó para outro a partir das conexões existentes entre eles.

A segunda abordagem é a Pesquisa Automática, que durante a realização de atividades relacionadas, um monitor é utilizado para verificar se essas atividades estão consistentes com as regras, critérios e restrições estabelecidas, avisando automaticamente as diferenças e contradições.

A terceira e última abordagem é a Recuperação baseada em Consulta, que permite realizar questionamentos que podem ser respondidos explorando-se diferentes opções encontradas na base de conhecimento.

No âmbito da Ferramenta de Suporte à Engenharia de Requisitos, foi implementada a abordagem de recuperação baseada em Navegação, pois de acordo com a criação e expansão dos modelos de representação, o usuário da ferramenta poderá encontrar a conexão existente em cada um dos nós existentes. Como trabalho futuro, é proposta a implementação das outras abordagens sugeridas.

2.4 Cenário e Tendências

Em grande parte das empresas de desenvolvimento de software, ainda é

comum deparar-se com sistemas que não possuem documentação e que passam pelos mesmos problemas diversas vezes, devido a diferentes fatores, desde a resistência de colaboradores para transferência do conhecimento com o receio de não serem mais necessários à organização, ou pela falta de compreensão da importância da criação de uma base de conhecimento compartilhada que auxilie na execução dos processos e reutilização de informação.

Em Burge (2008) é possível encontrar questionários realizados com profissionais da indústria e estudantes da área de Ciências da Computação a respeito da utilização de sistemas baseados em *Design Rationale*. Esse questionário ranqueou em quais atividades, segundo esses profissionais, a utilização do *Design Rationale* é mais relevante. Em primeiro lugar, foi escolhida a sua utilização para a fase de projeto do software, e em segundo lugar a utilização dessa técnica foi considerada interessante para a execução das atividades relacionadas a Engenharia de Requisitos.

No âmbito da aplicação da técnica de *Design Rationale* em Engenharia de Requisitos, encontram-se ferramentas que atuam em algumas atividades desse processo, tais como:

- **SEURAT:** é um plugin desenvolvido em Java para o ambiente de desenvolvimento Eclipse que tem o objetivo de apoiar a atividade de manutenção de software utilizando *Design Rationale*. O plugin foi desenvolvido segundo a notação DRL (*Decision Representation Language*) (BURGE e BROWN, 2008);
- **DREAMER:** é uma ferramenta que utiliza o *Design Rationale* na definição da relação entre requisitos no domínio de Iteração Humano-Computador (IHC) tratando das modificações realizadas na interface dos sistemas e fazendo com que fatores como usabilidade e ergonomia de sistemas sejam avaliados. Ela utiliza a notação TEAM que estende a notação QOC para representação das informações em *Design Rationale* (MARTINIE et al., 2010);

- **KUABA:** é uma ferramenta que se destina a capturar e representar o *Design Rationale* durante, relacionando os requisitos de design de software, apoiando o reúso desses artefatos. Ela é baseada em uma ontologia que estende conceitos de diversas representações em *Design Rationale* (SANTOS e MEDEIROS, 2011).

Especificamente no âmbito da Engenharia de Requisitos que é o foco desde trabalho, para elucidar a contribuição dada pela FSER através da implementação do módulo relacionado à automatização do *Design Rationale*, foi realizada uma análise comparativa entre três ferramentas destinadas ao suporte das atividades do processo de Engenharia de Requisitos disponíveis no mercado. Foram estabelecidos cinco critérios de avaliação e três conceitos para cada critério. Na Tabela 1 são descritos os Critérios de Avaliação e seus respectivos Conceitos.

Tabela 1. Critérios de Avaliação e Conceitos para avaliação de Ferramentas

Critério de Avaliação	**Descrição**	**Conceitos do Critério**	**Descrição**
Levantamento e Análise	Apoio às atividades de identificação e modelagem de requisitos	Não	Não possui nenhuma funcionalidade relacionada
		Sim	A funcionalidade existente suporta totalmente as atividades relacionadas
		Parcial	A funcionalidade existente suporta parcialmente as atividades relacionadas
Especificação	Apoio às atividades de classificação, organização e documentação de requisitos	Não	Não possui nenhuma funcionalidade relacionada
		Sim	A funcionalidade existente suporta totalmente as atividades relacionadas
		Parcial	A funcionalidade existente suporta parcialmente as atividades relacionadas

Critério de Avaliação	Descrição	Conceitos do Critério	Descrição
Validação	Apoio às atividades de avaliação da conformidade de requisitos com as necessidades identificadas	Não	Não possui nenhuma funcionalidade relacionada
		Sim	A funcionalidade existente suporta totalmente as atividades relacionadas
		Parcial	A funcionalidade existente não suporta totalmente as atividades relacionadas
Gerência	Apoio às atividades de acompanhamento de fontes, mudanças e evoluções de requisitos	Não	Não possui nenhuma funcionalidade relacionada
		Sim	A funcionalidade existente suporta totalmente as atividades relacionadas
		Parcial	A funcionalidade existente não suporta totalmente as atividades relacionadas
Gestão do Conhecimento	Apoio às atividades de armazenamento e compartilhamento de informações para apoio às decisões	Não	Não possui nenhuma funcionalidade relacionada
		Sim	A funcionalidade existente suporta totalmente as atividades relacionadas
		Parcial	A funcionalidade existente não suporta totalmente as atividades relacionadas

Tendo estabelecido os critérios e conceitos de avaliação, três ferramentas foram avaliadas: OSRMT (OSRMT, 2013)., RequisitePro (IBM, 2013) e CaseComplete (CaseComplete, 2013). Em trabalhos correlatos (FUZII et al., 2009; ITO et al., 2011; SIMONATO et al., 2012; MIRANDA, 2013) foram feitas avaliações com tais ferramentas, mostrando a evolução do desenvolvimento da FSER.

A ferramenta OSRMT é uma ferramenta de código fonte aberto e tem como principal finalidade auxiliar a gerência dos requisitos (OSRMT, 2013). A ferramenta RequisitePro foi desenvolvida pela IBM e gerencia requisitos e casos de usos identificados para o desenvolvimento de software (IBM, 2013). A ferramenta

CaseComplete permite a gerência de casos de uso a partir dos requisitos desenvolvidos (CaseComplete, 2013).

Na Tabela 2 é mostrado o resultado da avaliação das ferramentas, incluindo também a análise da FSER, segundo os Critérios de Avaliação e os Conceitos estabelecidos.

Tabela 2. Avaliação de ferramentas que automatizam atividades de Engenharia de Software

Ferramenta	Levantamento e Análise	Especificação	Validação	Gerência	Gestão do Conhecimento
OSRMT	Não	Parcial	Parcial	Parcial	Não
RequisitePro	Não	Parcial	Sim	Parcial	Não
CaseComplete	Não	Parcial	Sim	Parcial	Não
FSER	Sim	Sim	Sim	Parcial	Não

Diante deste cenário, observa-se que a automatização do *Design Rationale* como forma de viabilizar o seu uso no contexto de uma ferramenta de apoio à efetivação das atividades relacionadas à Engenharia de Requisitos, contribuirá para a melhoria de todo o ciclo de desenvolvimento do software, fornecendo às atividades ligadas à Gerência de Requisitos uma fonte de registro de conhecimento a ser difundida e reutilizada no projeto de um software.

2.5 Considerações Finais

Esse capítulo teve como objetivo abordar uma breve revisão bibliográfica a respeito da Gerência dos Requistos, definindo o processo de Engenharia de Requisitos e as habilidades necessárias para se gerenciar requisitos de um software. Foi apresentada a técnica de *Design Rationale*, suas definições, abordagens e mecanismos de captura, representação e recuperação, além das propostas atuais que aplicam essa técnica e que se referem à gestão do conhecimento no âmbito da Engenharia de Requisitos.

Assim, a proposta desse estudo foi demonstrar a contribuição decorrente da melhoria da gerência dos requisitos de software, por meio da utilização integrada da técnica de *Design Rationale* ao processo de Engenharia de Requisitos, com o

propósito de auxiliar a disseminação do conhecimento e apoio à tomada de decisões aos gerentes e equipes envolvidas em projetos de desenvolvimento de software. Para tanto, foi desenvolvido um módulo de Gerência de Requisitos integrado à FSER utilizando a notação de Modelo de Design Reflexivo para representar o conhecimento em *Design Rationale*.

Capítulo 3 – Expansão da Ferramenta de Suporte à Elicitação de Requisitos

3.1 Considerações Iniciais

A Ferramenta de Suporte à Elicitação de Requisitos vem sendo desenvolvida por membros do Grupo de Estudos e Pesquisas em Engenharia de Software (GEPES) na Universidade Estadual Paulista “Júlio de Mesquita Filho”, Instituto de Biociências, Letras e Ciências Exatas de São José do Rio Preto desde 2008, com o intuito de fornecer um mecanismo de apoio para elicitação de requisitos de projetos de desenvolvimento de software, facilitando a interação e comunicação entre analistas e clientes. Desde sua concepção até a atualidade, diversas modificações foram acrescentadas à ferramenta a fim de aumentar sua abrangência no que se diz respeito à Engenharia de Requisitos e melhorar sua interface (FUZII, 2009; ITO et al., 2011; SIMONATO et al., 2012; ZITELLI, 2012). Em Miranda (2013) a ferramenta sofreu uma modificação com relação à Gerência de Requisitos, onde foram implementados mecanismos de rastreabilidade (Matrizes de Rastreabilidade) através da geração de relatórios que permitem o acompanhamento da evolução dos requisitos.

Assim, na Seção 3.2 são apresentadas as funções atualmente disponíveis na Ferramenta de Suporte à Elicitação de Requisitos – FSER. Na Seção 3.3 é descrita a expansão da FSER, detalhando as novas funções incorporadas, visando a modelagem dos requisitos, e a integração da técnica de *Design Rationale*. A partir daí, a ferramenta será renomeada como Ferramenta de Suporte à Engenharia de

Requisitos, e na Seção 3.4 são apresentados os materiais e métodos utilizados para essa expansão, bem como os resultados. Finalmente, na Seção 3.5 são apresentadas as considerações finais.

3.2 Funções da Ferramenta de Suporte à Elicitação de Requisitos

Em sua versão atual, a Ferramenta de Suporte à Elicitação de Requisitos – FSER possui como funções principais a elicitação, validação, especificação, documentação e rastreabilidade dos requisitos, as quais são divididas por módulos e três níveis de usuário, a saber:

- O usuário Administrador é aquele que fornece as informações relacionadas à maior parte dos cadastros do sistema, tendo acesso total (inserção, alteração, exclusão e pesquisa) aos cadastros de áreas (Redes, Desenvolvimento), analistas e projetos. Esse usuário também possui acesso aos relatórios gerenciais ligados aos requisitos elicitados;
- O usuário Analista possui acesso total às principais funções relacionadas aos requisitos (especificação, validação, documento de requisitos e rastreabilidade), às técnicas de elicitação (avaliação de documentos, *brainstorming*, entrevista e questionário) e ao cadastro de clientes;
- O usuário Cliente possui acesso parcial às funções relacionadas aos requisitos e técnicas de elicitação. Os Clientes e Analistas vinculados a um projeto podem trocar mensagens uns com os outros.

Na Figura 13 é apresentado o Diagrama de Caso de Uso da FSER e, na sequência, são resumidamente descritas as funções contempladas em cada caso de uso (FUZII, 2008; ITO, 2009; SIMONATO, 2011; MIRANDA, 2013).



Figura 13. Diagramas de Casos de Uso da FSER.

a) **Cadastrar Áreas:** são registradas as áreas vinculadas aos analistas, que definem o domínio de atuação de cada analista envolvido no desenvolvimento do projeto, tais como Suporte, Desenvolvimento, Administração de Banco de Dados, entre outras;

b) **Cadastrar Analistas:** são registrados dados básicos referentes aos analistas, como nome, email, telefone, nome de usuário e senha, assim como são selecionados os projetos em que cada analista irá atuar e o nível de responsabilidade nesses projetos

(os analistas podem ser responsáveis ou não por um projeto) e a área em que atuam;

c) **Cadastrar Projetos:** são registradas informações sobre os projetos que serão desenvolvidos com o auxílio da ferramenta, tais como nome do projeto, data inicial prevista, data final prevista, data inicial real e data final real;

d) **Gerar Relatórios Gerenciais:** são disponibilizados diferentes relatórios, cada um com características bem definidas, a saber **Relatório Requisito X Analista**, sendo apresentada a porcentagem total de requisitos reprovados, ou seja requisitos que não foram aprovados pelos clientes, de acordo com os respectivos analistas que os especificaram; **Relatório Requisito X Projeto**, sendo apresentada a porcentagem total de requisitos reprovados para um determinado projeto; **Relatório Requisito X Critério**, sendo apresentada a porcentagem de requisitos reprovados em um critério específico em diferentes projetos; **Relatório Analista X Projeto**, sendo apresentada a porcentagem total de requisitos reprovados nos projetos que um determinado analista participou; **Relatório Projeto X Critério**, sendo apresentada a quantidade de requisitos reprovados em um projeto específico para cada critério analisado;

e) **Cadastrar Clientes:** são registrados conteúdos relativos à identificação dos clientes como nome, telefone, email, nome de usuário e senha, assim como o vínculo de cada cliente com os projetos;

f) **Realizar Entrevistas:** são elaboradas perguntas, em um formato de bate-papo *online*, simulando uma entrevista de um analista com um cliente;

g) **Realizar *Brainstormings*:** é baseada na discussão em grupo entre analistas e clientes através de um chat online, diferentemente da técnica de elicitação de entrevista que pode ser realizada apenas entre dois participantes;

h) **Realizar Questionários:** são elaboradas, pelo analista, questões objetivas com alternativas para serem respondidas pelos clientes;

i) **Realizar Avaliação de Documentos:** fornece documentação referente a processos e fluxos da organização para auxiliar o trabalho dos analistas;

j) **Especificar Requisitos:** é apresentado um formulário aos analistas a ser preenchido com os requisitos elicitados;

k) **Validar Requisitos:** são consultadas, pelos clientes, as informações referentes aos requisitos cadastrados pelos analistas para, na sequência, aprovarem ou reprovarem os requisitos elicitados;

l) **Gerar ERS:** são preenchidas as informações referentes ao Documento de Especificação dos Requisitos de Software de acordo com o padrão IEEE 830;

m) **Trocar Mensagens:** possibilita a manutenção de uma boa comunicação entre usuários por meio da troca de mensagens internas entre analistas e clientes vinculados a um determinado projeto;

n) **Rastrear Requisitos:** possibilita o acompanhamento da evolução dos requisitos através de Matrizes de Rastreabilidade e relatórios que mostram a ligação entre os requisitos e os *stakeholders*, além do histórico das modificações realizadas sobre os requisitos;

o) **Gerar Log:** possibilita, através do registro de todas as ações efetuadas no requisito, gerar um histórico com informações sobre qual o usuário fez a modificação no requisito, a ação tomada (inclusão, alteração, remoção ou reprovação) e detalhes sobre esta ação.

3.3 Expansão da FSER

Da análise da FSER, observou-se a carência de recursos para apoiar, não

apenas a disseminação do conhecimento adquirido com o desenvolvimento de projetos, por meio da integração da técnica de *Design Rationale*, mas também a modelagem desses requisitos.

Com relação à modelagem, com a utilização de diversos mecanismos e padrões existentes, o objetivo de sua utilização é contribuir para a modelagem da estrutura, comportamento, arquitetura, estrutura de dados e processos de negócio de um software (LEFFINGWELL e WIDRIG, 2003). A Unified Modeling Language (UML) é uma linguagem de modelagem gráfica padrão para elaboração da estrutura de projetos de software orientado a objetos que fornece diversos diagramas e modelos a fim de alcançar tais objetivos (LEFFINGWELL e WIDRIG, 2003).

Dentre os diversos diagramas da UML, os Diagramas de Casos de Uso têm a utilidade de entrada para vários artefatos subsequentes do desenvolvimento de software (LARMAN, 2007) independentemente do paradigma a ser seguido durante seu desenvolvimento. Esses diagramas fazem com que os requisitos fiquem acessíveis e compreensíveis, por apresentarem uma visão externa sobre como esses elementos podem ser utilizados no contexto do software (SANTOS, 2011).

Sendo assim, observou-se que a FSER possui uma função específica de cadastro de Casos de Uso ao gerar o Documento de Especificação de Requisitos de Software, sendo que apenas os usuários do tipo Analista possuem acesso à manipulação dessas informações. Diante disso, a criação dos Diagramas de Casos de Uso torna-se uma segunda opção para a visualização das informações registradas.

Com a implementação dessa nova função na FSER, o cadastro das narrativas de Casos de Uso torna-se mais fácil, pois as informações registradas nos Diagramas de Casos de Uso são reutilizadas para especificar as narrativas de Casos de Uso, na geração do Documento de Especificação de Requisitos. Outra contribuição dessa função é fazer com que os Analistas consigam ter uma melhor compreensão do cenário em que o requisito está incorporado.

Ao selecionar a opção de criação de Diagramas de Casos de Uso, o Analista tem a opção de clicar e arrastar sobre os elementos do diagrama, assim como removê-los. É possível também realizar a alteração desse diagrama, após ter sido salvo.

Finalmente, com relação à criação de um módulo voltado para a utilização da técnica de *Design Rationale* durante a definição dos requisitos do software, o

objetivo é utilizar Modelos de Representações para estruturar o conhecimento gerado com o emprego da técnica.

Para este trabalho, o modelo de representação empregado é o Modelo de Design Reflexivo, pois além de propor uma estrutura advinda dos conceitos mais importantes de outros modelos de representação de *Design Rationale*, utiliza uma estrutura gráfica simples para formalizar as informações utilizadas para a tomada de decisões.

Assim, na Figura 14 é apresentado o Diagrama de Caso de Uso da expansão da FSER, e na seqüência, são descritos os casos de uso adicionados à ferramenta.

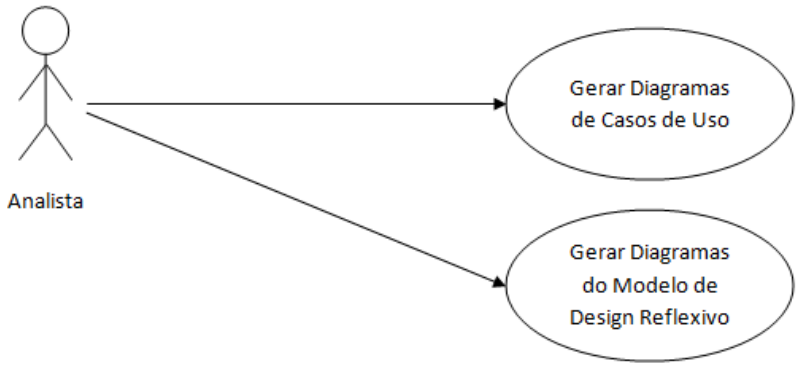


Figura 14. Diagrama de Casos de Uso da expansão da FSER.

a) **Gerar Diagramas de Casos de Uso:** com essa nova função, é possível especificar e armazenar casos de uso definindo os atores, os casos de uso e as relações existentes entre eles. Além disso, é possível acompanhar as versões dos diagramas, através do histórico de modificações dos mesmos;

b) **Gerar Diagramas do Modelo de Design Reflexivo:** nesta nova função é possível armazenar e recuperar informações relacionadas às tomadas de decisões e discussões realizadas à respeito do gerenciamento dos requisitos do projeto de desenvolvimento de software, segundo a técnica de *Design Rationale* do Modelo de Design Reflexivo. Também é possível acompanhar as versões desses diagramas, através de seus históricos de modificações.

3.4 Ferramenta de Suporte à Engenharia de Requisitos

Para a expansão da ferramenta, o padrão arquitetural de software estabelecido para o desenvolvimento da ferramenta é denominado MVC (Model-View-Controller). A arquitetura orientada a objeto MVC foi criada originalmente com a linguagem Smalltalk-80 como uma metodologia para separar a interface de apresentação ao usuário dos dados subjacentes. A proposta dessa metodologia é decompor o sistema como um todo em três subsistemas ou módulos. Cada módulo dessa arquitetura tem sua própria responsabilidade e as conexões entre esses módulos ocorrem da seguinte maneira: a camada de Controle coleta as entradas e faz as devidas validações, enviando essas entradas para o devido módulo na camada Modelo que é responsável pela persistência dos dados. Nessa iteração, a camada de Visão solicita as informações necessárias para criar uma representação de saída (QIAN et al., 2007).

A linguagem de programação utilizada no desenvolvimento da ferramenta foi a linguagem Java, principalmente as tecnologias para desenvolvimento Web JSP e Servlets (DEITEL, 2005), além da utilização da biblioteca gráfica mxGraph para o desenvolvimento dos diagramas. A linguagem HTML foi utilizada para a elaboração da interface do usuário, juntamente com a linguagem de estilos CSS e a biblioteca Javascript JQuery.

O sistema gerenciador de banco de dados escolhido para o armazenamento das informações da ferramenta foi o MySQL. O container de servelets, ou seja, o servidor web utilizado na ferramenta é o Apache Tomcat, da Apache Foundation.

Assim, nas subseções a seguir são apresentados os resultados obtidos com o desenvolvimento e integração das novas funções à FSER.

3.4.1 Reestruturação da Interface

Com o desenvolvimento e evolução da ferramenta algumas deficiências foram encontradas com relação à falta de estruturação de seu *layout*, embora a

FSER já tenha passado anteriormente por um processo de Reengenharia (ZITELLI, 2012).

Com isso, uma nova interface foi desenvolvida para a FSER. Essa nova proposta de *layout* visa estruturar e organizar as informações fornecidas pela ferramenta. Grande parte das modificações foram realizadas criando estilos em cascata para páginas HTML (CSS) padronizados e utilizando a biblioteca JQuery para melhorar a experiência do usuário com a ferramenta, minimizando o excesso de carregamento de páginas desnecessariamente. As Figuras 15 e 16 são exemplos dessa mudança sendo que são representadas, respectivamente, a interface antiga da FSER juntamente com a nova interface de acesso à ferramenta e a interface padrão do sistema.



Figura 15. Interface antiga da FSER e nova Página Inicial da FSER.



Figura 16. Nova interface padrão da FSER.

3.4.2 Diagramas de Casos de Uso

Os Diagramas de Casos de Uso foram integrados à FSER considerando que a ferramenta realiza o registro dos casos de uso do projeto para a geração do Documento de Especificação dos Requisitos do Software.

Com essa nova funcionalidade, a geração do Diagrama de Casos de Uso poderá ser feita a partir da atividade de elicitação dos requisitos. Observa-se que os níveis de usuário que possuem acesso à geração e alteração dos Diagramas de Casos de Uso são Analista e Cliente.

Para visualizar os diagramas, é necessário selecionar um projeto cadastrado no sistema, clicando sobre a opção do menu principal “Requisitos”. Relacionado a cada projeto, encontra-se a opção “Diagrama de Casos de Uso”. Na Figura 17 é mostrada essa situação.

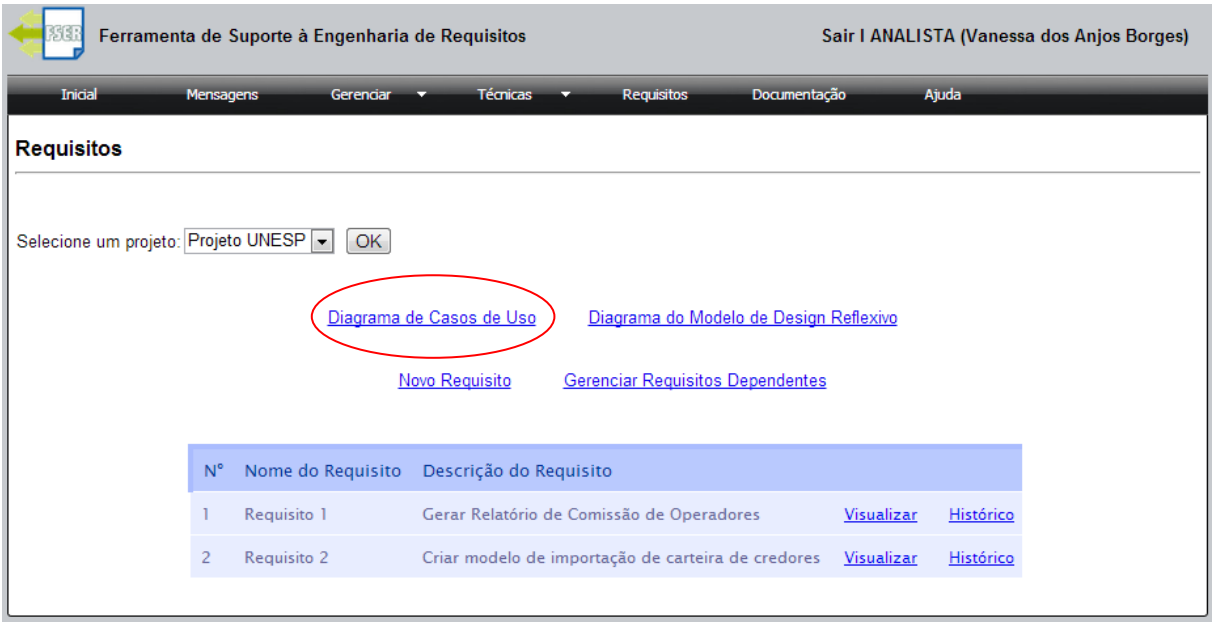


Figura 17. Opção para geração de Diagramas de Casos de Uso.

Ao clicar sobre a opção “Diagrama de Casos de Uso”, caso nenhum diagrama tenha sido criado, é informado ao usuário que não há nenhum registro de diagrama criado e a interface para criação do diagrama é apresentada, conforme mostrado na Figura 18, caso contrário, são informadas as versões existentes do diagrama, onde é possível selecionar a visualização de uma das versões ou alterar a última versão disponível, conforme demonstrado na Figura 19. Para o desenvolvimento dessa funcionalidade foi utilizada a biblioteca mxGraph.

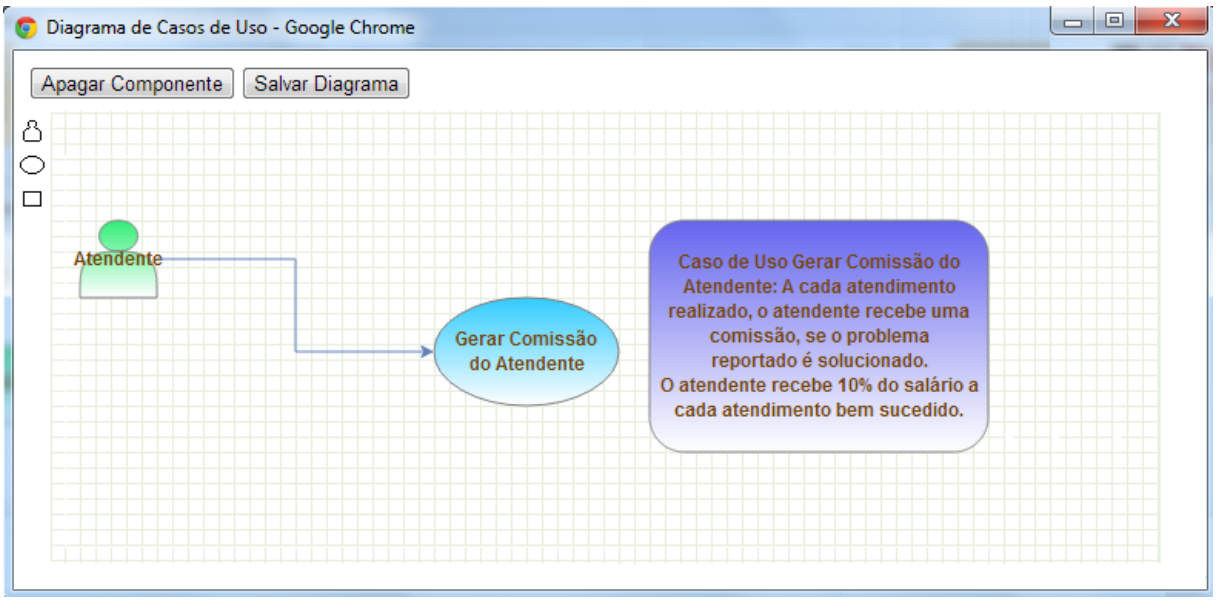


Figura 18. Funcionalidade de geração de Diagramas de Casos de Uso na FSER.



Figura 19. Consulta às versões do Diagramas de Casos de Uso na FSER.

Assim, a Figura 20 demonstra a relação entre o Diagrama de Casos de Uso e as narrativas de Casos de Uso. Ao descrever as narrativas para a geração do Documento de Especificação de Requisitos de Software, são sugeridos Casos de Uso segundo aqueles que foram especificados no Diagrama de Casos de Uso do projeto.

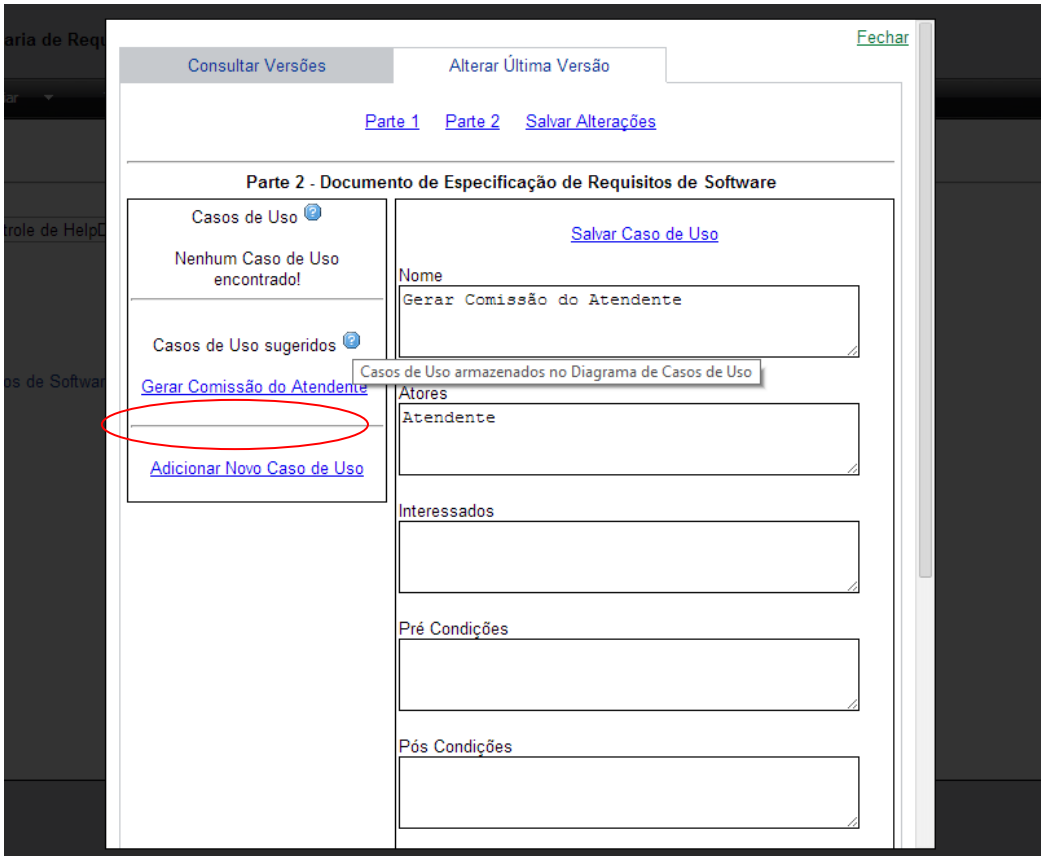


Figura 20. Relação entre Diagramas de Casos de Uso e narrativas de Casos de Uso.

3.4.3 Diagrama do Modelo de Design Reflexivo

Dando continuidade à implementação, foi desenvolvida a funcionalidade ligada à geração de Diagramas do Modelo de Design Reflexivo. Essa funcionalidade tem por principal finalidade auxiliar a tomada de decisão com base em informações de discussões realizadas e estruturadas de acordo com a representação de *Design Rationale* do Modelo de Design Reflexivo.

Assim como os Diagramas de Casos de Uso, é necessário selecionar o projeto que deseja gerenciar e então é apresentada a opção de geração de Diagramas do Modelo de Design Reflexivo, como mostrado na Figura 21.



Figura 21. Opção para geração de Diagramas do Modelo de Design Reflexivo na FSER.

Ao clicar sobre a opção “Diagrama do Modelo de Design Reflexivo”, caso nenhum Diagrama de Modelo de Design Reflexivo tenha sido criado, é informado ao usuário que não há nenhum registro de diagrama criado e a interface para criação do diagrama é apresentada, conforme exemplificado na Figura 22, caso contrário, são informadas as versões existentes do diagrama, onde é possível selecionar a visualização de uma das versões ou alterar a última versão disponível, conforme mostrado na Figura 23.

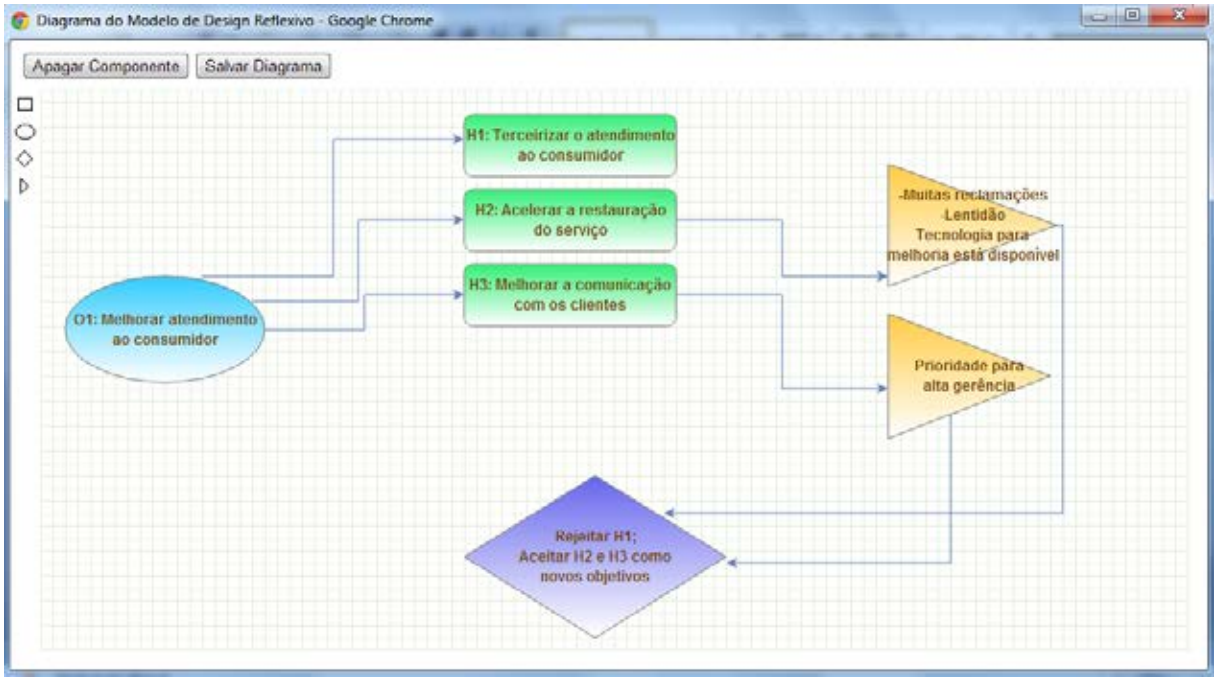


Figura 22. Funcionalidade de geração de Diagramas de Modelo de Design Reflexivo na FSER (Adaptado de Louridas e Loucopulos, 2000).



Figura 23. Consulta às versões do Diagramas do Modelo de Design Reflexivo na FSER

3.4 Considerações Finais

Neste capítulo, foram apresentadas as funções originais da FSER e o detalhamento da expansão desenvolvida.

Foi apresentada também a integração de duas novas funções à FSER: geração de Diagramas de Casos de Uso e Diagramas do Modelo de Design Reflexivo.

Assim, com a maior abrangência da ferramenta, é possível afirmar que diversos aspectos da Gerência dos Requisitos podem ser tratados com as novas funções implementadas, como facilitar o desenvolvimento dos requisitos, contribuir para melhoria do processo de análise e validação, além de criar uma base de conhecimento para auxiliar no desenvolvimento de habilidades de argumentação e documentação de equipes envolvidas em projetos de software.

No capítulo a seguir, é apresentada uma avaliação da ferramenta realizada por alunos do curso de graduação e pós-graduação da UNESP e profissionais da área de informática, a fim de confirmar a eficácia proposta pela ferramenta.

Capítulo 4 – Avaliação da Ferramenta de Suporte à Engenharia de Requisitos

4.1 Considerações Iniciais

A fim de observar a eficácia da Ferramenta de Suporte à Engenharia de Requisitos no processo de Gerência de Requisitos, foi realizado um experimento controlado, baseado no Processo de Experimentação proposto por Wohlin et al. (2012).

A elaboração e condução desse experimento contou com a colaboração do aluno de graduação do Bacharelado em Ciência da Computação da Universidade Estadual Paulista “Júlio de Mesquita Filha” UNESP, campus de São José do Rio Preto, James Willian Pontes Miranda, cujo trabalho final de graduação recentemente concluído está relacionado com a FSER, tendo sido implementadas funções de rastreabilidade de requisitos (MIRANDA, 2013).

Assim, na Seção 4.2 são apresentadas a definição e o planejamento do experimento. Na Seção 4.3 é apresentado detalhadamente o cronograma de execução do experimento e na Seção 4.4 é descrita a apresentação e análise de resultados obtidos. Na Seção 4.5 é apresentada a avaliação do treinamento pelos participantes e na Seção 4.6 são descritas as considerações finais do capítulo.

4.2 Definição e Planejamento do Experimento

Um experimento controlado tem por objetivo verificar e validar teorias, além de corrigir e aprimorar técnicas que são utilizadas por desenvolvedores de software (Basili et al., 1999).

As duas primeiras etapas de realização de um experimento controlado em Engenharia de Software são a definição e o planejamento.

Na definição do experimento é estabelecida a visão geral do experimento, incluindo o objeto de estudo, objetivo, foco de qualidade, perspectiva e contexto. No planejamento do experimento são definidas as hipóteses, variáveis e indivíduos, além de realizar o projeto do experimento (WOHLIN et al., 2012).

4.2.1 Definição do Experimento

O objetivo da realização desse experimento é avaliar a eficácia na utilização da Ferramenta de Suporte à Engenharia de Requisitos para apoiar a atividade de Gerência de Requisitos de software utilizando a técnica de *Design Rationale*. Para tanto, foram comparados os resultados das experiências dos participantes na gestão dos conhecimentos relacionados à Gerência de Requisitos de software com e sem a utilização da ferramenta, segundo a complexidade, eficiência e motivação no armazenamento, organização e recuperação das informações relativas à Gerência de Requisitos de software utilizando a técnica de *Design Rationale*.

Logo, a perspectiva do experimento considerou os resultados observados de acordo com a visão dos participantes do experimento. O experimento foi realizado nos laboratórios didáticos de informática do Departamento de Ciências de Computação e Estatística do Instituto de Biociências, Letras e Ciências em São José do Rio Preto, e do Departamento de Matemática e Computação da Faculdade de Ciências e Tecnologia em Presidente Prudente, ambos campus da Universidade Estadual Paulista “Júlio de Mesquita Filho” (UNESP).

4.2.2 Planejamento do Experimento

O experimento foi realizado com a participação de alunos do curso de graduação do Bacharelado em Ciência da Computação, da UNESP de São José do Rio Preto e Presidente Prudente e alunos do Programa Interunidades de Pós-Graduação em Ciência da Computação da UNESP, além de contar com a participação de profissionais da área de informática, tanto da cidade de São José do Rio Preto quanto de Presidente Prudente.

As hipóteses de um experimento têm como propósito fornecer uma análise estatística ao experimento por meio dos dados coletados (WOHLIN et al., 2012).

Para a efetiva avaliação da FSER quanto à utilização da função relacionada à técnica *Design Rationale* para apoiar a Gerência de Requisitos de software, foram elaboradas três hipóteses validadas pelos participantes. Nas Tabelas 3, 4 e 5 são descritas cada uma dessas hipóteses e as respectivas métricas estabelecidas, as quais são posteriormente detalhadas nas Tabelas 6, 7 e 8.

Para cada uma das hipóteses também foram consideradas duas abordagens. A abordagem *ad hoc* é tida como a realização de uma atividade de maneira informal e sem padronização estabelecida. A abordagem com a utilização da funcionalidade relacionada ao *Design Rationale* implementada na FSER é apoiada pelo suporte fornecido pela ferramenta.

Tabela 3. Hipóteses Nula H_0, Hipótese Alternativa H_{a0} e sua respectiva métrica

Complexidade no armazenamento e recuperação de informações relativas à Gerência de Requisitos	
Hipótese Nula (H_0)	Não há diferença de complexidade no armazenamento e recuperação de informações relativas à Gerência de Requisitos utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER e a abordagem *ad hoc*
Hipótese Alternativa (H_{a0})	Há diferença de complexidade no armazenamento e recuperação de informações relativas à Gerência de Requisitos utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER e a abordagem *ad hoc*
Métrica	Grau de Complexidade para armazenamento e recuperação de informações relativas à Gerência de Requisitos em cada abordagem

Tabela 4. Hipóteses Nula H_1, Hipótese Alternativa H_{a1} e sua respectiva métrica

Eficiência no armazenamento e recuperação de informações relativas à Gerência de Requisitos	
Hipótese Nula (H_1)	Não há diferença de eficiência no armazenamento e recuperação de informações relativas à Gerência de Requisitos utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER e a abordagem *ad hoc*
Hipótese Alternativa (H_{a1})	Há diferença de eficiência no armazenamento e recuperação de informações relativas à Gerência de Requisitos utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER e a abordagem *ad hoc*
Métrica	Grau de eficiência para armazenamento e recuperação de informações em cada abordagem

Tabela 5. Hipóteses Nula H_2, Hipótese Alternativa H_{a2} e sua respectiva métrica

Motivação no armazenamento e recuperação de informações relativas à Gerência de Requisitos	
Hipótese Nula (H_2)	Não há diferença quanto à motivação para armazenamento e recuperação de informações relativas à Gerência de Requisitos utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER e a abordagem *ad hoc*
Hipótese Alternativa (H_{a2})	Há diferença quanto à motivação para armazenamento e recuperação de informações relativas à Gerência de Requisitos utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER e a abordagem *ad hoc*
Métrica	Grau de motivação para armazenamento e recuperação de informações em cada abordagem

Tabela 6. Descrição da métrica Complexidade para a realização do experimento

Grau de Complexidade	**Descrição**
Alta	É exigido um alto grau de raciocínio para organizar e recuperar as informações relativas à Gerência de Requisitos
Média	É exigido um grau mediano de raciocínio para organizar e recuperar as informações relativas à Gerência de Requisitos
Baixa	É exigido um baixo grau de raciocínio para organizar e recuperar as informações relativas à Gerência de Requisitos

Tabela 7. Descrição da métrica Eficiência para a realização do experimento

Grau de Eficiência	Descrição
Eficiente	Não existem dificuldades para manipulação das informações relativas à Gerência de Requisitos utilizando o modelo proposto
Parcialmente Eficiente	A dificuldade para manipular as informações relativas à Gerência de Requisitos utilizando o modelo proposto é média
Não Eficiente	Existe muita dificuldade para manipulação das informações relativas à Gerência de Requisitos utilizando o modelo proposto

Tabela 8. Descrição da métrica Motivação para a realização do experimento

Grau de Motivação	Descrição
Existente	O participante se sente estimulado a utilizar a abordagem
Inexistente	O participante não se sente estimulado a utilizar a abordagem

As variáveis independentes de um experimento são os fatores essenciais para a entrada do processo de experimentação. As variáveis independentes consideradas para este experimento são Documentos de Requisitos A e B (vide Apêndice A), Atas de Reuniões (vide Apêndice B) e a nova funcionalidade integrada à FSER que implementa a técnica de *Design Rationale*.

As variáveis dependentes estão na saída do processo experimental e, neste experimento, são o grau de complexidade, eficiência e motivação no armazenamento e recuperação de informações relativas à Gerência de Requisitos.

Para a formalização do experimento, o mesmo foi realizado no formato de um curso de extensão, sob o título “Gerência de Requisitos: Conceitos e Técnicas”, que aconteceu no período de 11 a 14 de novembro de 2013 das 19h às 22h, na Central de Laboratórios II, laboratório de robótica (experimento) e na sala 4N do Departamento de Ciências de Computação e Estatística (treinamento); no campus da UNESP de São José do Rio Preto, e no período de 19 a 22 de novembro de 2013 das 19h às 22h, na Central de Laboratórios Didáticos de Computação, laboratório 4b e 5b (treinamento e experimento), no campus da UNESP de Presidente Prudente.

Foram preenchidas 8 vagas em São José do Rio Preto e 12 vagas em Presidente Prudente, por alunos de graduação do curso de Ciência da Computação (São José do Rio Preto e Presidente Prudente), alunos do Programa Interunidades

de Pós-Graduação em Ciência da Computação da UNESP, (São José do Rio Preto e Presidente Prudente) e profissionais da área de informática (8 profissionais), totalizando 20 participantes. Considerando a diversidade de níveis de conhecimento, o propósito do curso é a realização de um treinamento a fim de apresentar os conceitos relacionados às técnicas e mecanismos aplicados à Gerência de Requisitos.

Considerando que o experimento foi aplicado em duas cidades diferentes, os participantes da cidade de São José do Rio Preto foram denominados componentes da Turma A e os participantes da cidade de Presidente Prudente como componentes da Turma B. Antes da realização do experimento, foram realizados treinamentos sobre Engenharia de Software, Engenharia de Requisitos e técnicas da Gerência de Requisitos, para que os participantes adquirissem o nível mínimo de conhecimento para a realização das atividades do experimento. Para os experimentos, os participantes de cada turma foram selecionados de maneira aleatória e divididos em duplas. Os participantes foram agrupados em duplas para simular a situação real do ambiente de desenvolvimento de software, onde os projetos de desenvolvimento são realizados em equipes.

Antes da realização do experimento, os participantes preencheram um Termo de Consentimento Livre e Esclarecido (vide Apêndice C), que confirma a participação na pesquisa realizada e um Questionário para Identificação de Participantes (vide Apêndice C) para traçar o perfil dos participantes da pesquisa. Para realizar a comparação das duas abordagens propostas pelo experimento (*abordagem ad hoc* e abordagem com a utilização da funcionalidade relacionada ao *Design Rationale* implementada na FSER), foram elaborados dois modelos de Documento de Requisitos de Software que descrevem as funcionalidades requeridas em um sistema segundo a visão de seus usuários (Documento A e B).

A primeira etapa da experimentação foi desenvolver, a partir dos documentos, uma extração e organização dos requisitos especificados pelos usuários de maneira informal e sem padronização nenhuma estabelecida (abordagem *ad hoc*). Cada grupo recebeu um dos documentos de requisitos aleatoriamente, ou seja, na Turma A duas duplas receberam o Documento A e duas duplas receberam o Documento B; na Turma B três duplas receberam o Documento A e três duplas receberam o Documento B. Dando continuidade à abordagem *ad hoc*, os participantes receberam Atas de Reuniões relacionadas aos documentos de

requisitos. Os participantes então realizaram o armazenamento das ideias que surgiram das discussões das atas de maneira livre. Após a realização desse processo, os participantes preencheram o Formulário de Avaliação da Abordagem *ad hoc* (vide Apêndice C).

A segunda etapa da experimentação foi a realização das mesmas atividades propostas segundo a abordagem *ad hoc*, porém utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER, ou seja, os mecanismos propostos pela FSER para a identificação de requisitos e a funcionalidade de *Design Rationale* para o armazenamento e recuperação das ideias que surgiram das discussões das atas.

Para tanto, as duplas trocaram os documentos de requisitos utilizados na abordagem *ad hoc*, ou seja, as duplas que utilizaram o Documento A durante a abordagem *ad hoc* utilizaram o Documento B para a abordagem com a utilização da funcionalidade relacionada ao *Design Rationale* implementada na FSER; as duplas que utilizaram o Documento B durante a abordagem ad hoc utilizaram o Documento A para a abordagem com a utilização da funcionalidade relacionada ao *Design Rationale* implementada na FSER. Após a realização desse processo, os participantes preencheram o Formulário de Avaliação da Abordagem com utilização da FSER.

Ao final do curso, os participantes preencheram o Formulário sobre o Treinamento (vide Apêndice C), onde informaram o grau de satisfação em relação ao conteúdo transmitido.

As ferramentas utilizadas durante o processo de experimentação foram as de edição de texto disponíveis nos laboratórios de informática da UNESP de São José do Rio Preto e Presidente Prudente (LibreOffice) e a Ferramenta de Suporte à Engenharia de Requisitos (FSER). Os materiais utilizados para a realização do experimento foram: Termo de Consentimento Livre e Esclarecido, Questionário para Identificação de Participantes, Documento de Requisitos de Software A e B, Atas de Reuniões, Formulário de Avaliação da Abordagem *ad hoc*, Formulário de Avaliação da Abordagem com utilização da FSER e Formulário sobre o Treinamento (vide materiais em Apêndice A, B e C).

Para assegurar que o experimento tenha resultados válidos, é fundamental definir as validades do experimento, evitando ameaças que invalidem

os resultados obtidos. Na Tabela 9 são mostradas as validades consideradas para esse experimento.

Tabela 9. Validades do experimento

Validade Interna
Todos os participantes (Turma A e Turma B) tiveram o mesmo treinamento, não havendo diferenças que poderiam comprometer os resultados do experimento
Validade Externa
Um treinamento completo sobre os temas abordados no experimento foi realizado entre os estudantes de graduação e pós-graduação, propiciando a generalização dos resultados
Validade de Construção
As hipóteses do experimento não foram apresentadas aos participantes para não influenciar os resultados
Validade de Conclusão
Todos os participantes realizaram as mesmas atividades, tanto na realização da abordagem ad hoc quanto na realização da abordagem utilizando a funcionalidade relacionada ao Design Rationale implementada na FSER

4.3 Execução do Experimento

O experimento foi realizado durante quatro dias consecutivos, nos quais os dois primeiros dias foram dedicados exclusivamente para a realização de treinamentos e atividades. Os dois últimos dias foram dedicados à realização das atividades ligadas ao experimento efetivamente. Na Tabela 10 são mostradas detalhadamente as atividades realizadas nos quatro dias do Curso de Extensão.

Tabela 10. Cronograma de Execução

Dia 1 – Treinamento (Engenharia de Software e Engenharia de Requisitos)	
19:00 hrs – 20:20 hrs	Assinatura do Termo de Consentimento, Caracterização dos Participantes e Treinamento sobre Conceitos Fundamentais em Engenharia de Software
	Coffee-Break
20:40 hrs – 22:00 hrs	Treinamento sobre conceitos Fundamentais em Engenharia de Requisitos e realização de atividades

Dia 2 – Treinamento (Gestão de Conhecimento e Gerência de Requisitos)	
19:00 hrs – 20:20 hrs	Treinamento sobre Conceitos Fundamentais em Gestão do Conhecimento e realização de atividades
	Coffee-Break
20:40 hrs – 22:00 hrs	Treinamento sobre Conceitos Fundamentais em Gerência de Requisitos

Dia 3 – Realização do Experimento (Parte 1)		
19:00 hrs – 20:20 hrs	Sorteio das duplas e execução do experimento segundo a abordagem *ad hoc*	
	Dupla A *Execução do Experimento utilizando como referência o Documento de Requisitos de Software A e Atas de Reuniões*	*Dupla B* *Execução do Experimento utilizando como referência o Documento de Requisitos de Software B e Atas de Reuniões*
	Preenchimento do Formulário de Avaliação da Abordagem *ad hoc*	
	Coffee-Break	
20:40 hrs – 22:00 hrs	Treinamento sobre Técnicas para a Gerência de Requisitos de Software	

Dia 4 – Realização do Experimento (Parte 2)		
19:00 hrs – 20:20 hrs	Apresentação da Ferramenta de Suporte à Engenharia de Requisitos	
	Coffee-Break	
20:40 hrs – 22:00 hrs	Execução do experimento segundo a abordagem com a utilização da funcionalidade relacionada ao Design Rationale implementado na FSER	
	Dupla A *Execução do Experimento utilizando como referência o Documento de Requisitos de Software B e Atas de Reuniões*	*Dupla B* *Execução do Experimento utilizando como referência o Documento de Requisitos de Software A e Atas de Reuniões*
	Preenchimento Formulário de Avaliação da Abordagem com utilização da FSER	
	Encerramento	

4.4 Apresentação e Análise de Resultados do Experimento

De acordo com a realização do experimento, os resultados obtidos pelo

processo experimental foram agrupados em Turma A – São José do Rio Preto e Turma B – Presidente Prudente.

Após a realização dos treinamentos, os participantes realizaram as atividades propostas na execução do experimento e preencheram os formulários relacionados à avaliação das abordagens propostas.

4.4.1 Turma A – São José do Rio Preto

De acordo com o preenchimento do Questionário para Identificação dos Participantes (Apêndice C) realizado pelos oito participantes da cidade de São José do Rio Preto, foi possível estabelecer um perfil desse grupo. Na Tabela 11 são apresentadas as informações levantadas das características de cada participante.

Tabela 11. Perfil dos Participantes da Turma A – São José do Rio Preto

Perfil dos Participantes da Turma A – São José do Rio Preto		
Sexo	**Número**	**Porcentagem**
Masculino	8	100%
Feminino	0	0%
Formação Acadêmica	**Número**	**Porcentagem**
Graduação	6	75%
Pós Graduação (Especialização - Completo)	2	25%
Experiência com Projetos de Software	**Número**	**Porcentagem**
Participou em ambiente acadêmico	1	12,5%
Participou em ambiente profissional	7	87,5%
Tempo de Experiência com Projetos de Software	**Número**	**Porcentagem**
0 ano	1	12,5%
2 anos	3	37,5%
5 anos	2	25%
9 anos	1	12,5%
10 anos	1	12,5%
Já realizou atividades relacionadas à Gerência de Requisitos?	**Número**	**Porcentagem**
Sim	5	62,5%
Não	3	37,5%

Conhece alguma técnica ou ferramenta que automatize a atividade de Gerência de Requisitos?	**Número**	**Porcentagem**
Sim	3	37,5%
Não	5	62,5%

Você sabe o que é *Design Rationale*?	**Número**	**Porcentagem**
Sim	0	0%
Não	8	100%

TOTAL DE PARTICIPANTES	**8**

Na Figura 24 é mostrado o resultado da avaliação dos participantes segundo as abordagens *ad hoc* e utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER, em relação ao grau de complexidade de armazenamento e recuperação de informações relacionadas à Gerência de Requisitos.

Segundo o gráfico, é possível identificar que ao utilizar a FSER, a realização do armazenamento e recuperação das informações relacionadas à Gerência de Requisitos de software torna-se menos complexa. Assim pode-se afirmar que a Hipótese Alternativa 0 (H_{a0}) é verdadeira, sendo que a utilização da FSER torna a atividade mais fácil de se realizar.

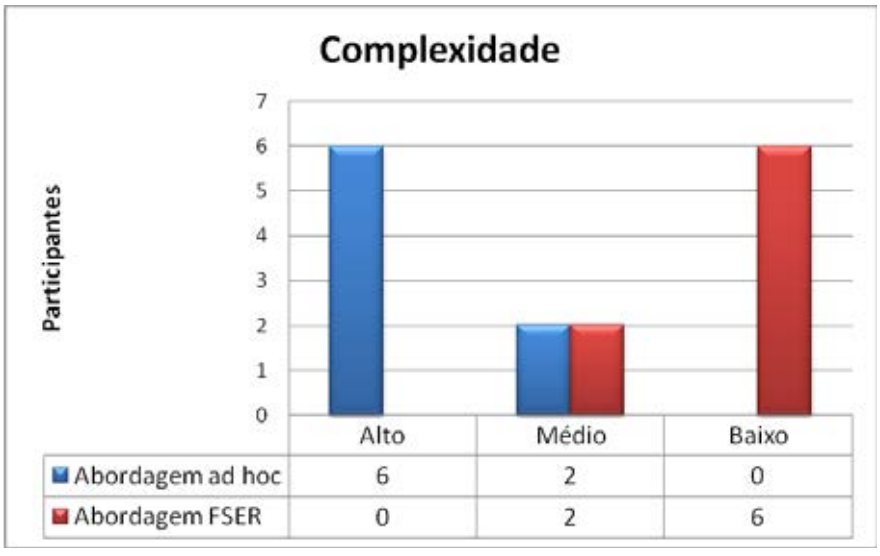


Figura 24. Avaliação da Complexidade das abordagens do experimento da Turma A

Na Figura 25 é mostrado o resultado da avaliação dos participantes segundo as abordagens *ad hoc* e utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER, em relação ao grau de eficiência de armazenamento e recuperação de informações relacionadas à Gerência de Requisitos.

Ao analisar a Figura 25, pode-se concluir que os participantes consideraram mais eficiente a realização das atividades tendo como suporte a FSER, e que 75% dos participantes consideraram fácil a realização das atividades. Assim pode-se afirmar que a Hipótese Alternativa 1 (H_{a1}) é verdadeira, considerando a abordagem com a utilização da FSER mais eficiente que a abordagem *ad hoc*.

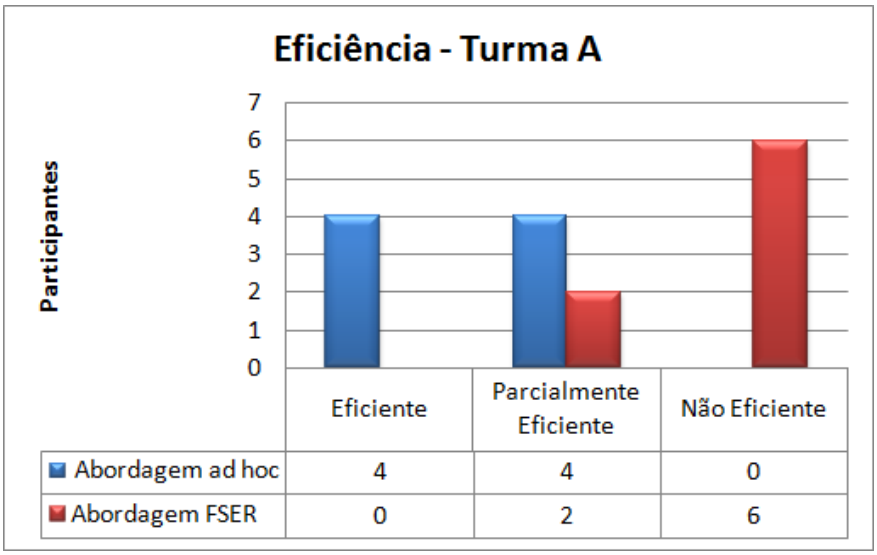


Figura 25. Avaliação da Eficiência das abordagens do experimento da Turma A

Na Figura 26 é mostrado o resultado da avaliação dos participantes segundo as abordagens *ad hoc* e utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER, em relação ao grau de motivação de armazenamento e recuperação de informações relacionadas à Gerência de Requisitos.

Considerando o resultado da avaliação da abordagem com a utilização da FSER, pode-se concluir que a Hipótese Alternativa 2 (H_{a2}) é verdadeira, pois todos os participantes sentiram-se motivados à utilizar a ferramenta como suporte à realização de atividades de Gerência de Requisitos, e 75% dos participantes não se sentiram motivados a realizar as atividades novamente segundo a abordagem *ad hoc*.



Figura 26. Avaliação da Motivação das abordagens do experimento da Turma A

4.4.2 Turma B – Presidente Prudente

De acordo com o preenchimento do Questionário para Identificação dos Participantes (Apêndice C) realizado pelos doze participantes da cidade de Presidente Prudente foi possível estabelecer um perfil desse grupo. Na Tabela 12 são apresentadas as informações levantadas das características de cada participante.

Tabela 12. Perfil dos Participantes da Turma B – Presidente Prudente

Perfil dos Participantes da Turma B – Presidente Prudente		
Sexo	**Número**	**Porcentagem**
Masculino	9	75%
Feminino	3	25%
Formação Acadêmica	**Número**	**Porcentagem**
Graduação	9	75%
Pós Graduação (Especialização - Completo)	1	8,3%
Pós Graduação (Mestrado - Cursando)	2	16,7%
Experiência com Projetos de Software	**Número**	**Porcentagem**
Nunca participou de projetos de software	1	8,3%
Participou em ambiente acadêmico	11	91,7%
Participou em ambiente profissional	5	41,7%

Tempo de Experiência com Projetos de Software	**Número**	**Porcentagem**
0 ano	1	8,3%
1 anos	5	41,7%
2 anos	1	8,3%
3 anos	3	25%
4 anos	1	8,3%
10 anos	1	8,3%

Já realizou atividades relacionadas à Gerência de Requisitos?	**Número**	**Porcentagem**
Sim	5	41,7%
Não	7	58,3%

Conhece alguma técnica ou ferramenta que automatize a atividade de Gerência de Requisitos?	**Número**	**Porcentagem**
Sim	4	33,3%

Conhece alguma técnica ou ferramenta que automatize a atividade de Gerência de Requisitos?	**Número**	**Porcentagem**
Não	8	66,7%

Você sabe o que é *Design Rationale*?	**Número**	**Porcentagem**
Sim	1	8,3%
Não	11	91,7%

TOTAL DE PARTICIPANTES	**12**

Na Figura 27 é mostrado o resultado da avaliação dos participantes segundo as abordagens *ad hoc* e utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER, em relação ao grau de complexidade de armazenamento e recuperação de informações relacionadas à Gerência de Requisitos.

Segundo os resultados obtidos na Turma B, a abordagem com utilização da FSER pode ser considerada de baixa complexidade, onde a maioria considerou a complexidade da abordagem baixa, podendo confirmar a veracidade da Hipótese Alternativa 0 (H_{a0}).

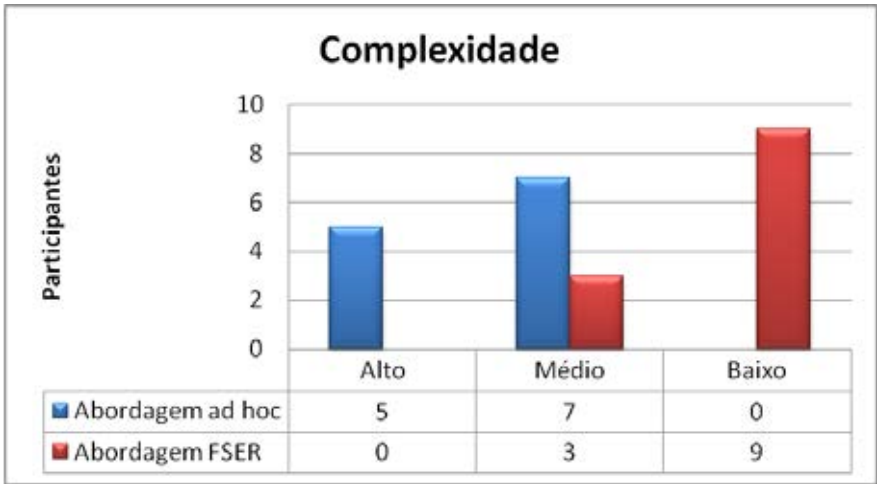


Figura 27. Avaliação da Complexidade das abordagens do experimento da Turma B

Na Figura 28 é mostrado o resultado da avaliação dos participantes segundo as abordagens *ad hoc* e utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER, em relação ao grau de eficiência de armazenamento e recuperação de informações relacionadas à Gerência de Requisitos.

Considerando os resultados obtidos, nenhum participante considerou a abordagem com utilização da FSER difícil, e que na abordagem ad hoc 58,3% (7 participantes) consideraram a abordagem difícil, pode-se concluir que a Hipótese Alternativa 1 (H_{a1}) é verdadeira, além de confirmar que a abordagem utilizando a FSER é mais eficiente que a abordagem *ad hoc*.

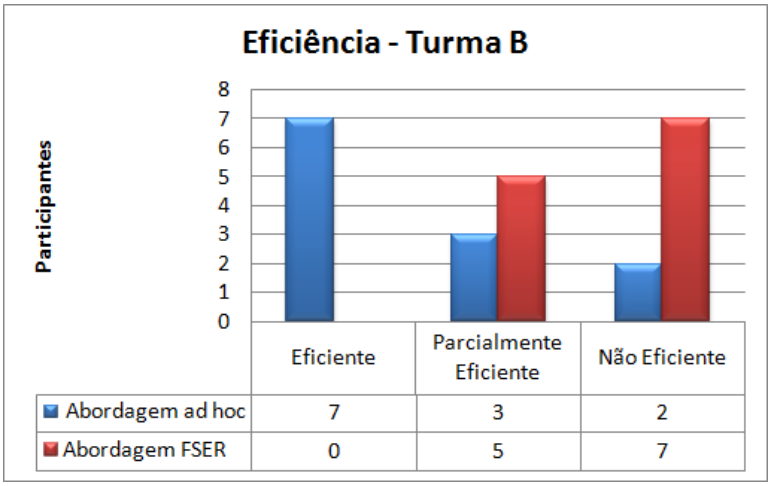


Figura 28. Avaliação da Eficiência das abordagens do experimento da Turma B

Na Figura 29 é mostrado o resultado da avaliação dos participantes segundo as abordagens *ad hoc* e utilizando a funcionalidade relacionada ao *Design Rationale* implementada na FSER, em relação ao grau de motivação de armazenamento e recuperação de informações relacionadas à Gerência de Requisitos.

Segundo todos os participantes da Turma B, existe motivação na utilização da abordagem com utilização da FSER, o que não pode ser constatado na abordagem *ad hoc*, onde apenas dois participantes sentiram-se motivados a voltar a utilizá-la. Assim, confirma-se a validade da Hipótese Alternativa 2 (H_{a2}), onde foi confirmada a motivação na realização das atividades segundo a abordagem com a utilização da FSER.



Figura 29. Avaliação da Motivação das abordagens do experimento da Turma B

4.5 Avaliação do Treinamento

Antes do encerramento do curso de extensão, os participantes informaram através do Formulário de Avaliação do curso (vide Apêndice C) seu grau de satisfação em relação ao conteúdo que foi abordado durante o treinamento. Na Figura 30 é apresentada uma avaliação geral com os principais aspectos avaliados pelos participantes. Nesta figura é apresentado o resultado da avaliação dos participantes da Turma A (São José do Rio Preto) e da Turma B (Presidente Prudente).

Em geral, o curso foi considerado ótimo por mais de 40% dos participantes. O treinamento sobre a FSER foi considerado bom por mais de 50% dos participantes. Sobre o treinamento da FSER, grande parte dos participantes sugeriu um treinamento mais amplo, pois ficaram interessados nas outras funções fornecidas pela ferramenta e que não foram apresentadas, pois não faziam parte do contexto do curso, que focou a gerência dos requisitos.

Além disso, foram feitas considerações para a melhoria do treinamento, como, por exemplo, a criação de um material apostilado para melhor guiar os participantes durante o curso. Os graus de satisfação são definidos como: Excelente (100% de satisfação), Ótimo (80% de satisfação), Bom (60% de satisfação), Regular (40% de satisfação) e Ruim (20% de satisfação).

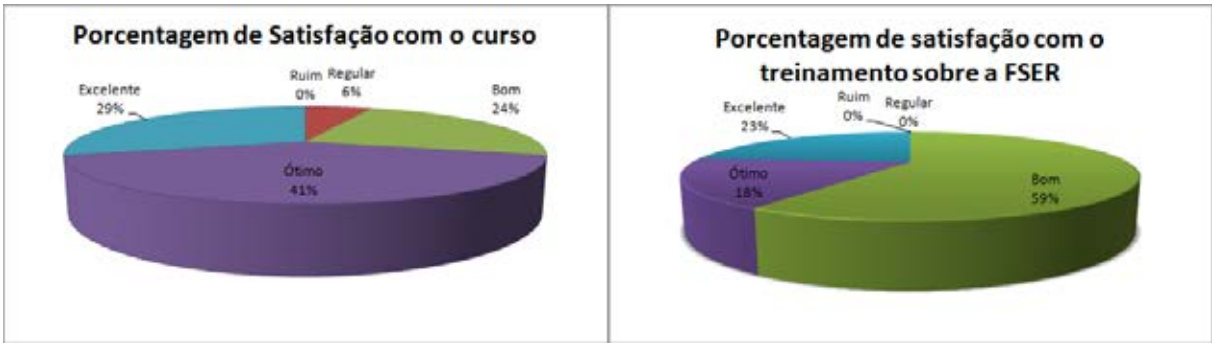


Figura 30. Avaliação do nível de satisfação dos participantes quanto ao curso e o treinamento sobre a FSER

4.6 Considerações Finais

Como pode ser visto nos resultados obtidos na realização do experimento, as hipóteses alternativas foram confirmadas tanto pela Turma A de São José do Rio Preto quanto pela Turma B de Presidente Prudente. As avaliações dos participantes puderam confirmar a eficiência da utilização da FSER no apoio à realização das atividades de Gerência de Requisitos, utilizando a técnica de *Design Rationale* para formalizar e armazenar as informações advindas das discussões realizadas sobre os exemplos dados nos Documentos de Requisitos e Atas de Reuniões.

Além disso, pode-se constatar a contribuição do curso e do treinamento sobre a ferramenta através da avaliação feita pelos participantes, que em geral consideraram o curso e o treinamento bons.

Capítulo 5 – Conclusão

5.1 Considerações Finais

Este trabalho apresentou uma proposta de integração da técnica de *Design Rationale* na Gerência de Requisitos de software, por meio da expansão funcional da Ferramenta de Suporte à Engenharia de Requisitos - FSER.

Durante o processo de integração, notou-se que para uma maior abrangência da ferramenta no que tange à Gerência de Requisitos, fez-se necessário desenvolver também uma funcionalidade adicional: Diagramas de Casos de Uso. Na sequência, foi proposto e executado um Experimento Controlado a fim de avaliar a eficiência dos resultados obtidos no âmbito da Gerência de Requisitos de software.

Assim, na seção 5.2 deste capítulo, são apresentadas as contribuições finais deste trabalho e na seção 5.3 as propostas de trabalhos futuros para este trabalho.

5.2 Contribuições

Durante todo o processo de desenvolvimento de software, é necessário empregar uma atenção especial às mudanças e obstáculos que surgem e podem afetar os requisitos do software.

Nesse contexto, é possível notar que as novas funções agregadas à Ferramenta de Suporte à Engenharia de Requisitos – FSER são de importantes.

Com relação à criação de Diagramas de Casos de Uso possibilita uma melhor visão dos requisitos, facilitando o trabalho de modelagem, análise e validação dos requisitos. A aplicação da técnica de *Design Rationale*, por sua vez, contribui para auxiliar as tomadas de decisão, pois ela torna clara determinadas perspectivas de um problema que por vezes não são retidas e compartilhadas.

Como a proposta inicial e central desta pesquisa foi a integração da técnica de *Design Rationale* para apoiar tomadas de decisão no âmbito da Gerência de Requisitos de software, tal funcionalidade foi submetida a uma avaliação através da execução de um Experimento Controlado.

Da análise dos resultados obtidos com o experimento realizado, é possível afirmar que a ferramenta apóia efetivamente a Gerência de Requisitos no que concerne o armazenamento de informações sobre discussões e experiências, de maneira fácil, contribuindo com a disseminação do conhecimento e com a formação de uma base de dados que apóie o desenvolvimento de projetos futuros.

5.3 Trabalhos Futuros

As alterações e expansões realizadas na FSER atualmente contemplam, mesmo que parcialmente, todas as atividades da Engenharia de Requisitos de Software. Porém, diversas modificações podem ser realizadas a fim de aprimorar as funções já existentes, como a ampliação das técnicas de rastreabilidade contempladas pela ferramenta.

No que concerne à atividade de Gerência de Requisitos e gestão do conhecimento, existe a possibilidade de desenvolvimento de funções relacionadas à Web Semântica, propiciando a criação de dicionário de definição de termos e fornecendo a recuperação da informação de maneira inteligente e customizada.

Referências Bibliográficas

ALEXANDER, I.; BEUS-DUKIC, L. **Discovering Requirements: How to Specify Products and Services**. London: Wiley, 1° ed., 2009.

ARCHIBALD, R.; PRADO, D. S. Pesquisa sobre maturidade em gerenciamento de projetos - Relatório anual de 2010. **Project Management Maturity Research**. Disponível em <http://www.maturityresearch.com/novosite/2010/downloads/Pesquisa Maturidade2010-Rel.GeralCompleto-V4.pdf>. Acesso em: 03 ago. 2012.

AURUM, A.; WOHLIN, C. **Engineering and Managing Software Requirements.** Berlin: Springer, 1° ed., 2005.

BASILI, V. R. et al. **Building knowledge through families of experiments. IEEE Transactions on Software Engineering**, v. 25, n. 4, 1999.

BURGE, J. Design rationale: Researching under uncertainty. **Artificial Intelligence for Engineering Design, Analysis and Manufacturing**. NewYork, v. 22, n. 4, 2008.

BURGE, J. E.; BROWN, D. C. SEURAT: integrated rationale management. In: International Conference on Software Engineering, 30., New York, USA. **Anais…** New York: v. 1, n. 835-838, 2008.

BERENBACH, B. et al. **Software & Systems Requirements Engineering in Practice**. New York: Mc Graw Hill, 3° ed., 2009.

CAMPONOGARA, M.; SILVEIRA, M. QOC*: utilizando Design Rationale como ferramenta para gerenciar conhecimento em projetos de software. **Revista de Informática Teórica e Aplicada.** Rio Grande do Sul, v. 16, n. 2, p. 9-23, 2009.

CaseComplete. Disponível em <http://www.casecomplete.com/>. Acesso em setembro/2013.

CIMIANO, P., REYLE, U. Towards Foundational Semantics - Ontological Semantics Revisited. In: International Conference on Formal Ontology in Information Systems, 4., Baltimore, USA. **Anais...** Baltimore: v. 9, n. 51-62, 2006.

DEITEL, H. **Java: Como Programar**. 6a Edição. Pearson Education do Brasil. 2005.

FUZII, R. et al. Apoio automatizado para aplicação de técnicas de elicitação de requisitos. **Revista Eletrônica de Sistemas de Informação.** Campo Largo, v. 8, n. 1, p. 1-14, 2009.

GOTEL O.; FUNKELSTEIN, A. An Analysis of the Requirements Tracebility Problem. In: Conference on Requirements Engineering, 1., Springs, USA. **Anais...** Springs: p. 94-101, 1994.

HAZAN, C.; LEITE, J. Indicadores para a Gerência de Requisitos. In: Workshop em Engenharia de Requisitos, 6., Piracicaba. **Anais…** Rio de Janeiro: Rio de Janeiro: Pontifícia Universidade Católica do Rio de Janeiro, p. 285-308, 2003.

IBM. **Rational RequisitePro**. Disponível em <http://www-01.ibm.com/software/awdtools/reqpro/>. Acesso em setembro/2013.

IEEE. **IEEE Standards Collections, Software Engineering.** IEEE 610.12-1990§3. 1996.

IEEE. **Recommended Practice for Software Requirements Specifications.** IEEE Standards Board. 1998.

ITO, M. et al. Support Tool to the Validation Process of Functional Requirements. **IEEE Latin America Transactions.** São Paulo, vol. 9, n. 5, p 889-894, 2011.

JACOBSON et al. **Unified software development process, the complete guide to the unified process from the original designers**. Chicago: Addison-Wesley Publishing Company, 1° ed., 1999.

KAWAMOTO, S. **Proposta de um modelo de representação de Design Rationale para projetos de sistemas ERP.** Dissertação (Mestrado). São Paulo: Escola Politécnica da USP – POLI/USP, 2007.

KLEIN, M. An Exception Handling Approach to Enhancing Consistency, Completeness and Correctness in Collaborative Requirements Capture. In: International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, 6., Cambridge, USA. **Anais…** Cambridge: IEEE Computer Society, 1997.

KUNZ, W.; RITTEL, H. **Issues as Elements of Information Systems**. Berkeley: University of California, 1° ed., 1970.

LAMSWEERDE, A. **Requirements Engineering: From System Goals to UML Models to Software Specifications.** Chichester: John Wiley, 1° ed., 2009.

LARA, S. M. **Um suporte à captura informal de Design Rationale**. Dissertação (Mestrado). São Carlos: Instituto de Ciências Matemáticas e Computação – ICMC/USP, 2005.

LARMAN, C. **Utilizando UML e Padrões**. Rio Grande do Sul: Bookman, 3° ed., 2007.

LAUSEN, S. **Software Requirements: Styles & Techniques**. New York: Addison-Wesley, 1° ed., 2002.

LEE, J. SIBYL: A qualitative decision management system. WINSTON, P. H.; SHELLARD, S.. **Artificial Intelligence at MIT: Expanding Frontiers** (p. 104-113). Cambidge: MIT Press, 1° ed., 1990.

LEE, J. Design Rationale Systems: Understanding the Issues. **IEEE Expert.** Honolulu, v. 12, n. 3, p. 78-85, 1997.

LEE, J.; LAI, K. A Comparative Analysis of Design Rationale Representations. **Human-Computer Interaction on Design Rationale**. Massachusetts, 1991.

LEFFINGWELL, D.; WIDRIG, D. **Managing Software Requirements: a Use Case Approach.** New York: Addison-Wesley, 2° ed., 2003.

LOURIDAS, P; LOUCOPOULUS, P. A generic model for reflective design. **ACM Transactions on Software Engineering and Methodology**. New York, v. 9, p. 199 – 237, 2000.

LU, H. et al. KDD, techniques and applications. In: Pacific-Asia Conference on Knowledge Discovery and Data Mining, 1., Singapura. **Anais…** Singapura: Information Technology Institute & National University of Singapore, 1997.

MARTINIE, C. et al. DREAMER: a design rationale environment for argumentation, modeling and engineering requirements. In: ACM International Conference on Design of Communication, 28., New York. **Anais…** New York: SIGDOC'11, p. 73-80, 2010.

MCLEAN et al. Questions, Options, and Criteria: Elements of Design Space Analysis. **Human-Computer Interactions**, v. 6, p. 201 – 250, 1991.

MICHAELLIS, 2013. Disponível em: <www.uol.com.br/michaelis>. Acesso em 10 de janeiro de 2013.

MIRANDA, J. W. P. **Automatização de técnicas de rastreabilidade de requisitos para expansão de Ferramenta de Apoio à Engenharia de Requisitos**. Monografia (Graduação). São José do Rio Preto: Universidade Estadual Paulista – UNESP, 2013.

MORAN, T.; CARROL, J. **Design Rationale: Concepts, Techniques, and Use.** New Jersey: Lawrence Erlbaum Associates, 1° ed., 1996.

OSRMT. **Open Source Requirement Management Tool**. Disponível em <http://sourceforge.net/projects/osrmt/>. Acesso em setembro/2013.

PATTON, R. **Software Testing**. Boston: Sams Publishing, 2° ed., 2001.

POTTS, C.; BRUNS, G.. Recording the Reasons for Design Decisions. In: International Conference on Software Engineering, 10., Singapore. **Anais…** Singapura: p. 418-427,1988.

POTTS, C. A Generic Model for Representing Design Methods. In: International Conference on Software Engineering, 10., Pittsburgh. **Anais...** 1989.

PRESSMAN, R. **Engenharia de Software.** São Paulo: McGraw-Hill, 7° ed., 2010.

QIAN et al. **Java Web development illuminated**. New York: Jones & Bartlett Learning, 1° ed., 2007.

REGLI et al. A Survey of Design Rationale Systems: Approaches, Representation, Capture and Retrieval. **Engineering with Computers**, vol. 16, n. 3-4, 2000.

ROBINSON, W. N.; VOLKOV, S.. Supporting the Negotiation Life Cycle. **Communication of ACM**, vol. 41, n.5, pp 95-102. 1998.

SANTOS, E. G.; MEDEIROS, A. P.. Design Rationale Representation in Requirements Engineering using the KAOS meta-model. In: Workshop em Engenharia de Requisitos, 14., Rio de Janeiro. **Anais...** Rio de Janeiro: Pontifícia Universidade Católica do Rio de Janeiro, p. 383-394, 2011.

SHUM et al. Graphical argumentation and design cognition. **Human-Computer Interaction**, vol. 12, n. 3, 1997.

SIMONATO et al. Apoio Computacional para o Processo de Especificação de Requisitos. In: Conferencia Latinoamericana en Informática, 38., Medellin, Colombia. **Anais…** Medellin: Electronic Notes in Theoretical Computer Science, 2012.

SOMMERVILLE, I. Engenharia de Software. São Paulo: Addison-Wesley, 9° Ed., 2011.

STANDISH GROUP. **Chaos Report**. Boston: The Standish Group International, 2011.

TOULMIN, S. E. **The uses of Argument**. Cambridge: Cambridge University Press, 1958.

WANG, H. et al. The retrieval of structured design rationale for the re-use of design knowledge with an integrated representation. **Advanced Engineering Informatics Journal.** Amsterdam, v. 26, n. 2, p. 251-266, 2012.

WOHLIN, C. et al. **Experimentation in software engineering: An introduction**. Boston: Kluwer Academic Publishers, 2012.

ZITELLI, J. M. M. **Reengenharia de Software voltada para melhoria de Interface**. Monografia (Graduação). São José do Rio Preto: Universidade Estadual Paulista – UNESP, 2012.

Apêndice A: Documentos de Requisitos de Software

Documento de Requisitos A: Posto de Serviços Royal

O posto de serviços Royal fornece três tipos de serviços para seus clientes: abastecimento, manutenção de veículos e estacionamentos. Isto é, um cliente pode colocar combustível no tanque de seu veículo (carro, motocicleta ou caminhão), pode ter o veículo reparado ou pode estacionar seu carro em uma vaga disponível.

Um cliente tem a opção de ser cobrado automaticamente no momento da prestação do serviço (abastecimento, manutenção ou estacionamento) ou de receber uma conta mensalmente. Em qualquer um dos casos, o cliente pode pagar utilizando dinheiro, cartão de crédito ou cheque.

O preço dos combustíveis vendidos pelo Posto de Serviços Royal depende do tipo do combustível, como diesel, gasolina comum, gasolina aditivada ou álcool.

O serviço de manutenção é cobrado de acordo com o custo das peças e da mão de obra.

O estacionamento é cobrado com base em taxas diárias, semanais ou mensais.

Os preços do combustível, dos serviços de manutenção, das peças e do estacionamento podem variar. Somente Marcos, o gerente do posto, pode modificar os preços. De acordo com o seu critério, o gerente pode dar descontos a alguns clientes em particular, e esse desconto pode variar de cliente para cliente. Uma sobretaxa de cinco por cento, referente ao imposto local, é aplicada a todas as compras.

1. O sistema proposto deverá fazer um controle das contas mês a mês, e dos produtos e serviços fornecidos pelo posto de gasolina, diariamente. Esses resultados devem ser relatados ao gerente, quando ele solicitar.

2. O gerente do posto utilizará o sistema para controlar o estoque. O sistema alertará se o estoque estiver baixo e, automaticamente, fará os pedidos de novas peças e de mais combustível.
3. O sistema analisará o histórico de crédito e enviará avisos de cobrança aos clientes cujos pagamentos estão atrasados. As faturas devem ser enviadas aos clientes no primeiro dia do mês, depois que as compras forem feitas. O pagamento deve ser realizado no primeiro dia do mês seguinte. Qualquer conta que não seja paga dentro de um prazo de 90 dias da data de faturamento acarretará o cancelamento do crédito do cliente.
4. O sistema enviará mensagens periódicas para os clientes, lembrando-os de quando seus veículos devem fazer manutenção. Normalmente, é necessária uma manutenção a cada seis meses.

Os clientes podem alugar vagas no estacionamento por dia. Cada cliente deve solicitar, a partir do sistema, uma vaga disponível. O gerente do posto pode ter um relatório mensal resumindo quantas vagas do estacionamento estão sendo ocupadas ou estão disponíveis.

Documento de Requisitos B: Cervejaria YesBeer

A cervejaria YesBeer produz diversos produtos líquidos, tais como: cerveja branca, cerveja escura, guaraná normal, guaraná light, água mineral com gás, água mineral sem gás, etc.

A cervejaria possui diversas unidades de produção (fábricas). Cada produto pode ser produzido em mais de uma unidade de produção. Uma unidade de produção pode produzir mais de um produto, mas não necessariamente todos. Por exemplo, na unidade da Bahia, são produzidas apenas cerveja branca e cerveja escura. No Espírito Santo são produzidas, por exemplo, cerveja branca, água mineral com gás e sem gás e guaraná normal.

Os produtos são engarrafados em embalagens dos mais variados tipos: garrafas de plástico de tamanhos variados, garrafas de vidro, latinhas de alumínio de tamanhos variados, etc. Um produto pode ser armazenado em mais de uma embalagem e uma embalagem pode ser usada por mais de um produto.

A cervejaria possui várias equipes de vendas espalhadas pelo país. Uma equipe atende somente a uma região por vez. Uma região pode ter mais de uma equipe de vendas ao mesmo tempo. Uma equipe não muda de região. Cada equipe é composta por vendedores e por um gerente.

1. O sistema deverá armazenar o histórico da gerência de cada equipe, manipulando e armazenando o período de cada um dos integrantes em cada equipe.
2. O sistema deverá manipular os dados dos clientes, como razão social, CNPJ, endereço, telefone e responsável.

3. O sistema deverá gerenciar os pedidos de venda, primeiramente armazenando dados do vendedor e cliente envolvido, o número do pedido e sua data de emissão, e os produtos vendidos.
4. Periodicamente, o gerente solicitará a emissão de relatórios gerenciais que informem a quantidade de produção das bebidas por unidade de produção.

A Cervejaria YesBeer faz constantes campanhas publicitárias, que devem ser controladas pelo sistema. Em cada campanha publicitária é necessário relacionar os preços promocionais de cada produto e o nome do garoto/garota propaganda da campanha.

Apêndice B: Atas de Reuniões

Ata de Reunião dia 29/02/2001 – Posto de Serviços Royal

Participantes: Carlos Augusto Birigui (Analista de Negócios)
Marcos Itacará (Gerente do Posto)
Vinícius de Almeida Prado (Frentista do Posto)
Felipe Soares dos Santos (Caixa do Posto)
Ausentes: Vicente de Sá Madeiro (Proprietário do Posto)

PAUTA:

- Recolher assinatura do contrato de prestação de serviços dos responsáveis pelo Posto de Serviço Royal;
- Definir as responsabilidades dos envolvidos no processo de desenvolvimento do sistema para o Posto de Serviços Royal.

DESENVOLVIMENTO:

A reunião começou às quinze horas do dia trinta do mês de fevereiro do ano de dois mil e um.

- Primeira pauta: não foi possível recolher todas as assinaturas dos responsáveis pois o patrocinador do projeto (Vicente de Sá Medeiro) de desenvolvimento do sistema para o Posto de Serviços Royal não estava presente. A decisão dos participantes da reunião foi marcar uma nova reunião com a participação do patrocinador.
- Segunda pauta: após diversas discussões sobre as responsabilidades dos envolvidos, ficou estabelecido que:
  i. O frentista Vinícius de Almeida Prado possui ligação direta com todos os requisitos relacionados aos serviços de abastecimento.
  ii. O caixa Felipe Soares dos Santos será responsável pela validação das informações relacionadas à contabilidade e finanças.
  iii. O gerente Marcos Itacará terá responsabilidade sobre todo o desenvolvimento do sistema.

iv. O analista Carlos Augusto Birigui será responsável por acompanhar o desenvolvimento de todo o sistema.

Não havendo mais assuntos a tratar, eu Carlos Augusto Birigui, lavrei a presente ata, que segue assinada por mim e por todos os presentes.

São José do Rio Preto, 29 de fevereiro de 2001.

Ata de Reunião 14/03/2001 – Posto de Serviços Royal

Participantes: Carlos Augusto Birigui (Analista de Negócios)
Marcos Itacará (Gerente do Posto)
Vinícius de Almeida Prado (Frentista do Posto)
Felipe Soares dos Santos (Caixa do Posto)
Vicente de Sá Madeiro (Proprietário do Posto)

PAUTA:

- Informar aos envolvidos o status do desenvolvimento do sistema do Posto de Serviços Royal;
- Realização de discussões gerais sobre o desenvolvimento do sistema do Posto de Serviços Royal.

DESENVOLVIMENTO:

A reunião começou às dez horas e vinte minutos do dia quatorze do mês de março do ano de dois mil e um.

- Primeira pauta: O analista de negócios Carlos Augusto Birigui informa aos participantes do projeto que:

i. Os requisitos relacionados à cobranças de estacionamento já estão na fase de construção (**Lembre-se que até chegar a essa fase, o requisito passou por outras fases também!*).

ii. Os requisitos relacionados ao serviço de manutenção estão na fase de elaboração (**Lembre-se que até chegar a essa fase, o requisito passou por outras fases também!*).

- Segunda pauta: Após discussões realizadas entre os participantes da reunião, o patrocinador do projeto, Vicente de Sá Madeiro, fez as seguintes considerações:

i. Os requisitos relacionados à cobrança dos clientes: ele deseja que o sistema faça o controle de um *crediário próprio*, ou seja, devem ser realizadas mudanças nos requisitos relacionados à cobranças: O cliente tem a opção de ser cobrado automaticamente no momento da prestação do serviço (abastecimento, manutenção ou estacionamento) ou de receber uma conta mensalmente. O cliente pode pagar utilizando dinheiro, cartão de débito ou *crediário próprio*. (*Como desenvolver esse crediário? A equipe conseguirá*

desenvolver esse sistema dentro do prazo estabelecido? Como funciona um sistema de crediário próprio? Isso afetará o orçamento do projeto? Afetará outros requisitos definidos?)

ii. O gerente Marcos Itacará não fará mais parte da equipe, e deverá ser desvinculado ao projeto, sendo que o patrocinador do projeto ficará responsável por todos os requisitos identificados no projeto.

Não havendo mais assuntos a tratar, eu Carlos Augusto Birigui, lavrei a presente ata, que segue assinada por mim e por todos os presentes.

São José do Rio Preto, 14 de março de 2001.

Ata de Reunião 15/12/2008 – Cervejaria YesBeer

Participantes: Ana Maria Barbosa (Gerente da Região Oeste de São Paulo)
Viviane das Dores (Vendedora da Região Oeste de São Paulo)
Plínio Teixeira (Vendedor da Região Oeste de São Paulo)
Janaína Teixeira (Analista de Negócios)

Ausentes: Edson Pigosi (Analista de Sistemas)

PAUTA:

- Estabelecimento de prioridades para o desenvolvimento do sistema;
- Definir as responsabilidades dos envolvidos no processo de desenvolvimento do sistema.

DESENVOLVIMENTO:

A reunião começou às vinte horas e dez minutos do dia trinta do mês de dezembro do ano de dois mil e oito.

- Primeira pauta: Como o sistema será primeiramente utilizado pela equipe de representantes da empresa YesBeer da Região Oeste do estado de São Paulo, eles serão os principais interessados no sistema e deverão participar do processo de desenvolvimento. A equipe, após uma longa conversa, decidiu que é prioridade o desenvolvimento do pedido de venda dos produtos, enfatizando que a maior contribuição do sistema deve ser o envio dos pedidos às unidades de produção mais próximas.
- Segunda pauta: após diversas discussões sobre as responsabilidades dos envolvidos, ficou estabelecido que:

i. A gerente Ana Maria Barbosa ficará responsável por supervisionar o desenvolvimento das funcionalidades de pedidos de venda e histórico da gerência.

ii. A vendedora Viviane das Dores ficará responsável pela validação da funcionalidade relacionada aos clientes.

iii. O vendedor Plínio Teixeira será responsável pelas campanhas publicitárias.
iv. O analista de sistemas Edson Pigosi será responsável pelo desenvolvimento do sistema, juntamente com a analista de negócios Janaína Teixeira.

Não havendo mais assuntos a tratar, eu Janaína Teixeira, lavrei a presente ata, que segue assinada por mim e por todos os presentes.

Presidente Prudente, 15 de dezembro de 2008.

Ata de Reunião 25/01/2009 – Cervejaria YesBeer

Participantes: Ana Maria Barbosa (Gerente da Região Oeste de São Paulo)
Viviane das Dores (Vendedora da Equipe da Região Oeste de São Paulo)
Plínio Teixeira (Vendedor da Região Oeste de São Paulo)
Janaína Teixeira (Analista de Negócios)
Edson Pigosi (Analista de Sistemas)

PAUTA:
- Informar aos envolvidos o status do desenvolvimento do sistema;
- Solicitação de mudanças no sistema.

DESENVOLVIMENTO:
A reunião começou às dezenove horas do dia vinte e cinco do mês de janeiro do ano de dois mil e nove.
- Primeira pauta: O analista de negócios Carlos Augusto Birigui informa aos participantes do projeto que todos os requisitos identificados estão na fase de elaboração, devido a férias coletivas concedidas aos funcionários da empresa que está desenvolvendo o sistema (**Lembre-se que até chegar a essa fase, o requisito passou por outras fases também!*).
- Segunda pauta: o presidente da YesBeer enviou um comunicado ao gerente da Região do Oeste de São Paulo solicitando o desenvolvimento de funcionalidades no sistema ligadas à tecnologia de *Business Inteligence*, mais especificamente um módulo que implemente *funções de CRM* (*Como desenvolver essa nova funcionalidade? A equipe conseguirá desenvolver esse sistema dentro do prazo estabelecido? Como funciona um sistema de CRM? Isso afetará o orçamento do projeto? Afetará outros requisitos definidos?*).

Não havendo mais assuntos a tratar, eu Janaína Teixeira, lavrei a presente ata, que segue assinada por mim e por todos os presentes.

Presidente Prudente, 25 de janeiro de 2009.

Apêndice C: Formulários

TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO

Pelo presente instrumento, declaro que _____, portador da cédula de identidade RG de número _____, e do CPF de número _____, concordou em participar da pesquisa científica de nível acadêmico de mestrado intitulada **Apoio à Gerência de Requisitos de Software utilizando *Design Rationale***, realizando atividades de Gerência de Requisitos de Software e estando ciente dos seguintes termos:

A participação na pesquisa terá duração de aproximadamente 4 (quatro) dias não necessariamente consecutivos, com as datas a serem determinadas, tendo como local a **Faculdade de Ciências e Tecnologia da Universidade Estadual Paulista Júlio de Mesquita Filho**, **UNESP**, situada à Rua Roberto Simonsen, 305, CEP 19060-90, nesta cidade de Presidente Prudente-SP. As informações do participante serão tratadas de maneira anônima e confidencial, ou seja, em nenhum momento será divulgado o nome do participante em qualquer fase do estudo. Os dados coletados poderão ser utilizados em pesquisas futuras, permanecendo o anonimato do participante. A participação é voluntária e sem remuneração, podendo o participante desistir a qualquer momento de participar da pesquisa. A participação consistirá em responder questionários e realizar atividades de Gerência de Requisitos de Software, não havendo riscos de qualquer natureza. O benefício relacionado à participação consiste em aumentar o conhecimento científico para a área de Ciência da Computação.

Declaro estar ciente do inteiro teor deste termo de consentimento e estar de acordo em participar da pesquisa proposta.

Presidente Prudente, ____ de _____ de 2013.

_____
Assinatura do participante

_____
Assinatura do aluno

_____
Assinatura do orientador

TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO

Pelo presente instrumento, declaro que _____, portador da cédula de identidade RG de número _____, e do CPF de número _____, concordou em participar da pesquisa científica de nível acadêmico de mestrado intitulada **Apoio à Gerência de Requisitos de Software utilizando *Design Rationale***, realizando atividades de Gerência de Requisitos de Software e estando ciente dos seguintes termos:

A participação na pesquisa terá duração de aproximadamente 4 (quatro) dias não necessariamente consecutivos, com as datas a serem determinadas, tendo como local o **Instituto de Biociências, Letras e Ciências Exatas da Universidade Estadual Paulista Júlio de Mesquita Filho**, **UNESP**, situada à Rua Cristóvão Colombo, 2265, CEP 15054-000, nesta cidade de São José do Rio Preto-SP. As informações do participante serão tratadas de maneira anônima e confidencial, ou seja, em nenhum momento será divulgado o nome do participante em qualquer fase do estudo. Os dados coletados poderão ser utilizados em pesquisas futuras, permanecendo o anonimato do participante. A participação é voluntária e sem remuneração, podendo o participante desistir a qualquer momento de participar da pesquisa. A participação consistirá em responder questionários e realizar atividades de Gerência de Requisitos de Software, não havendo riscos de qualquer natureza. O benefício relacionado à participação consiste em aumentar o conhecimento científico para a área de Ciência da Computação.

Declaro estar ciente do inteiro teor deste termo de consentimento e estar de acordo em participar da pesquisa proposta.

São José do Rio Preto, ____ de _____ de 2013.

_____
Assinatura do participante

_____
Assinatura do aluno

_____
Assinatura do orientador

QUESTIONÁRIO PARA IDENTIFICAÇÃO DE PARTICIPANTES

1. Sexo:
   a: Masculino
   b: Feminino

2. Formação acadêmica:
   a: Graduação
   b: Especialização
   c: Mestrado
   d: Doutorado
   e: Outro (especificar): _____

3. Em relação a sua formação acadêmica (graduação), você:
   a: Está cursando
   b: Já concluiu

4. Se concluiu, qual é o tempo decorrido após sua formação acadêmica (em anos): _____

5. Em relação a sua experiência com projetos de desenvolvimento de software (marque “S” ou “N”):
   ( ) Nunca participei de projetos de desenvolvimento de software
   ( ) Tenho participado de projetos de desenvolvimento de software na área acadêmica

( ) Tenho participado de projetos de desenvolvimento de software profissionalmente

6. Tempo de experiência com projetos de desenvolvimento de software (em anos): _____

7. Você já realizou atividades de Gerência de Requisitos de software?
   a: Sim
   b: Não

8. Você conhece alguma técnica ou ferramenta para Gerência de Requisitos de software?
   a: Sim
   b: Não

9. Você sabe o que é *Design Rationale*?
   a: Sim
   b: Não

10. Você conhece ou já usou alguma ferramenta para Gerência de Requisitos de software?
    a: Sim
    b: Não

11. Para organizar, documentar e gerenciar informações relativas aos requisitos de um projeto de desenvolvimento de software, o que você utiliza (marque “S” ou “N”)?
    ( ) Código fonte (incluindo comentários no código fonte)
    ( ) Leitura de documentação específica (Documentos de Especificação de Requisitos, Entrevistas, Questionários, Diagramas UML)
    ( ) Outro (especificar): _____

QUESTIONÁRIO PARA AVALIAÇÃO DO TREINAMENTO

1. Você compreendeu o objetivo deste curso?
   a: Sim
   b: Não
   c: Parcialmente

2. Você compreendeu o que é Engenharia de Software?
   a: Sim
   b: Não
   c: Parcialmente

3. Você compreendeu o que é Engenharia de Requisitos?
   a: Sim

b: Não
c: Parcialmente

4. Você compreendeu a importância da realização de atividades relacionadas à Gerência de Requisitos?
   a: Sim
   b: Não
   c: Parcialmente

5. Você compreendeu a importância da utilização de técnicas para a realização das atividades relacionadas à Gerência de Requisitos?
   a: Sim
   b: Não
   c: Parcialmente

6. Como você julga o treinamento recebido sobre a Ferramenta de Suporte de Engenharia de Requisitos?
   a: Ruim
   b: Regular
   c: Bom
   d: Ótimo
   e: Excelente

7. Quais os seus comentários para melhorar os treinamentos teóricos e práticos que você recebeu?

_____
_____
_____
_____
_____
_____
_____
_____
_____
_____

FORMULÁRIO PARA DESCRIÇÃO DOS RESULTADOS DA EXECUÇÃO DAS ABORDAGENS

Abordagem Utilizada: □Abordagem *ad hoc* □Abordagem com utilização da FSER

1. Grau de Complexidade na realização da atividade de organização e armazenamento das informações:
   a: Alto
   b: Médio
   c: Baixo

2. Grau de Eficiência da Abordagem
   a: Fácil
   b: Regular
   c: Difícil

3. Grau de Motivação da Abordagem
   a: Existente
   b: Inexistente