JOSÉ ANDRÉ DORIGAN

UM MODELO DE PROCESSO DE ENGENHARIA DE REQUISITOS PARA PADRONIZAÇÃO E AUMENTO DA QUALIDADE

Londrina - Pr
2013

JOSÉ ANDRÉ DORIGAN

UM MODELO DE PROCESSO DE ENGENHARIA DE REQUISITOS PARA PADRONIZAÇÃO E AUMENTO DA QUALIDADE

Dissertação apresentada ao Programa de Mestrado em Ciência da Computação Departamento de Computação da Universidade Estadual de Londrina, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

Orientador: Dr. Rodolfo Miranda de Barros

Londrina - Pr
2013

Catalogação elaborada pela Divisão de Processos Técnicos da Biblioteca Central da Universidade Estadual de Londrina.

Dados Internacionais de Catalogação-na-Publicação (CIP)

D697m Dorigan, José André.
Um modelo de processo de engenharia de requisitos para padronização e aumento da qualidade / José André Dorigan. – Londrina, 2013.
81 f. : il.

Orientador: Rodolfo Miranda de Barros.
Dissertação (Mestrado em Ciência da Computação). – Universidade Estadual de Londrina, Centro de Ciências Exatas, Programa de Pós-Graduação em Ciência da Computação, 2013.
Inclui bibliografia.

1. Engenharia de requisitos – Teses. 2. Software – Desenvolvimento – Teses. 3. Engenharia de software – Teses. I. Barros, Rodolfo Miranda de. II. Universidade Estadual de Londrina. Centro de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação. III. Título.

CDU 519.68.02

JOSÉ ANDRÉ DORIGAN

UM MODELO DE PROCESSO DE ENGENHARIA DE REQUISITOS PARA PADRONIZAÇÃO E AUMENTO DA QUALIDADE

Dissertação apresentada ao Programa de Mestrado em Ciência da Computação Departamento de Computação da Universidade Estadual de Londrina, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação.

BANCA EXAMINADORA

_____

Prof. Dr. Rodolfo Miranda de Barros
Universidade Estadual de Londrina

_____

Prof. Dr. Lourival Aparecido de Góis
Universidade Tecnológica Federal do Paraná

_____

Prof. Dr. Mario Lemes Proença Jr.
Universidade Estadual de Londrina

_____

Prof. Dr. Alan Salvany Felinto
Universidade Estadual de Londrina

Londrina, 19 de Fevereiro de 2013.

A toda minha família e amigos pelo apoio e compreensão.

AGRADECIMENTOS

Primeiramente, gostaria de agradecer aos meus pais, José Luis e Aparecida pelo incentivo, carinho e compreensão que tiveram comigo ao longo dos meus 4 anos de Graduação e agora 2 anos de Mestrado. Mesmo estando tão longe eu sempre soube que quando eu precisasse eles estariam lá por mim. À eles um muito obrigado seria pouco.

As minhas irmãs Carina e Patrícia que me ajudaram em várias oportunidades e que eu pude ajudar também com pouco, mas inestimável, experiência que eu pude ter morando longe de casa.

A Deus, por me permitir ter tantas oportunidades acadêmicas, sociais e profissionais.

Ao Professor Dr. Rodolfo Miranda de Barros, que além de orientador nesse trabalho é um amigo que tenho muito orgulho de ter conhecido, sempre que precisei de algo me ajudou e continua ajudando nesse final de caminhada. Espero ainda, que possamos trabalhar juntos, quem sabe, num possível doutorado.

Ao Professor Dr. Mario Lemes Proença Jr. com quem tive a oportunidade de trabalhar desde a metade do ano de 2009, no projeto de pesquisa GAIA - Fábrica de Projetos de TIC, que juntamente com o Prof. Rodolfo, incentivou a mim e aos outros componentes da equipe mostrando que deveríamos pensar à frente, buscando utilizar novas tecnologias e sempre manter o comprometimento com o trabalho desenvolvido.

A todos os professores e funcionários do Departamento de Computação, que buscaram preparar ao máximo os alunos, para que tivéssemos uma ótima qualificação acadêmica. Mesmo nos momentos difíceis, sempre existiu alguém em quem pudemos confiar e que trabalhou para nos oferecer o melhor.

Aos meus amigos de longa data (os que ficaram lá em Amparo e os de Londrina), aos meus amigos de hoje, e também a aqueles que estiveram presente nesse tempo que estou em Londrina. Sempre me lembrarei das risadas, festas e aulas em que estivemos juntos. Todos foram e são importantes para mim, e fico muito feliz de tê-los conhecido.

Por fim, a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) por proporcionar incentivo econômico e a Universidade Estadual de Londrina, por oferecer a qualidade de ensino e o curso de Mestrado em Ciência da Computação.

"*Não importa o quão estreito seja o portão e quão repleta de castigos seja a sentença, eu sou o dono do meu destino, eu sou o capitão da minha alma*".

William Ernst Henley.

DORIGAN, José André. **Um modelo de processo de engenharia de requisitos para padronização e aumento da qualidade**. 2013. 81 f. Dissertação (Mestrado em Ciência da Computação) – Universidade Estadual de Londrina, Londrina, 2013.

RESUMO

Um dos maiores problemas encontrados na Engenharia de Requisitos é o fato dos requisitos estarem mal especificados, inconsistentes com a necessidade do cliente ou mal escritos. Este trabalho apresenta um modelo de Processo de Engenharia de Requisitos que possibilita a padronização da descrição de requisitos, através do reuso de palavras, buscando aumentar a qualidade da especificação. O processo aqui proposto atuará na área de Análise, Especificação e Validação de Requisitos auxiliando o Engenheiro de Requisitos na escrita dos requisitos em Linguagem Natural. Apresenta-se um estudo comparativo entre processos que lidam com a garantia de qualidade em requisitos e um Estudo de Caso para avaliar e validar o processo proposto, identificando benefícios da sua utilização num desenvolvimento de software.

Palavras-chave: Processo de Engenharia de Requisitos. Especificação e Validação de Requisitos. Qualidade na Descrição de Requisitos. Reuso de Requisitos.

DORIGAN, José André. **A process model of requirements engineering for standardization and increased quality**. 2013. 81 f. Dissertation (Measter's Degree in Computer Science) – Universidade Estadual de Londrina, Londrina, 2013.

ABSTRACT

One of the biggest problems encountered in Requirements Engineering is the fact that requirements are poorly specified, inconsistent with the client needs or badly written. This paper presents a model of Requirements Engineering Process for requirements description standardization through the reuse of words, seeking to improve the requirements specification quality. The proposed process will act in Requirements Analysis, Specification and Validation assisting the Requirements Engineer on writing requirements in Natural Language. We present a comparative study between processes that deal with requirements quality assurance and a Case Study to evaluate and identify benefits of its use in a software development.

Key words: Requirements Engineering Process. Requirements Specification and Validation. Quality on Requirements Description. Requirements Reuse.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

ER	Engenheiro de Requisitos
IEEE	Instituto de Engenheiros Elétricos e Eletrônicos - *Institute of Electrical and Electronics Engineers*
UML	Linguagem de Modelagem Unificada - *Unified Modeling Language*
GRE	Gerenciamento de Requisitos
ASD	A Ser Determinado
SRS	Especificação de Requisitos de Software - *Software Requirements Specification*
PBR	Leitura Baseada em Perspectiva - *Perspective-Based Reading*
LN	Linguagem Natural
DRS	Documento de Requisitos de Software
BDHR	Banco de Dados Histórico de Requisitos
TIC	Tecnologia da Informação e Comunicação
COU	Clínica Odontológica Universitária
CPOD	Dentes Cariados, Perdidos ou Obturados
CPOS	Superfície Cariada, Perdida ou Obturada
CEO	Dentes Cariados, Extraídos ou Obturados
IHOS	Índice de Higiene Oral Simplificado
PDU	Prontuário Dentário Único
CASE	Engenharia de Software Assistida Computacionalmente - *Computer-Aided Software Engineering*

SUMÁRIO

1 INTRODUÇÃO

Levantar corretamente os requisitos é umas das tarefas mais importantes na construção de um software. Se o requisito é expresso em termos do comportamento do software, esse deve ser percebido por um observador externo ao software [1][2].

Aos Requisitos estão associados os principais problemas do desenvolvimento de software, na maioria dos casos eles não refletem as reais necessidades dos usuários, por serem incompletos ou inconsistentes [1].

Uma das principais dificuldades é fazer com que a especificação dos requisitos esteja em conformidade com os requisitos que o cliente deseja [3][4]. Na maioria das vezes acontece uma interpretação errada por parte do Engenheiro de Requisitos (usaremos a sigla *ER* neste trabalho) ou o Cliente não consegue expressar de forma clara suas reais necessidades [4]. De toda forma, estes problemas na especificação criam requisitos despadronizados e inconsistentes.

Um modo de evitar estas inconsistências seria se as organizações dispusessem de um processo de Engenharia de Requisitos definido [5], que ofereça padronização na descrição dos requisitos, para que todos os requisitos de um projeto estejam definidos de maneira clara, aumentando assim a qualidade da Especificação de Requisitos.

As normas IEEE 830-1998 [6] e 1233a-1998 [7] apontam diversas propriedades para que a especificação de requisitos de software e também os requisitos de sistema obtenham um bom nível de qualidade, dentre estas propriedades estão a isenção de ambiguidade, o uso da linguagem natural para descrição dos requisitos e a possibilidade da verificação de conformidade dos requisitos.

Juntamente com outras normas, processos e propriedades, a garantia da qualidade de requisitos de software se faz necessária para que ocorra a padronização dos requisitos e principalmente para que eles estejam em conformidade com a necessidade passada pelo cliente.

O modelo de processo proposto tem fundamento na ideia de que quando a padronização da descrição dos requisitos é feita através do reuso de palavras identifica-se um ganho em tempo de desenvolvimento, durante a Especificação de Requisitos, aumento na qualidade da descrição e confirmação das necessidades do cliente através da validação dos requisitos criados.

Com o aumento na qualidade da descrição dos requisitos, busca-se ainda evitar que erros surgidos nos requisitos possam afetar o desenvolvimento do projeto de

software. Segundo [1][2], o custo de correção de defeitos, erros, inconsistências na fase de Projeto é cerca de 3 a 6 vezes mais alto do que se o mesmo erro fosse identificado na fase de Análise de Requisitos. Esse custo aumenta ainda mais, quando a correção do erro é realizada nas fases posteriores do desenvolvimento do software, como, por exemplo, na fase de Testes.

Assim, o objetivo deste trabalho, é propor um modelo de processo de Engenharia de Requisitos que forneça subsídios para a padronização da descrição e o reuso das palavras utilizadas na criação dos requisitos, buscando aumento na qualidade da especificação e que esteja de acordo com as necessidades do cliente.

Além de definido e analisado em função de outros modelos e processos já existentes na vasta literatura de Engenharia de Requisitos, busca-se também verificar e validar a ocorrência de benefícios através da utilização do processo em um desenvolvimento de software. Com esse objetivo, utilizou-se a representação de um cenário de desenvolvimento real, dentro de um Estudo de Caso, onde pôde-se aplicar o modelo de processo proposto, obtendo resultados que comprovam sua eficácia e utilidade.

Tomam-se como base os trabalhos de [8][9] para a quantificação e avaliação dos resultados obtidos. Assim demonstram-se os resultados teóricos esperados através da experimentação na indústria de software.

Este trabalho está dividido da seguinte forma: no Capítulo 2 apresenta-se a fundamentação teórica em que o trabalho está baseado, no Capítulo 3 é exposto o modelo de processo de Engenharia de Requisitos proposto. No Capítulo 4 apresentam-se os trabalhos relacionados ao tema e uma Avaliação Comparativa tratando da garantia de qualidade nos requisitos, no Capítulo 5 apresenta-se um Estudo de Caso para avaliação e validação do processo proposto apontando benefícios e considerações sobre sua utilização, e finalmente no Capítulo 6 apresentam-se as conclusões deste trabalho e os trabalhos futuros previstos como continuação das atividades.

2 FUNDAMENTAÇÃO TEÓRICA

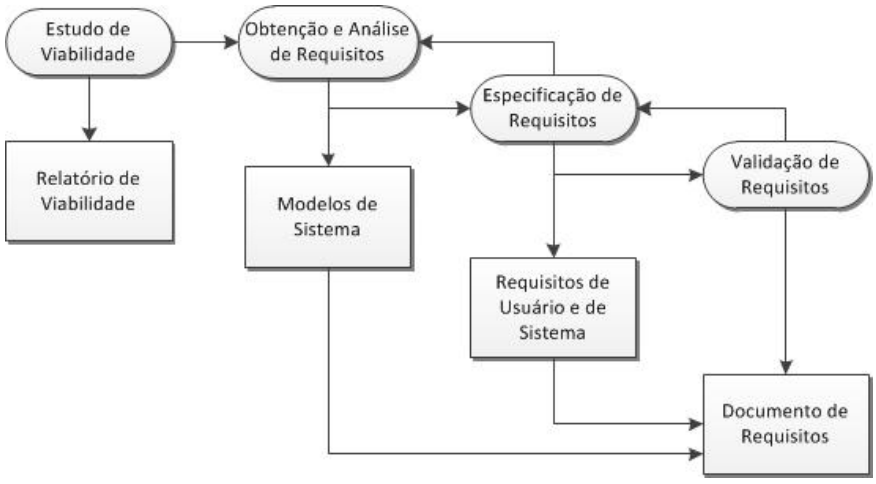
Neste capítulo buscam-se situar as áreas onde o trabalho está inserido, bem como apresentar os principais conceitos utilizados nele, tais como: Engenharia de Requisitos, Reuso de Requisitos, Garantia da Qualidade em Requisitos e Documentação de Requisitos em Linguagem Natural..

2.1 ENGENHARIA DE REQUISITOS

Para elaborar e manter uma especificação de requisitos é necessário que os desenvolvedores executem um conjunto estruturado de atividades destinadas a identificar, analisar, especificar e validar requisitos. Este conjunto de atividades iterativas recebe o nome de Processo de Engenharia de Requisitos [2][10].

Na Figura 2.1 é apresentado um diagrama de como é definido o processo de Engenharia de Requisitos e os relacionamentos entre suas áreas.

Figura 2.1 - Processo de Engenharia de Requisitos



Fonte: Sommerville, 2007 [2]

Quando a organização não dispõe deste processo definido formalmente e amplamente divulgado, os desenvolvedores elaboram as especificações de forma empírica, executando atividades não padronizadas e definidas individualmente [1][3].

Se isto ocorre, a qualidade da especificação dependerá exclusivamente da experiência e formação das pessoas, com elevada probabilidade de ocorrerem conflitos [11].

As atividades relacionadas ao processo de Engenharia de Requisitos, apontadas por [2][12], e sendo considerado um conjunto de boas práticas são apresentadas a seguir.

2.1.1 Identificação de Requisitos

Na criação de um sistema, o processo de Engenharia de Requisitos pode ser utilizado para que as necessidades dos clientes sejam identificadas e utilizadas na forma de requisitos pelos desenvolvedores do sistema [10][12].

A primeira atividade deste processo é a Identificação dos requisitos do sistema, que por sua vez tem como subárea, conforme apresenta a Figura 2.1, o Estudo de Viabilidade.

Este é um estudo, direcionado a responder algumas perguntas:

- O sistema contribui para os objetivos gerais da organização?
- O sistema pode ser implementado com a utilização de tecnologia atual dentro das restrições de custo e prazo?
- O sistema pode ser integrado com outros sistemas já em operação?

A questão sobre se o sistema contribui ou não para os objetivos da organização é fundamental. Se um sistema não for compatível com esses objetivos, ele não terá nenhum valor real para a empresa. Embora possa parecer óbvio, muitas organizações desenvolvem sistemas, mas que não contribuem para seus objetivos, seja porque não existe uma declaração clara desses objetivos ou porque outros fatores políticos ou organizacionais influenciam na aquisição do sistema [13].

Preparar este tipo de estudo envolve avaliar e coletar informações e redigir relatórios. A fase de avaliação identifica as informações exigidas para responder às três perguntas apresentadas. Uma vez identificadas, é preciso questionar as fontes de informação, a fim de encontrar as respostas para estas perguntas [12][14].

Alguns exemplos de possíveis perguntas que podem ser feitas são:

- Como a organização se comportaria, se o sistema não fosse implementado?
- Quais são os problemas com os processos atuais e como um novo sistema ajudaria a diminuir estes problemas?
- Que contribuição direta o sistema trará para os objetivos da empresa?

- As informações podem ser transferidas para outros sistemas organizacionais e também podem ser recebidas à partir deles?
- O sistema requer tecnologia que não tenha sido utilizada anteriormente na organização?
- O que precisa e o que não precisa ser compatível com o sistema?

Entre as fontes de informação estão os gerentes de departamentos em que o sistema será utilizado, os engenheiros de software que estão familiarizados com o tipo de sistema proposto, peritos em tecnologia, usuários finais de sistemas, etc. Eles devem ser entrevistados, a fim de coletar as informações exigidas.

2.1.2 Análise de Requisitos

Após a atividade de Identificação de Requisitos, juntamente com o Estudo de Viabilidade, chega-se a Análise de Requisitos.

Nesta atividade, os membros da equipe de desenvolvimento trabalham com o cliente e usuários finais do sistema para descobrir mais informações sobre o domínio da aplicação, que serviços o sistema deve fornecer, o desempenho exigido do sistema, as restrições de hardware, etc.

A Análise de Requisitos pode envolver diferentes tipos de pessoas em uma organização. O termo *stakeholder* é utilizado para se referir a qualquer pessoa que terá alguma influência direta ou indireta sobre os requisitos do sistema. Dentre os *stakeholders* destacam-se: os usuários finais que interagirão com o sistema e todo o pessoal; os engenheiros que estão desenvolvendo o sistema ou fazendo a manutenção de outros sistemas relacionados; os gerentes de negócio; os representantes de sindicatos; entre outros, podem ser também considerados *stakeholders* do sistema.

Esta é uma atividade complicada, segundo [10][11][15], por diversas razões:

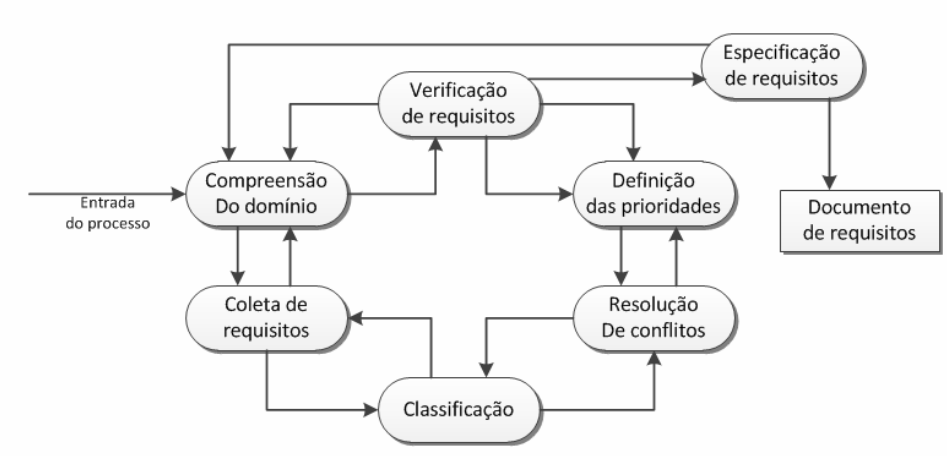
- Os *stakeholders* frequentemente não sabem na realidade o que querem do sistema, a não ser em termos muito gerais; eles podem achar difícil articular o que desejam do sistema; eles podem fazer pedidos não realistas, por não terem noção do custo de suas solicitações;
- Os *stakeholders*, em um sistema, expressam naturalmente os requisitos em seus próprios termos e com o conhecimento implícito de sua área de

atuação. Os ER que não tem experiência no domínio do cliente devem compreender esses requisitos;

- Diferentes *stakeholders* têm em mente diferentes requisitos e podem expressá-los de maneiras distintas. Os ER precisam descobrir todas as possíveis fontes de requisitos e encontrar os pontos comuns e os conflitos;
- Fatores políticos podem influenciar os requisitos do sistema. Eles podem provir de gerentes que definem requisitos específicos de sistema para aumentar sua influência na organização;
- O ambiente econômico e de negócios, no qual, a análise de requisitos ocorre, é dinâmico. Inevitavelmente, ele se modifica durante o processo de análise. Como consequência, a importância dos requisitos específicos pode mudar. Novos requisitos podem surgir por parte dos novos *stakeholders*, que não haviam sido consultados inicialmente.

Um modelo genérico do processo de análise de requisitos é mostrado na Figura 2.2. Cada organização tem sua própria versão, modelo, framework ou ainda uma versão mais definida desse modelo geral, dependendo de fatores locais, como a experiência da equipe, o tipo de sistema em desenvolvimento, os padrões utilizados, entre outros.

Figura 2.2 - Modelo da atividade de Análise



Fonte: Sommerville, 2007 [2]

As atividades do processo são:

- **Compreensão do domínio**: Os analistas devem desenvolver sua compreensão do domínio da aplicação. Por exemplo, se for exigido um sistema para um supermercado, o analista deverá descobrir como operam os supermercados.
- **Coleta de requisitos**: É o processo de interagir com os *stakeholders* do sistema para descobrir seus requisitos. A compreensão do domínio se desenvolve mais durante esta atividade.
- **Classificação**: Esta atividade considera o conjunto não estruturado dos requisitos e os organiza em grupos coerentes.
- **Resolução de conflitos**: Quando múltiplos *stakeholders* estão envolvidos, os requisitos apresentarão conflitos. Essa atividade se ocupa em encontrar e solucionar esses conflitos.
- **Definição das prioridades**: Em qualquer conjunto de requisitos, alguns serão mais importantes do que outros. Esse estágio envolve a interação com os *stakeholders*, para descobrir os requisitos mais importantes.
- **Verificação de requisitos**: Os requisitos são verificados, a fim de se descobrir se eles são completos e consistentes e se estão em concordância com o que os *stakeholders* realmente desejam do sistema.

Na Figura 2.2 percebe-se que a atividade de Análise de Requisitos é um processo iterativo, com *feedback* contínuo de cada atividade para as outras. O ciclo começa com a compreensão do domínio e termina com a verificação dos requisitos. A compreensão dos requisitos pelo analista melhora a cada fase do ciclo.

2.1.3 Especificação de Requisitos

Uma vez compreendidos e analisados, os requisitos devem ser documentados com um nível de detalhamento adequado, produzindo a especificação de requisitos de software. Pode ser utilizada a linguagem natural ou diagramas, como os propostos pela *Unified Modeling Language (UML)* [16].

Um dos diagramas que se encaixam perfeitamente na atividade de Especificação de Requisitos é o Diagrama de Caso de Uso.

Os casos de uso são técnicas para obtenção de requisitos, eles se tornaram atualmente uma característica fundamental da UML para descrever modelos de sistemas

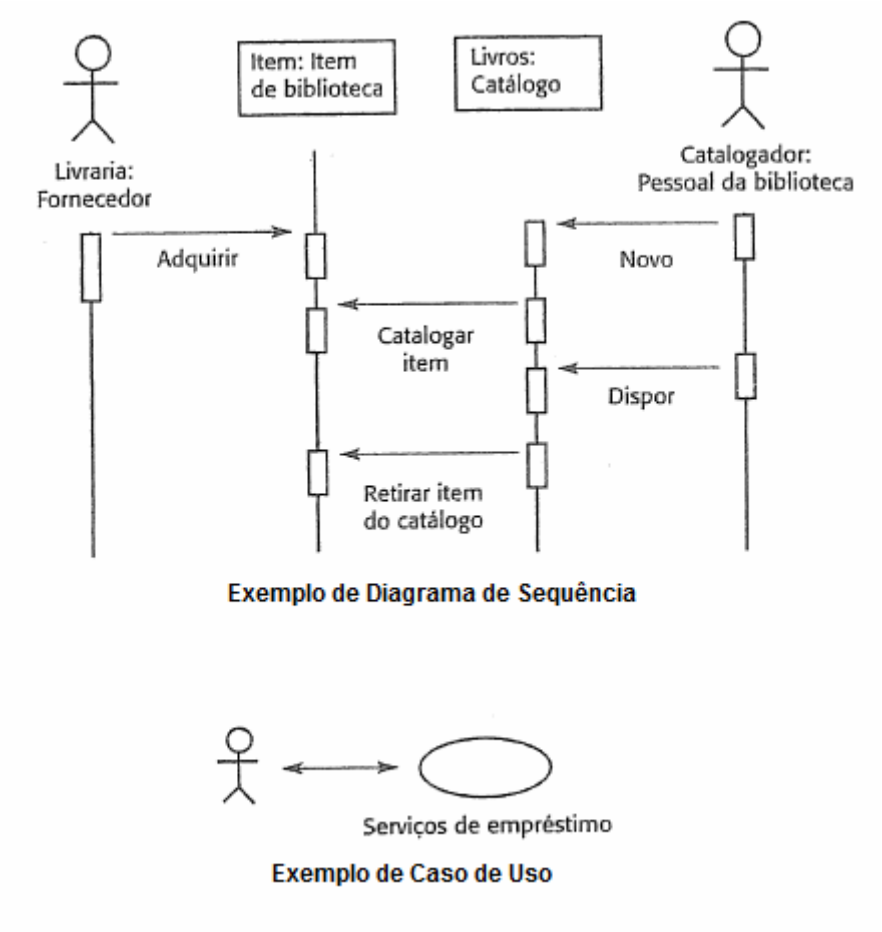
orientados à objetos, na sua forma mais simples, um caso de uso identifica os agentes envolvidos em uma interação e especifica o tipo de interação.

Os agentes no processo são representados por bonecos e cada classe de interação é representada por uma elipse com um nome. O conjunto de casos de uso representa todas as possíveis interações que serão representadas nos requisitos do sistema.

Dentro da UML, Diagramas de Sequência podem ser utilizados para acrescentar informações a um caso de uso. Esses diagramas de sequência mostram os agentes envolvidos na interação, os objetos dentro do sistema com os quais eles interagem e as operações que estão associadas a estes objetos.

A Figura 2.3 apresenta um exemplo do Diagrama de Sequência, juntamente com um exemplo de um Caso de Uso.

Figura 2.3 - Exemplos de Especificação



Fonte: adaptado de Sommerville, 2007 [2] e Booch, 2005 [16]

A UML é um padrão para a modelagem orientada a objetos; assim os casos de uso e a obtenção de requisitos com base neles são cada vez mais utilizados em desenvolvimento de sistemas [2][11].

2.1.4 Validação de Requisitos

A Validação de Requisitos mostra que os requisitos realmente definem o sistema que o cliente deseja [17]. Possui uma abordagem muito semelhante com a Análise de Requisitos, uma vez que se preocupa em descobrir problemas nos requisitos. Contudo esses são processos distintos, já que a validação elabora um esboço completo do documento de requisitos, enquanto a análise trabalha com requisitos incompletos [10][17].

A validação de requisitos é importante, pois a ocorrência de erros num documento de requisitos pode levar a grandes custos relacionados ao retrabalho, quando esses erros são descobertos durante o desenvolvimento ou após a entrega do sistema [1][2].

O custo para realizar uma modificação no sistema, resultante de um problema de requisito, é muito maior do que reparar erros de projeto ou de codificação [3]. A razão disso é que uma mudança nos requisitos, em geral, significa que o projeto do sistema e a implementação também devem ser modificados e que o sistema necessita ser testado novamente.

Durante o processo de Validação de Requisitos, diferentes tipos de verificação devem ser realizados sobre os requisitos no documento de requisitos [17]. Dentre estas verificações destacam-se:

- **Verificação de validade**: Um usuário pode pensar que um sistema é necessário para realizar certas funções. Contudo, mais estudos e análises podem identificar funções adicionais ou diferentes, que são exigidas. Os sistemas têm diversos usuários com necessidades diferentes e qualquer conjunto de requisitos é inevitavelmente uma solução conciliatória da comunidade de usuários;
- **Verificações de consistência**: Os requisitos em um documento não devem ser conflitantes, ou seja, não devem existir restrições contraditórias ou descrições diferentes para uma mesma função do sistema;

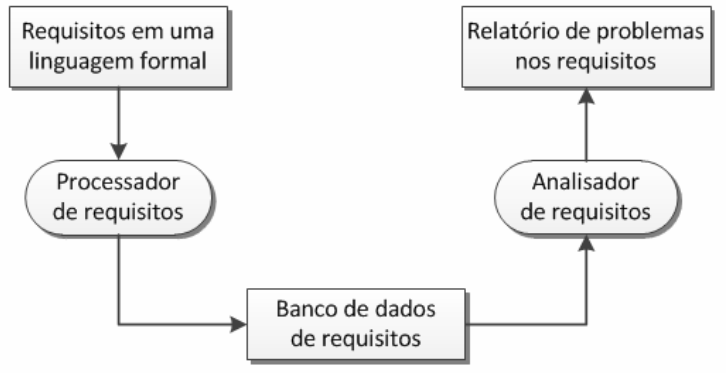
- **Verificações de completeza**: O documento de requisitos deve incluir requisitos que definam todas as funções e restrições exigidas pelo usuário do sistema;
- **Verificações de realismo**: Utilizando o conhecimento da tecnologia existente, os requisitos devem ser verificados, a fim de assegurar que eles realmente podem ser implementados. Essas verificações devem também levar em conta o orçamento e os prazos para o desenvolvimento do sistema;
- **Facilidade de verificação**: Para reduzir o potencial de divergências entre cliente e fornecedor, os requisitos do sistema devem sempre ser escritos de modo que possam ser verificados. Isso significa que um conjunto de verificações pode ser projetado para mostrar que o sistema entregue cumpre com estes requisitos.

Existe uma série de técnicas de validação de requisitos que podem ser utilizadas em conjunto ou individualmente. Segundo [2][17[[18], algumas destas técnicas são:

- **Revisões de requisitos**: Os requisitos são analisados sistematicamente por uma equipe de revisores;
- **Prototipação**: Nessa abordagem de validação, um modelo executável do sistema é mostrado aos usuários finais e clientes. Eles podem experimentar o modelo para verificar se ele atende às suas necessidades;
- **Geração de casos de testes**: Como modelo ideal, os requisitos deveriam ser testáveis. Se os testes para requisitos são criados como parte do processo de validação, isso, muitas vezes, revela problemas com os requisitos. Se um teste é difícil ou impossível de ser projetado, isso frequentemente significa que os requisitos serão de difícil implementação e devem ser reconsiderados;
- **Análise automatizada de consistência**: Se os requisitos são expressos como um modelo de sistema em uma notação estruturada ou formal, então as ferramentas CASE podem ser utilizadas para verificar a consistência do modelo. A Figura 2.4 ilustra esta análise. Para verificar a consistência, a ferramenta CASE deve construir uma base de dados de requisitos e, então, utilizando-se as regras do método ou da notação,

verificar todos os requisitos na base de dados. Uma análise de requisitos pode produzir um relatório das inconsistências que foram descobertas.

Figura 2.4 - Análise de Consistência



Fonte: Sommerville, 2007 [2]

As dificuldades de validação de requisitos não devem ser subestimadas [17]. É difícil demonstrar que um conjunto de requisitos atende às necessidades de um usuário. Os usuários devem pensar no sistema em operação e imaginar como este sistema se adequaria ao seu trabalho. Este não é um tipo de análise fácil, mesmo para profissionais de computação habilitados, tornando-se mais difícil ainda para os usuários do sistema.

Como resultado, a validação de requisitos raramente descobre todos os problemas com os requisitos, e as modificações para corrigir omissões e falhas de compreensão, depois que o documento de requisitos foi aceito, são inevitáveis.

2.1.4.1 Revisões de requisitos

Uma revisão de requisitos é um processo manual, que envolve muitos leitores, tanto do pessoal do cliente como dos desenvolvedores, que verificam o documento de requisitos a fim de detectar inconsistências ou omissões [18].

O processo de revisão pode ser gerenciado, ou organizado, em maior escala com muitos participantes envolvidos na verificação de diferentes partes do documento.

As revisões de requisitos podem ser informais ou formais. As revisões informais simplesmente os desenvolvedores que discutem os requisitos com tantos *stakeholders* quantos forem possíveis [2][17]. Em muitos casos a comunicação entre desenvolvedores e *stakeholders* termina depois da obtenção de requisitos, e não existe

nenhuma confirmação de que os requisitos documentados são os que os *stakeholders* realmente solicitaram.

Em uma revisão formal de requisitos, a equipe de desenvolvimento deve "conduzir" o cliente pelos requisitos do sistema, explicando as implicações de cada um. A equipe de revisão deve verificar cada requisito, em termos de sua consistência, e checar os requisitos como um todo, sob o ponto de vista de sua completeza.

Os revisores podem também checar:

- **Facilidade de verificação**: O requisito é realmente passível de ser testado, como foi definido?
- **Facilidade de compreensão**: O requisito pode ser adequadamente compreendido pelos clientes e usuários finais do sistema?
- **Facilidade de rastreamento**: A origem do requisito é claramente definida? Pode ser preciso retornar a origem do requisito, a fim de avaliar o impacto de uma mudança. A facilidade de rastreamento é importante porque permite avaliar o impacto de uma mudança no restante do sistema.
- **Adaptabilidade**: O requisito é adaptável? Em outras palavras, ele pode ser modificado, sem que isso provoque efeitos em grande escala em outros requisitos do sistema?

Conflitos, contradições, erros e inconsistências nos requisitos devem ser destacados durante a revisão e formalmente registrados. Fica, então, por conta dos usuários, clientes e desenvolvedores do sistema negociar a solução para estes problemas identificados.

2.2 Reuso de Requisitos

O Reuso de Requisitos é sugerido como uma das maneiras de alcançar a melhoria do desenvolvimento de software e principalmente a qualidade dos requisitos.

Quando desenvolvemos requisitos para um novo sistema, deve-se, na medida do possível, reutilizar os requisitos de outros sistemas que foram desenvolvidos para a mesma área da aplicação [1][2]. Complementando esta ideia, [19] cita que ao reusarmos requisitos, fazemos também a reutilização do conhecimento adquirido em experiências anteriores para melhorar as futuras especificações.

A reutilização de requisitos oferece às organizações a capacidade única de compartilhar uma exigência em um projeto, sem absorver a duplicação desnecessária de

artefatos dentro de um repositório. Esta é uma capacidade fundamental que acelera o tempo de entrega do software para o mercado e reduz os custos de desenvolvimento.

Requisitos compartilhados podem acompanhar as mudanças feitas pelos clientes, como também podem permanecer estáticos, se forem às necessidades do projeto [20]. Além disso, a mudança de um requisito comum pode ser feita por qualquer pessoa, e o sistema deve estar preparado para processar a ramificação e evolução desse requisito apropriadamente.

O conceito de reutilização é uma noção familiar dentro do desenvolvimento de software, mas menos comum quando considerado no campo da gestão de requisitos. Existem várias definições, que devem ser levadas em consideração na implementação de uma solução para lidar com a reutilização de requisitos [19][20].

Assim, os elementos que compõem um requisito e estabelece o entendimento comum de como os requisitos devem evoluir, como essa evolução é mantida, e como as organizações podem reutilizar requisitos para acelerar a inovação de negócios, devem ser analisados de maneira a reduzir a complexidade e custos deste controle e gerenciamento.

2.2.1 Gerenciando o Reuso de Requisitos

Para entendermos o conceito de reutilização de requisitos [21][22], devemos primeiro olhar para as várias partes de um requisito: dados, metadados e relacionamentos.

1. **Dados**: Descreve um objeto, e é relevante para o objeto em si. Um exemplo pode ser um resumo ou descrição de um requisito.
2. **Metadados**: São dados sobre os dados, ajudando na organização e utilizando o objeto dentro de um processo. Normalmente descreve o estado atual do objeto, e tem o mesmo âmbito que os dados em si. Por exemplo, os metadados podem descrever o estado/estágio, dentro de um fluxo de trabalho do requisito (ou seja, aprovado, rejeitado, satisfeito, e testado).
3. **Relacionamentos**: Esta característica de um requisito permite modelar sua(s):
   - estrutura (ou seja, “o requisito consiste em...”, “o requisito inclui...”);
   - história (ou seja, “o requisito é uma revisão dos...”, “o requisito é derivado de...”);

- ligações conceituais ou traços (ou seja, “o requisito satisfaz...”);
- referências (ou seja, “o requisito é definido por...”, “o requisito decompõe-se em...”);
- segurança (ou seja, “o requisito é autorizado para/por...”).

Qualquer requisito pode oferecer informações em cada uma das categorias de dados, metadados e relacionamentos. Quando os requisitos são reutilizados, qualquer ou toda a informação também pode ser reutilizado [21].

Uma ferramenta de gestão de requisitos escolhida pela organização precisa ter uma arquitetura e também capacidades do usuário para suportar o nível estratégico de reutilização ditada pelas necessidades da organização. Como a reutilização pode ocorrer em diferentes níveis, aproveitando os elementos de metadados, de dados e relacionamento de um requisito, a flexibilidade também é fundamental para a solução do desafio de reutilização.

Isso muitas vezes não ocorre de forma sistemática [20], uma vez que as dependências dos requisitos não são explicitamente modeladas na maioria dos casos. Como consequência, quando um requisito é reutilizado, alguns artefatos relacionados a este requisito (partes interessadas, casos de teste, etc.) não podem ser devidamente reutilizados [23].

2.2.2 História, versões e linhas de base de requisitos

Ao implementar um cenário de reutilização complexo, ou mesmo um sistema onde os requisitos persistem, mesmo após serem lançadas várias versões do sistema, deve-se ter a capacidade de identificar pontos importantes na evolução do requisito.

No desenvolvimento de sistemas, esses pontos significativos são chamados de "versões". Este termo pode obter significados diferentes para pessoas diferentes, na definição de "versão" aplicada à reutilização de requisitos, pretende-se mostrar como ele se relaciona com termos semelhantes, como história, referências e marcos [15].

Considerando um sistema onde os requisitos são especificados dentro de documentos de requisitos, mas são armazenados como itens individuais dentro do repositório. Segundo [15], teremos os itens que representam estes requisitos como:

- **História**: termo usado para descrever a trilha de verificações e validações para um item individual ou requisito. Todas as alterações feitas para o requisito: dados, metadados, ou seus relacionamentos são

capturados em sua história. A história apresenta as respostas para “o que”, “quando” e “quais” com respeito a alterações no requisito;

- **Versão**: representa um ponto significativo na história um requisito individual. Nem todas as alterações em um requisito são significativas e garantem uma nova versão. Por exemplo, na mudança de um requisito de “Pedro” para “Paulo” não seria algo necessário para se criar um identificador de versão específica. A alteração é registrada para a história do item, mas uma nova versão não é criada;
- ***Baseline***: conceito muito semelhante à versão, mas tem um alcance muito diferente. Requisitos são muitas vezes organizados em grupos ou conjuntos. No domínio da gestão de requisitos esses conjuntos são chamados de documentos e uma *baseline* torna-se um ponto significativo na história de um documento.

Algumas organizações usam uma definição um pouco diferente para *baseline*. Ao invés de ser um ponto instantâneo no tempo para um determinado documento, uma *baseline*, definida aqui no contexto de reutilização de requisitos, é uma meta para se alcançar.

A Gestão de requisitos reivindica permitir a versão de requisitos individuais. Muitas ferramentas com suporte a versionamento por meio de clonagem ou pela copia do requisito inteiro. Alguns trabalhos sugerem também relacionar a cópia ao requisito original.

Podemos dizer que o relacionamento, o versionamento e a reutilização não são os mesmos itens dentro do Gerenciamento de Requisitos (GRE). Os conceitos de controle de versão são muitas vezes confundidos com o de reutilização. A próxima seção apresenta cenários de reutilização, segundo [13][15], para ilustrar as diferenças e os benefícios de versões e de reutilização.

2.2.3 Cenários de Reutilização de Requisitos

Cenário 1: Reutilização de requisitos sem reutilização - Compartilhamento

A capacidade de compartilhar um item entre os projetos, documentos ou outros artefatos de trabalho pode ser considerada uma forma de reutilização. Sob esta definição, todos os projetos que estão compartilhando o requisito poderão “vê-lo”, e

eventualmente, contribuir para a sua evolução. Os metadados sobre o requisito serão compartilhados como também todas as relações e os dados.

Este não é um cenário de reutilização real, pois o requisito em si está sendo compartilhado em sua completude.

Cenário 2: Reutilização de requisitos sem herança - Cópia

Como mencionado anteriormente, a cópia de um requisito a partir de um lugar para outro também pode ser considerada uma forma de reutilização. Na verdade, esta é a forma de reutilização que várias ferramentas que não apresentam gerenciamento de requisitos suportam.

Quando um analista abre um documento, seleciona algum conteúdo e executa um gesto de copiar/colar em outro documento, está utilizando a reutilização do conteúdo para um novo propósito. Esta forma de reutilização não tem conhecimento de herança ou "de onde veio" e as mudanças em um documento não têm impacto sobre as mudanças no outro. Na verdade, as mudanças são completamente independentes e um documento não tem conhecimento de que a mudança ocorreu no outro, muito menos o que a mudança poderia ter sido.

Este também não pode ser considerado um cenário de reutilização real. A reutilização deve, mesmo que minimamente, incluir uma indicação de onde o conteúdo original veio.

Cenário 3: Reutilização de requisitos com herança

Considerando os dois primeiros cenários, podemos supor que seja possível responder a pergunta "de onde veio", deixando a cópia do requisito com um ponteiro de volta à sua origem oferecendo várias opções para reutilização. É a forma em que esta ligação é feita que irá diferenciar cada um dos modelos de reutilização seguintes.

A maioria das ferramentas de Gerenciamento de Requisitos disponíveis hoje utilizam links ou relações, se não no nível de requisito individual, no nível de documento. Ligações no nível de documento são melhores do que não se apresentar nenhuma ligação, mas são muito pouco poderosas. Em longo prazo, essas ligações não respondem à pergunta de rastreabilidade com detalhes suficientes para serem significativas.

Tendo uma ligação à origem de um requisito, podemos apontar este cenário como o início de reutilização real, embora ainda muito primitivo.

Cenário 4: Reutilização de requisitos com notificação de alteração

Neste cenário, um requisito e todas as informações (dados, metadados e relacionamentos), são reutilizados na sua totalidade. O estado do projeto determina o estado dos requisitos no momento da reutilização, e qualquer mudança em um requisito num cenário de reutilização provoca um efeito cascata, sinalizando todos os outros artefatos relacionados a esse requisito com uma marcação de verificação.

Cenário 5: Reutilização de requisitos com controle de mudança

Reutilização de requisitos com controle de mudanças é semelhante a reutilização com notificação de alteração, quando os dados, metadados e os relacionamentos são reutilizados em sua totalidade.

Este é o mesmo cenário que o cenário 1 discutido acima, no entanto, há uma diferença significativa, os dois projetos que compartilham o mesmo requisito apenas compartilham-no até o momento em que um projeto precisa muda-lo. Quando a informação muda uma nova versão é criada e apenas os itens que fazem referência a esta nova versão são sinalizados para serem verificados. Todos os outros projetos ou documentos não são afetados.

Cenário 6: Reutilização de requisitos com anotações

Nos dois últimos cenários de reutilização acima, os requisitos e informações relacionadas (dados, metadados e relações) são reutilizados na sua totalidade. Aqui, na reutilização com anotações, apenas algumas das informações pertencentes a um requisito são identificadas como candidatas para compartilhamento e reutilização. O restante da informação é específico para o projeto ou documento.

A informação compartilhada entre os requisitos é mantida no repositório de informações, enquanto as outras informações continuam pertencendo ao projeto ou fazendo referência ao documento.

Cada instância do requisito a ser reutilizado tem seus próprios metadados e relacionamentos. O projeto ou documento pode ser independente do estado dos requisitos que

estão contidos no seu interior. Novas versões do requisito são criadas automaticamente quando as informações compartilhadas no repositório são alteradas. Essas mudanças podem desencadear novas revisões e podem ainda sinalizar outras referências, bem como outros itens no sistema, pelo efeito cascata dessa mudança. Por exemplo, mudanças em alguns requisitos podem afetar casos de teste ou especificações funcionais já criadas.

Assim que o projeto ou documento ganha independência em termos de metadados, tem-se a capacidade de modelar tanto a parte dinâmica (compartilhamento) como a parte estática (reutilização) do requisito ao mesmo tempo.

O gerente de projeto ou analista deve decidir se querem manter a coerência do requisito com a necessidade do cliente evoluindo-o de forma dinâmica ou se deseja bloquear a necessidade, de tal forma que o impacto da mudança não afete seu projeto.

Cenário 7: Reutilização de requisitos com anotações e gerenciamento de mudanças

Aplicando mudanças e paradigmas de gerenciamento de configuração para a disciplina de gerenciamento de requisitos pode-se criar uma única solução integrada e rastreável trazendo o poder de reutilização para um novo nível.

Ao incorporar um processo em cima de reutilização e controlar como e quando os requisitos podem ser modificados e reutilizados permite colher benefícios sem desnecessariamente criar ramificações e versionamentos de objetos, a menos que seja autorizado e apropriado fazê-lo.

Os Pedidos de Mudança nos requisitos serão filtrados e dirigidos à conselhos de revisão diferentes. Alguns destes pedidos serão aprovados e atribuídos a desenvolvedores que efetuarão as alterações solicitadas. Idealmente, esse processo de gestão da mudança pode definir que tipos de mudanças podem ser feitas: modificação, ramificação, aplicação de uma *baseline* entre outros.

Apenas quando um pedido aprovado é associado com um requisito, um analista pode modificar o requisito, fazendo com que o sistema esteja corretamente versionado, em conformidade, e notificando os componentes relacionados de forma adequada.

No projeto de um software deve-se pensar em soluções que eliminem custos e tempo na elicitação de requisitos. O reuso de requisitos busca maior agilidade nas fases de Análise e Especificação de Requisitos [22], consequentemente menor tempo para produção do software e menor tempo de entrada no mercado.

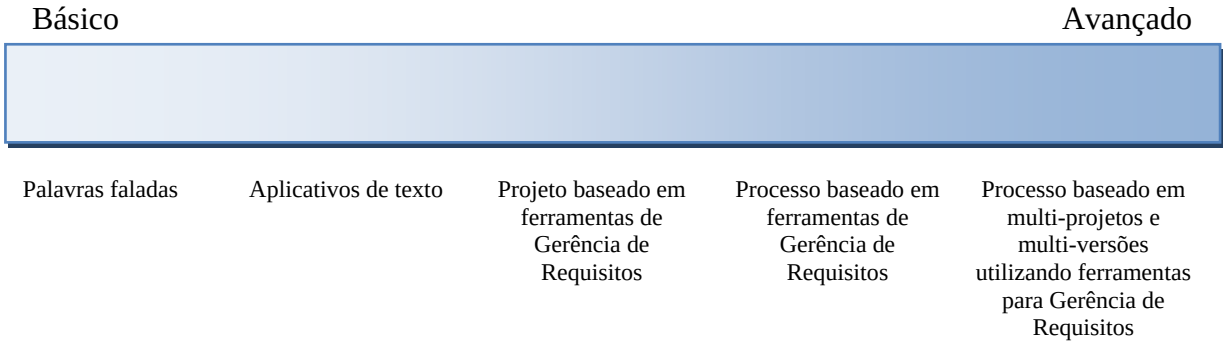
Existem ainda diversos cenários, ou modelos de reutilização, adicionais que não estão aqui descritos: reutilização em nível de componente, reutilização de documentos e várias combinações destes com anotações e gerenciamento de mudanças, por exemplo. As necessidades de negócio e metas estratégicas como um todo irão determinar qual o modelo é mais eficaz para o projeto ou organização.

2.2.4 Reutilização de Requisitos dentro de uma organização

Muitas empresas ainda estão na infância do Gerenciamento de Requisitos. Ainda não adotaram uma ferramenta para GRE, e atualmente estão usando aplicativos de produtividade empresarial, como o *Word* ou o *Excel* da *Microsoft* para especificar e acompanhar requisitos. Estas empresas olham somente para as capacidades, tais como a facilidade de importação de documentos, suporte de texto, e rastreabilidade que facilitam a aprovação do negócio. O que ainda não percebem, é que estão em um ponto de sofisticação onde é necessário que os requisitos obtenham apoio para a reutilização.

Há um amplo campo de necessidades para ferramentas de gestão de requisitos no mercado, e as organizações precisam primeiro saber onde elas estão na curva de maturidade requisitos [24]. A Figura 2.5 apresenta um exemplo de curva de maturidade para o Gerenciamento de Requisitos.

Figura 2.5 - Curva de Maturidade de GRE



Fonte: adaptado de Davis, 2005 [24]

A curva de maturidade requisitos não é realmente uma curva, mas uma medição do processo atual e as ferramentas utilizadas e/ou necessárias dentro de uma organização quando se trata de gerenciamento de requisitos. Como as organizações evoluem ao longo da curva, a necessidade de mais recursos, tais como: gestão de mudanças, de

processos, fluxo de trabalho, rastreabilidade e reutilização aumentam dentro de seu quadro de gestão de requisitos.

No entanto, se uma organização tem progredido na curva de maturidade com relação ao gerenciamento de requisitos, e está gerenciando vários projetos e milhares de requisitos em paralelo, buscando reduzir a complexidade, menor custo de desenvolvimento, e encurtar os ciclos de inovação, então a reutilização de requisitos é um conceito que deve ser trabalhado [25].

Independentemente de onde uma organização cai na curva, reutilizar os requisitos em sua forma mais básica trará um impulso à produtividade. Em vez de reescrita de requisitos, copiá-los e modificá-los para as necessidades de cada projeto, a estrutura destes requisitos será gerenciada utilizando a experiência adquirida melhorando a especificação e consequentemente aumento a qualidade.

Dentro de qualquer organização, a necessidade de padronizar e simplificar o acesso ao sistema em desenvolvimento existe, e reutilizar os requisitos de um projeto para outro, fornecendo este tipo de funcionalidade pode trazer ótimos benefícios [24][25].

Em qualquer caso, é necessário concentrar-se no domínio do problema antes de chegar à questão da reutilização de requisitos [26]. Algumas perguntas podem ser utilizadas para definir se uma organização pode inferir que a reutilização é um conceito à ser aproveitado, e se for, qual tipo de reutilização é mais adequado à necessidade. Abaixo são apresentados alguns exemplos de perguntas que podem ser aplicadas com esta finalidade.

1. Como é feita a captura de todos os requisitos de cliente e de negócio? (Criação)
2. Como a equipe de projeto utiliza o trabalho já criado em outros projetos? (Reutilização)
3. Como é feita a verificação de que cada mudança feita para sistemas de software acompanha diretamente o requisito de negócio? (Rastreabilidade)
4. Como as variações do produto são rastreadas? (Reutilização)
5. Existem requisitos alterados, removidos ou adicionados no último mês? Como é feito este controle? (Gestão de mudanças)
6. Como avaliar o impacto da alteração de requisitos nos controles de custo e prazo do sistema? (Análise de Impacto)
7. Como avaliar o efeito sobre o compartilhamento de requisitos entre projetos ao reutilizar requisitos? (Reutilização)

A maioria das organizações terá dificuldades em responder mais de uma das perguntas acima, assim ao utilizar um processo de especificação e gestão de requisitos bem definido com o apoio de uma ferramenta que incorpora a criação efetiva, a rastreabilidade, gestão de mudanças e reutilização, ao mesmo tempo proporcionando semelhantes capacidades para gerenciar as atividades de gestão de teste e mudança e gerenciamento de configuração permitirá maior controle e agilidade sobre os projetos de software.

Reutilização não pode ser parte da estratégia de curto prazo, mas deve-se pensar com um investimento no futuro, o que significa que a utilização de um processo ou uma ferramenta irá crescer com a organização, à medida que amadurece ao longo do tempo.

O Reuso de requisitos pode, portanto, proporcionar ganhos significativos de produtividade no desenvolvimento e na qualidade dos produtos de software resultantes [26]. Além disso, a reutilização de requisitos tem sido apontada como uma necessidade urgente e também um grande desafio na área de pesquisa e prática de Engenharia de Software.

2.3 Garantia da Qualidade em Requisitos

A garantia da qualidade é extremamente importante quando os requisitos são especificados dentro do desenvolvimento de software. Nesta área existem algumas normas internacionais como a IEEE 830-1998 [6] e 1233a-1998 [7] que provém boas práticas para garantir que os requisitos especificados obtenham maior nível de qualidade.

Desta maneira, 8 propriedades importantes, apontadas pelas normas citadas, que demonstram uma boa especificação de requisitos são apresentadas a seguir.

2.3.1 Propriedade 1: Correta

Uma especificação de requisitos é considerada correta se cada requisito expresso nela são requisitos que o software deva atender. Não há nenhuma ferramenta ou procedimento que garante este item.

A especificação de requisitos deve ser comparada com qualquer especificação aplicável superior, tal como uma especificação de requisitos de sistema, com a documentação de outro projeto, e com outras normas aplicáveis, garantindo que todos estes itens estejam em concordância.

Alternativamente, o cliente pode determinar se a especificação reflete corretamente suas necessidades reais. A rastreabilidade torna este procedimento mais fácil e menos propenso a erros.

2.3.2 Propriedade 2: Sem Ambiguidades

Uma especificação de requisitos é isenta de ambiguidades se cada requisito expresso nela tem apenas uma interpretação. No mínimo, isto requer que cada característica do produto final seja descrito usando um termo único. Nos casos em que um termo utilizado, num contexto particular, tenha múltiplos significados, este termo deve ser incluído em um glossário onde o seu significado é determinando de maneira mais específica.

A especificação de requisitos é uma parte importante do ciclo de vida do software e é usada na implementação, monitoramento, verificação e validação. Assim, a especificação deve estar livre de ambiguidades, tanto para aqueles que criam quanto para aqueles que a usam. No entanto, estes grupos muitas vezes não têm o mesmo conhecimento e, portanto, não descrevem os requisitos de software da mesma maneira. Algumas representações que melhoram a especificação de requisitos para o desenvolvedor podem ser ruins, na medida em que diminuem a compreensão para o usuário e vice-versa.

Os itens 2.3.2.1 ao 2.3.2.3 apresentam recomendações de como evitar ambiguidades nos requisitos de software.

2.3.2.1 Armadilhas da linguagem natural

Requisitos são geralmente escritos em linguagem natural (por exemplo, em Português ou Inglês). A linguagem natural é inerentemente ambígua. Uma especificação de requisitos escrita em linguagem natural deve ser revista e validada para que seja identificado o uso ambíguo da linguagem e assim possa ser corrigido.

2.3.2.2 Linguagens de especificação de requisitos

Uma maneira de evitar a ambiguidade inerente à linguagem natural é escrever a especificação em um determinado idioma de especificação de requisitos. Assim, os processadores de idiomas automaticamente detectam muitos erros lexicais, sintáticos e semânticos.

Uma desvantagem na utilização de tais idiomas é o tempo necessário para aprendê-los. Além disso, muitos usuários não técnicos terão dificuldades em entender a linguagem. Outro ponto importante é que essas linguagens tendem a ser melhores em expressar certos tipos de necessidades e resolver certos tipos de sistemas. Assim, eles podem influenciar os requisitos de maneiras sutis.

2.3.2.3 Ferramentas de representação

Em geral, os métodos de requisitos, linguagens e as ferramentas que lhes dão suporte são classificados em três categorias gerais.

1. **Abordagens orientadas a objetos** organizam os requisitos em termos de objetos do mundo real, seus atributos e os serviços realizados por esses objetos;
2. **Abordagens baseadas em processos** organizam os requisitos em hierarquias de funções que se comunicam através de fluxos de dados;
3. **Abordagens comportamentais** descrevem o comportamento externo do sistema em termos de alguma noção abstrata (como cálculo de predicados), funções matemáticas, ou máquinas de estado.

O grau em que tais ferramentas e métodos podem ser úteis na preparação de uma especificação de requisitos depende do tamanho e complexidade do sistema esperado. Existem abordagens como [27][28] que também tratam da questão de ambiguidade em requisitos de maneiras sistemáticas aplicadas ao desenvolvimento de software.

A melhor forma de usar qualquer uma dessas abordagens é manter as descrições em linguagem natural. Dessa forma, os clientes que não estão familiarizados com as notações podem entender a especificação.

2.3.3 Propriedade 3: Completa

Uma especificação de requisitos é completa se inclui os seguintes elementos:

- Todos os requisitos devem ser significativos, seja em matéria de funcionalidade, desempenho, restrições de design, atributos ou interfaces externas. Em particular, quaisquer exigências externas impostas por um sistema ou especificação devem ser reconhecidas e tratadas.

- Existe uma definição das respostas do software para todas as classes realizáveis de dados de entrada em todas as situações. É de extrema importância especificar as respostas para os valores de entrada válidos e inválidos.
- Devem estar definidos os rótulos completos e referências a todas as figuras, tabelas e diagramas na especificação.

2.3.3.1 Utilização de ASDs

Qualquer especificação que use a frase "a ser determinado" (ASD) não é uma especificação de requisitos completa. O ASD é, no entanto, por vezes, necessário e deve ser acompanhado por:

- Uma descrição das condições que causam o ASD (por exemplo, uma resposta não é conhecida), de modo que a situação possa ser resolvida;
- A descrição do que deve ser feito para eliminar o ASD, o que é responsável pela sua eliminação, e quando ele deve ser eliminado.

2.3.4 Propriedade 4: Consistente

Consistência refere-se à consistência interna. Se uma especificação de requisitos de software não está de acordo com algum documento de nível superior, como uma especificação de requisitos de sistema, então ela não é considerada consistente.

Uma especificação é internamente consistente se nenhum subconjunto de requisitos individuais nela descritos estão em conflito. Os três tipos de conflitos prováveis em um *Software Requirement Specification* (SRS) são:

- As características específicas de objetos do mundo real podem entrar em conflito. Por exemplo, um requisito pode afirmar que todas as luzes devem ser verdes enquanto outro pode afirmar que todas as luzes devem ser azuis.
- Pode haver um conflito lógico ou temporal entre duas ações. Por exemplo, um requisito pode especificar que o programa irá adicionar duas entradas e outro pode especificar que irá multiplicá-los.

- Dois ou mais requisitos podem descrever o mesmo objeto do mundo real, mas usam termos diferentes para aquele objeto. Por exemplo, uma entrada do usuário pode ser chamada de “imediata” em um requisito e “dependente” em outro. O uso de terminologia padrão e definições promovem a consistência.

2.3.5 Propriedade 5: Classificada por importância e/ou estabilidade

Uma especificação de requisitos de software é classificada por importância e/ou estabilidade se cada exigência tem um identificador para importância ou estabilidade de um requisito em particular. Tipicamente, todos os requisitos que se relacionam com um produto de software não são igualmente importantes. Alguns requisitos podem ser essenciais, especialmente para aplicações críticas, enquanto outros podem ser desejáveis.

Cada requisito deve ser identificado para que essas diferenças sejam claras e explícitas [12][24]. Identificar os requisitos dessa forma ajuda o cliente, fazendo com que ele seja mais cuidadoso considerando para cada necessidade, e ajuda também os desenvolvedores a tomar decisões de projeto corretas e dedicar níveis adequados de esforço para as diferentes partes do produto de software.

2.3.5.1 Grau de estabilidade

Um método de identificação de requisitos usa a dimensão de estabilidade. A estabilidade pode ser expressa em termos do número de mudanças esperadas para qualquer exigência, com base na experiência ou conhecimento dos eventos, que afetam a organização, as funções, e as pessoas suportadas pelo sistema de software.

2.3.5.2 Grau de necessidade

Outra forma de classificar os requisitos é distinguir classes de requisitos como essencial, condicional e opcional.

1. **Essencial**: Implica que o software não será aceitável, a menos que estes requisitos sejam fornecidos da maneira acordada.
2. **Condicional**: implica que estes são requisitos que melhoram o produto de software, mas não o tornam inaceitável se estiverem ausentes.

3. **Opcional**: implica uma classe de funções que podem ou não podem ser benéficos.

2.3.6 Propriedade 6: Verificável

Uma especificação de requisitos de software é verificável se cada requisito expresso nela é verificável. Isto também é apontado por [22][28], determinando que, um requisito é verificável se existe algum processo finito com o qual uma pessoa ou máquina pode verificar que o produto de software atende a exigência.

Em geral, **qualquer requisito ambíguo não é verificável**. Requisitos não verificáveis incluem declarações como "funciona bem", "boa interface humana", e "normalmente acontece". Estes requisitos não podem ser verificados porque é impossível definir os termos "bem", "boa" ou "normalmente".

Uma afirmação do tipo "o programa não deverá entrar em loop infinito" é não verificável também porque o teste desta qualidade é teoricamente impossível.

Um exemplo de um requisito verificável poderia ser:

“A Saída do programa deve ser produzido dentro de 20 s depois do evento em 60% do tempo, e deve ser apresentada dentro de 30 s depois do evento em 100% do tempo.”

Esta afirmação pode ser verificada porque ele usa termos concretos e quantidades mensuráveis. Se um método não pode ser concebido para determinar se o requisito atende a uma necessidade específica, então este requisito deve ser removido ou revisto.

2.3.7 Propriedade 7: Modificável

Uma especificação de requisitos de software é modificável se, e somente se, sua estrutura e estilo são tais que, quaisquer mudanças nos requisitos podem ser feitas facilmente, completa e consistente, mantendo a estrutura e estilo. Isso geralmente requer:

- Ter uma organização coerente e fácil de usar com uma tabela de conteúdos, um índice e referências explícitas;
- Não ser redundante (ou seja, o mesmo requisito não deve aparecer em mais de um lugar na especificação);

- Cada requisito é expresso separadamente, ao invés de misturados com outros requisitos.

Importante destacar que **a redundância em si não é um erro**, mas pode facilmente levar a erros.

Redundância pode ocasionalmente ajudar a fazer uma especificação mais legível, mas um problema pode surgir quando o documento redundante é atualizado. Por exemplo, um requisito pode ser alterado em somente um dos lugares onde ele aparece. A especificação, em seguida, torna-se inconsistente. Sempre que a redundância é necessária, a especificação deve incluir explícitas referências cruzadas para torná-la modificável.

2.3.8 Propriedade 8: Rastreável

Uma especificação de requisitos é rastreável se a origem de cada um de seus requisitos é clara e facilita a referência de cada requisito no desenvolvimento ou documentação futura. Os dois tipos de rastreabilidade recomendadas são:

a) **Rastreabilidade para trás (*Backward*)**: às fases anteriores de desenvolvimento. Isso depende de cada requisito e sua referência explícita a fonte em documentos anteriores.

b) **Rastreabilidade para frente (*Forward*)**: a todos os documentos gerados pela especificação. Isso depende de cada requisito ter um único nome ou número de referência. A rastreabilidade para frente é especialmente importante quando o produto de software entra em operação e na fase de manutenção. Como documentos de código e de concepção são modificados, é essencial determinar o conjunto completo de requisitos que podem ser afetados por essas modificações.

2.4 Documentação de Requisitos em Linguagem Natural

Empresas de tecnologia estão desenvolvendo um número muito alto de produtos de software cada vez mais complexos, e eventualmente, enfrentam o desafio de lidar com informações em enormes fluxos, que podem sobrecarregar a sua gestão e a capacidade de análise destas informações.

Os requisitos, que serão gerados de toda essa informação, são particularmente difíceis de tratar de maneira eficaz, devido à sua natureza não estruturada. Os requisitos têm também um potencial para crescimento, chegando a taxas que as informações

específicas e desafios de gestão do conhecimento começam a emergir: deterioração do repositório de requisitos e uma crescente dificuldade de identificar e manter os requisitos e suas inter-relações.

Uma das principais razões para esses problemas são os requisitos descritos em linguagem natural. Isso induz vários problemas como imprecisão, ambiguidade, imperfeição, conflito e inconsistência, que levam tempo para encontrar uma solução [29].

Processos de gerenciamento de requisitos podem ser muito diferentes no seu desenvolvimento. No entanto, as empresas que reconhecem tanto o envolvimento do cliente e seu próprio potencial inovador como meio para descoberta de serviços de produtos de sucesso e funcionalidades, são confrontados com um desafio comum: análise e avaliação de todos os requisitos, juntamente com o desejo do cliente e sugestão técnica rápida e possível.

Na gestão de requisitos tradicional há um foco implícito sobre especificações isoladas [23]. É um desafio, gerenciar uma enorme quantidade de requisitos que devem ser continuamente analisados e consolidados, deixando geralmente alguns destes intocados. Isso se reflete também pelas atuais ferramentas de gerenciamento de requisitos, que não fornecem a funcionalidade para criar ligações entre os requisitos, não fornecendo também nenhuma assistência na correspondência real entre os muitos requisitos já analisados.

Ferramentas de gerenciamento de requisitos poderiam fazer melhor do que oferecer mecanismos de busca simples por palavras-chaves, aliviando a carga manual de consolidação quando existem grandes quantidades de requisitos.

As empresas de software que enfrentam esses desafios podem chegar a uma encruzilhada, onde a escolha é reduzir o fluxo de requisitos ou atribuir mais recursos para lidar com eles. No entanto, visto de uma perspectiva de negócios, nenhuma dessas abordagens é particularmente gratificante, e em muitas situações, impossível.

Sufocando o levantamento de requisitos, a criação de novas exigências vai aumentar o risco de perder potenciais oportunidades de negócio, e adicionando mais pessoas para fazer o trabalho pode ser muito mais caro e às vezes improdutivo.

Assim, é necessário que os processos, ou ferramentas, que lidam com requisitos, estes sendo descritos em linguagem natural, devam apresentar o foco na qualidade da análise dos requisitos, para que não se perca a ligação entre as interdependências criadas à partir da especificação e a necessidade apresentada pelo cliente.

2.4.1 O papel dos Requisitos em Linguagem Natural

A maioria dos projetos de software desenvolvidos na indústria utiliza seus requisitos escritos e comunicados em Linguagem Natural (LN). Ainda assim, é complicado compreender e melhorar a forma como os requisitos podem ser especificados e formulados, o estado da prática é, geralmente, que as diretrizes de qualidade dos requisitos raramente são aplicadas [17][21].

Há uma grande diferença entre os modelos formais de escrita de requisitos e a informalidade que predomina na indústria. Várias razões podem ser identificadas para dizer por que os requisitos são inicialmente especificados em LN e, em muitos casos mantidos sob essa forma em todo o processo de desenvolvimento. Segundo [17], Algumas delas são:

- LN é a língua de comunicação primária, que é compartilhado por todos os interessados e participantes no processo de desenvolvimento. Linguagens formais exigem formação específica, o que é difícil de conseguir de todas as partes interessadas e, em particular de clientes ou usuários finais;
- A Engenharia de Requisitos é um processo social e evolucionário onde os requisitos são extraídos e especificados em diferentes níveis de abstração em diferentes pontos do processo de desenvolvimento. A LN é universal, o que significa que ela pode ser usada para falar sobre domínios arbitrários e em níveis de abstração. Muitas linguagens formais não têm essa força;
- Em grande escala de desenvolvimento, há relativamente poucos requisitos propostos sendo realmente selecionadas para implementação. Uma vez que nem todos os requisitos devem ser implementados, há pouca motivação para passar o tempo formalizando-se a escrita. Em particular, a experiência aponta que as empresas que valorizam a interação com seus clientes e reação rápida às condições de mercado não encontra custo-benefício para traduzir todos os requisitos em especificações formais;
- Muitos métodos formais não oferecem qualquer apoio para a gestão e análise de requisitos errados, incompletos ou parcialmente especificados. Em contraste, técnicas de análise de LN adaptam-se naturalmente a tais

situações, o que, na prática, faz-se em grande parte do ciclo de vida de um requisito;

- Enquanto linguagens formais podem melhorar a capacidade de verificar a consistência interna e integridade dos requisitos (um processo muitas vezes referido como verificação), eles não podem capturar as propriedades externas dos requisitos, por exemplo, a correspondência entre os requisitos e as intenções reais do usuário. Isso requer uma boa comunicação e interação com clientes para verificar as propriedades (validação), sendo assim, a LN é uma linguagem mais adequada.

Assim, apesar de suas deficiências reconhecidas, há poucos incentivos para evitar a linguagem natural. Deve-se, esperar que o seu uso não seja ignorado, assim as necessidades dos usuários e uma nova versão do sistema serão suportadas através de métodos e técnicas que reconhecem, de alguma forma, a comunicação em linguagem natural.

Um número crescente de empresas de desenvolvimento de software estão optando por se afastar de projetos isolados, como os de contrato de desenvolvimento, também chamado de desenvolvimento de software sob medida, para que o desenvolvimento de software atinja um mercado mais amplo. Este é, por exemplo, também chamado de interesse em comercial de prateleira, que enfrentam desafios distintos. Assim, para ficar à frente dos concorrentes é fundamental reduzir o tempo de lançamento no mercado.

Depois de uma primeira versão de um produto ter sido liberada, existe uma necessidade de um processo dinâmico de elicitação e prioridade. Neste ambiente dinâmico, onde os requisitos chegam de muitas fontes e partes interessadas diferentes (clientes, representantes de vendas, desenvolvedores, pessoal de apoio), a decisão de quais requisitos devem ser incluídos na próxima versão do produto deve, na maioria dos casos, ser feita com base na LN dos requisitos disponíveis no repositório, além da experiência e habilidade do Engenheiro de Requisitos e alguns pedidos inegociáveis feitos pelos clientes.

Em essência, as empresas enfrentam um problema de sobrecarga de informação. Mas, as soluções mais aparentes, reduzir a especificação de requisitos ou aumentar a quantidade de desenvolvedores, não são satisfatórios. Assim, a abordagem de utilização de LN para apoiar as atividades de análise e especificação de requisitos continua sendo usada e aperfeiçoada.

3 O MODELO DE PROCESSO PROPOSTO

Neste capítulo é apresentado o modelo de processo proposto para prover auxílio ao ER na análise, especificação e validação de requisitos, buscando aumentar a padronização da escrita e minimizar a ocorrência de inconsistências dentro dos requisitos.

Primeiramente faz-se menção a alguns conceitos que serão utilizados na descrição, deixando claro o papel que cada um deles representa dentro do modelo de processo proposto.

- **Engenheiro de Requisitos (ER)**: Responsável pela atividade de análise, especificação e validação dos requisitos;
- **Cliente**: Responsável por prover informações sobre suas necessidades, para que os requisitos possam estar de acordo e refleti-las dentro do software;
- **Requisitos**: São as necessidades que o cliente tem e que o software deve implantar. Serão descritos utilizando linguagem natural, assim poderemos tratar sua padronização através das palavras utilizadas na especificação;
- **Contexto Geral**: São palavras que irão identificar de uma maneira geral em que área/contexto o requisito a ser especificado estará inserido;
- **Contexto Específico**: Atua da mesma forma que o Contexto Geral, melhorando ainda mais a separação dos requisitos dentro da área/contexto, inserindo uma subárea;
- **Nível de Rigorosidade**: O nível de rigorosidade utilizado fica a critério do ER, levando em conta as palavras com inconsistências e a obtenção de uma maior qualidade na padronização da descrição;
- **Lista de Requisitos**: Uma lista comum que contém todos os requisitos identificados;
- **Documento de Requisitos**: O processo não faz menção a um modelo de Documento de Requisitos de Software (DRS) específico, apenas apresenta-se a possibilidade de criação deste documento.

O modelo de processo proposto está dividido em 3 fases: **Análise**, **Especificação** e **Validação**. Apresenta-se também para cada fase uma descrição mostrando o que é abordado naquela fase, qual o objetivo e seus artefatos de entrada e saída.

3.1 1ª FASE: ANÁLISE

Artefatos de Entrada: Descrição das necessidades do cliente.

Descrição: 1º Passo: O processo é iniciado quando cliente e ER interagem em reuniões iterativas e incrementais, **debatendo quais serão os requisitos do sistema**. Estas reuniões podem ser realizadas onde o cliente achar necessário, na maioria das vezes elas acontecem na empresa do cliente ou na empresa responsável pelo desenvolvimento do software. A cada nova reunião os assuntos discutidos, bem como as necessidades já identificadas, são retomados para que sejam incrementados até que um consenso final seja elaborado.

2º Passo: Em seguida, o ER transcreve as necessidades passadas pelo cliente e identificadas como possíveis requisitos do sistema.

Busca-se que todo o processo seja desenvolvido nas reuniões com o cliente, ou seja, todas as fases do processo proposto terão seu início e fim durante as reuniões. Quando isso não for possível, o ER deve terminar as fases o mais rápido possível para que a validação dos requisitos possa ser feita por ele e o cliente em um momento próximo.

Objetivo: A experiência e os trabalhos de [3][4][11] mostram que neste momento existe grande possibilidade das informações passadas pelo cliente serem incorretas, ou não representarem realmente a necessidade que ele tem. Assim, nessa fase é de extrema importância que o ER consiga o máximo de informações sobre as necessidades que o cliente tem para com o sistema, para que possa interpretá-las, transformando-as em requisitos do sistema.

Artefatos de Saída: Descrição das necessidades transcritas pelo ER.

3.2 2ª FASE: ESPECIFICAÇÃO

Artefatos de Entrada: Descrição das necessidades transcritas pelo ER.

Descrição: **1º Passo**: Tendo conhecimento sobre as necessidades do cliente, o ER poderá identificar, ou criar, em primeiro lugar o que chamamos de **Contexto Geral**, sendo palavras que identificarão onde, a que área, o requisito a ser especificado estará contido

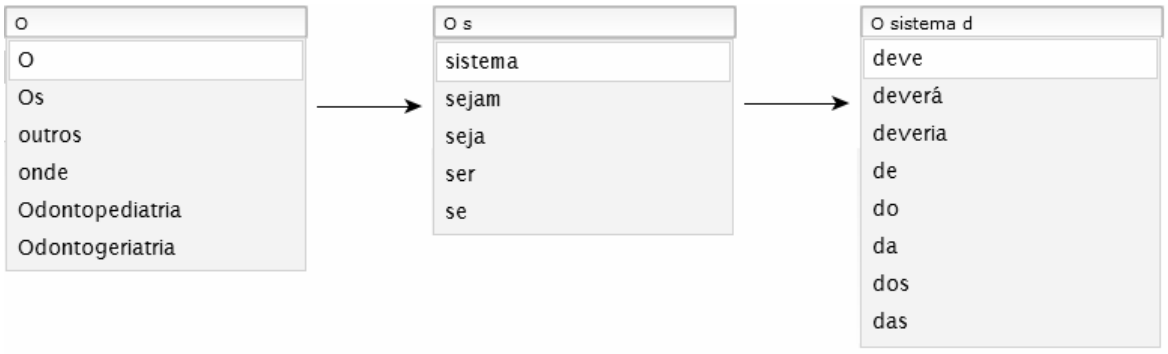
dentro do projeto. Esta primeira divisão serve, por exemplo, para um projeto desenvolvido em diversos módulos, assim os requisitos de cada módulo estarão separados através do Contexto Geral que os identifica, facilitando uma futura busca.

A segunda divisão que chamamos de **Contexto Específico**, tem a mesma função do Contexto Geral, mas tornando mais específica a área em que o requisito estará incluído. Utilizando a mesma ideia do exemplo acima, um projeto com diversos módulos, o Contexto Específico indica dentro de cada módulo onde o requisito a ser especificado ficará contido.

2º Passo: Após a criação dos contextos, o ER descreve na forma de requisito, utilizando palavras em Linguagem Natural, a necessidade já identificada através do cliente.

Na Figura 3.1 é apresentado um exemplo do início da descrição de um requisito, mostrando como o passo descrito acima ocorre, apresentam-se também alguns exemplos de palavras sugeridas.

Figura 3.1 - Exemplo do início de uma descrição



3º Passo: Depois da descrição, nosso modelo de processo traz uma classificação pela funcionalidade do requisito, tratando-o de acordo com [2], como **Requisito Funcional** e **Requisito Não-Funcional**.

Um dos diferenciais do modelo de processo proposto acontece durante o 1º e 2º passos desta fase, quando as palavras são descritas, tanto no Contexto Geral e Específico quanto na descrição do requisito, ocorre um auxílio ao ER, que obterá sugestões de palavras, já utilizadas anteriormente numa outra especificação, e poderá usar ou não para continuar sua descrição.

Importante salientar que, mesmo a reutilização de palavras proposta pelo modelo sendo opcional ao ER, caso as palavras não sejam reutilizadas isso refletirá na validação do requisito, pois a qualidade esperada não será alcançada, resultando numa validação negativa.

Para o modelo de processo proposto, um requisito estará completo quando estiver ligado a um Contexto Geral e Específico, sua descrição estiver completa e classificado como Requisito Funcional ou Não-Funcional.

Objetivo: Com esta fase, o processo proposto tem 4 objetivos.

1. **Facilitar a separação** dos requisitos, utilizando o Contexto Geral e Específico, que foram baseados nos trabalhos de [11][23], criando áreas e subáreas para que posteriormente a busca pelos requisitos seja efetiva e mais simples.
2. **Esclarecer as ambiguidades** que poderão ocorrer num projeto maior quando a quantidade de requisitos é muito grande e alguns requisitos se tornam muito parecidos com outros. Serão os contextos que esclarecerão as dúvidas que surgirem sobre a que parte do projeto pertence cada requisito em conflito.
3. **Padronização da descrição** dos requisitos, já que o ER terá disponíveis sugestões de palavras já utilizadas em outra especificação e que poderão ser utilizadas ou não, de acordo com a aceitação do ER e posterior validação do cliente.
4. **Reutilização de palavras**, com a possibilidade de estender-se a reutilização total ou parcial de um requisito já especificado. Diferentemente do reuso proposto em [30], nosso processo propõe o reuso de palavras já utilizadas e aponta a possibilidade do reuso parcial ou total de requisitos.

Artefatos de Saída: Especificação do Requisito de Software.

3.3 3ª FASE: VALIDAÇÃO

Artefatos de Entrada: Especificação do Requisito de Software.

Descrição: **1º Passo**: De acordo com níveis de rigorosidade pré-definidos pelo ER e cliente, a descrição do Contexto Geral e Específico e a descrição do requisito são avaliados e validados.

A validação ocorre em conjunto com o cliente, e está baseada no que chamamos de **Banco de Dados Histórico de Requisitos (BDHR)**, que contém todas as palavras utilizadas na especificação de requisitos em qualquer projeto dentro da organização. Estas palavras já haviam sido validadas, consequentemente, o ER junto com o cliente, deverão avaliar a utilização destas palavras na especificação do novo requisito criado, verificando se elas foram empregadas de maneira correta e representam a necessidade esperada.

Quando tratamos da validação de requisitos baseada em BDHR é necessário um cuidado para que o novo requisito especificado, utilizando palavras validadas de outros requisitos, obtenha a qualidade desejada [10]. Por isso, o modelo de processo propõe a validação, em conjunto com o cliente, do requisito completo (Contexto Geral, Específico, Descrição e Classificação), verificando a consistência de todos estes itens.

O parâmetro de avaliação proposto diz respeito ao conceito de nível de rigorosidade que o ER e cliente esperam do requisito. Ele leva em consideração a quantidade de palavras com inconsistências identificadas, ou seja, caso o ER utilize muitas palavras não sugeridas nos 1º e 2º passos da 2ª fase, significa que essas palavras ainda não foram utilizadas em nenhuma outra especificação de requisito. Se isso acontece, a chance da palavra utilizada estar escrita incorretamente, conter inconsistência, ou qualquer outro problema é muito maior.

2º Passo: Caso cliente e ER entendam a validação como **negativa**, propõe-se que sejam alterados os itens (contexto geral, específico e descrição do requisito) que contém inconsistências, e após essas alterações possa ocorrer uma nova validação.

Uma validação negativa pode representar que a necessidade do cliente não estava corretamente traduzida na forma de requisito, isto por ele passar informações incorretas ou pelo ER não conseguir identificar corretamente a necessidade passada pelo cliente; haviam erros e/ou inconsistências na especificação feita pelo ER ou os contextos não estavam claros o suficiente para separar o requisito dentro do projeto.

3º Passo: Caso cliente e ER entendam a validação como **positiva** propõe-se que todos os itens sejam armazenados, primeiramente em uma lista que chamamos de Lista de

Requisitos, que seria uma lista comum que contém todos os requisitos identificados, disponíveis e com fácil visualização, e em seguida armazenados no BDHR.

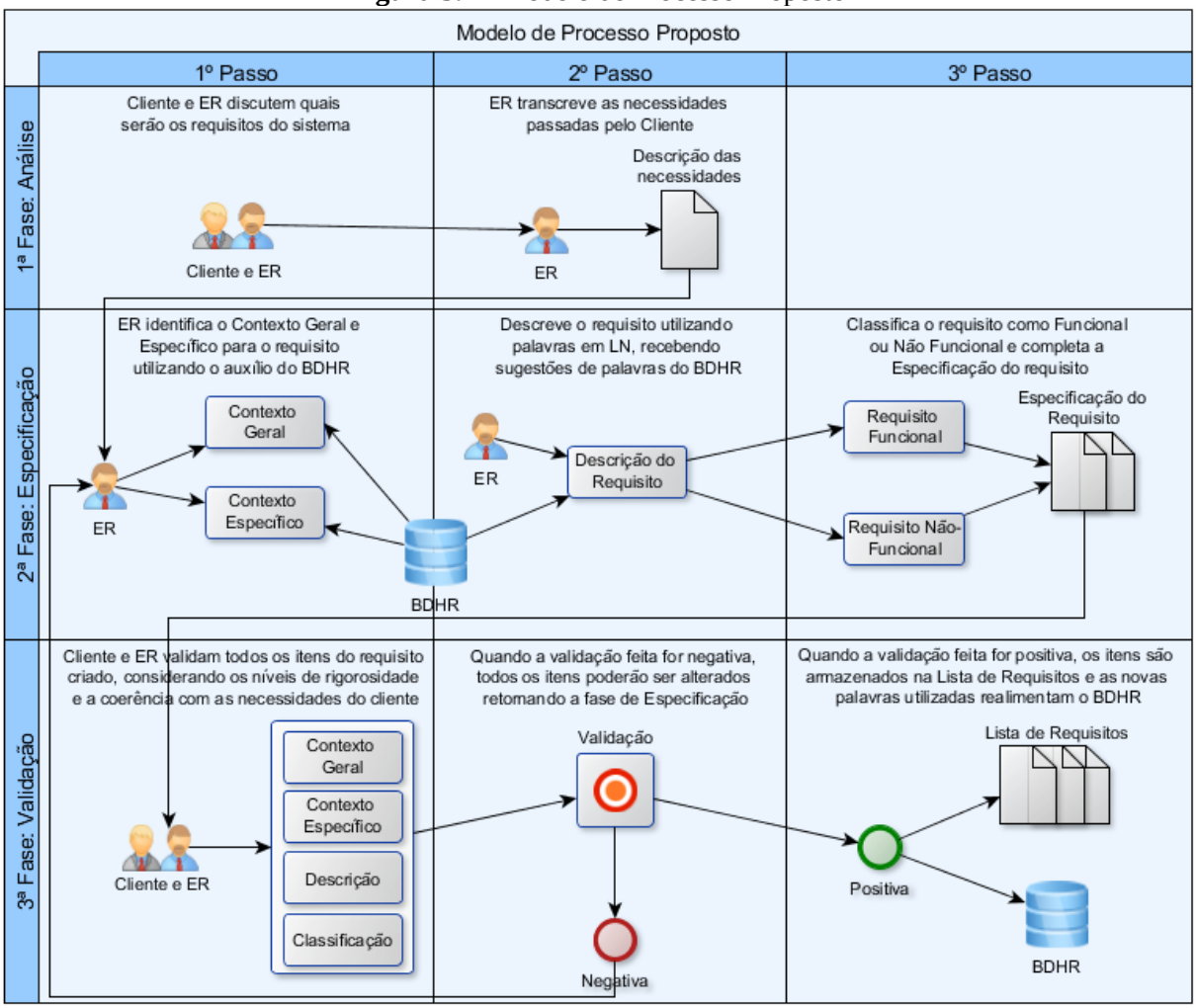
A validação positiva representa que o cliente pôde ler, compreender e validar o requisito descrito pelo ER. Desta forma, tem-se a confirmação de que a especificação do requisito está em conformidade com a necessidade que o cliente tem para com o software em desenvolvimento.

Objetivo: O objetivo da 3ª Fase do modelo de processo proposto é validar se o requisito descrito reflete as necessidades passadas pelo cliente, contendo a padronização de descrição proposta e alcançando a qualidade esperada. Neste momento o cliente pode perceber a tradução da sua necessidade expressa anteriormente em um requisito para o software. Assim a sua análise, juntamente com o ER, pode confirmar se o requisito criado é realmente o requisito desejado.

Artefatos de Saída: Lista de Requisitos contendo todos os requisitos especificados.

Exemplifica-se na Figura 3.2 o modelo de processo proposto, com suas fases e artefatos de entrada e saída.

Figura 3.2 - Modelo de Processo Proposto



Após o término de todas as reuniões, e todos os requisitos definidos e validados será possível a criação de um modelo de Documento de Requisitos de Software utilizando a Lista de Requisitos que o processo propõe. O processo não especifica um modelo ou template padrão, por não ser a intenção do trabalho, apenas busca-se oferecer todos os requisitos validados e armazenados, atribuídos a um projeto de software em desenvolvimento.

4 TRABALHOS RELACIONADOS

Na literatura existem diversos textos científicos e livros que tratam da garantia de qualidade nos requisitos, apontando benefícios e trazendo experiências que nos ajudam a entender melhor como criar e manter uma especificação de requisitos com qualidade. Esta seção busca apresentar alguns destes trabalhos, apontando a relação e o cenário em que estes e também o trabalho aqui apresentado está inserido.

Em seu trabalho, [31] descreve uma técnica que utiliza um pré-processamento da linguagem natural usada na criação dos requisitos de um software. Esse pré-processamento faz uso de domínios gerais e específicos para separar os requisitos, após isso, a técnica faz uma busca por palavras chamadas de “objetivos” pelo autor, que são descritos como a parte central do requisito. Neste trabalho o autor deixa claro que o objetivo da técnica é auxiliar o ER provendo informações extraídas da escrita dos requisitos.

Por sua vez, [28] tratou em seu trabalho de um paradigma interessante dentro da área de Engenharia de Requisitos, quando ele ponderou os requisitos de software com as mesmas regras que um medicamento precisa obter para que seja liberado pelos órgãos reguladores. Essa comparação tem validade, pois para que um medicamento seja liberado para consumo, ele precisa antes passar por testes que comprovem sua efetividade, seus riscos, suas limitações e todas as outras implicações que ele pode trazer na vida de um paciente. De maneira análoga os requisitos de software também deveriam passar por esses mesmos testes, já que eles podem afetar da mesma maneira a “vida” de um software. O autor obteve resultados significativos da adaptação destas regras para a Engenharia de Requisitos.

Segundo [32], a aplicação de técnicas de leituras sistemáticas como o *Perspective-Based Reading (PBR)* e não sistemáticas como Checklist durante a análise de requisitos trouxe bons resultados. Nestas técnicas vários inspetores inspecionam o documento de contexto de um software buscando erros ou inconsistências antes de transcrever o documento de requisitos. Esses erros são posteriormente avaliados para a comparação entre as duas técnicas. O trabalho também tratou da avaliação das perspectivas de inspetores, mostrando que diferentes papéis como ER, Arquitetos e Testadores podem apresentar diferentes resultados quando da utilização de uma técnica de leitura.

O modelo de processo proposto por [30] tem como objetivo alcançar a variabilidade da modelagem de requisitos, facilitando a definição de modelos complexos de requisitos, permitindo assim a sua reutilização. O trabalho ainda apresenta uma ferramenta

que ajuda na criação dos meta-modelos de requisitos apresentados pelo autor. A ferramenta definida facilita a criação e reutilização dos modelos de requisitos de maneira global.

A abordagem definida por [21] para o processo de Engenharia de Requisitos foi aplicar o reuso de requisitos utilizando padrões e catálogos de requisitos para a criação e especificação, e a rastreabilidade de requisitos para manter e gerenciar os requisitos criados. Foram avaliadas diversas abordagens de processos, sendo possível identificar os benefícios oferecidos por cada uma, criando uma abordagem que após a identificação do requisito, aplica-se um padrão de escrita de requisito, também definido dentro da abordagem e armazenado num catálogo de requisitos, gerando o requisito formalmente escrito e possibilitando assim, a sua reutilização.

Da mesma forma que [21], anteriormente [19] já havia apresentado seu modelo de processo para a elicitação de requisitos, que também utiliza padrões de escrita de requisitos, armazenados num catálogo de requisitos. A importante diferença entre os trabalhos está no modelo de processo apresentado, [19] trata, através de formas de exploração, as necessidades passadas pelos clientes em forma de requisito antes da aplicação dos padrões de escrita de requisitos. Outro ponto importante a se destacar na abordagem de [19] é a verificação de conformidade do modelo com a norma IEEE-830 [6].

Outro processo de Engenharia de Requisitos foi proposto em [33]. Os autores abrangem todo o processo de Engenharia de Requisitos, propondo dividi-lo em quatro fases: Elicitação e Desenvolvimento de Requisitos, Documentação de Requisitos, Verificação e Validação de Requisitos e por fim Gerenciamento e Planejamento de Requisitos. Os requisitos ficam armazenados num documento de SRS e os autores apontam que o diferencial do trabalho é, além de abranger todas as áreas, possibilitar o gerenciamento de mudanças em requisitos já acordados dentro de um desenvolvimento de software.

Ainda poderíamos citar trabalhos como [22][26] que tratam da classificação dos requisitos dentro de cenários de desenvolvimento e sua posterior reutilização, bem como os trabalhos de [20][24][25] que tratam das fases da Engenharia de Requisitos, juntamente com a reutilização, através da visão da indústria de software e os benefícios que essas abordagem podem trazer, como o aumento da qualidade da especificação de requisitos.

4.1 Avaliação Comparativa

Apresenta-se nesta seção, uma avaliação comparativa entre o modelo de processo proposto e alguns dos trabalhos apresentados anteriormente.

Dentre os trabalhos que tratam da garantia de qualidade nos requisitos, três foram selecionados para que pudéssemos compará-los ao modelo de processo aqui proposto. Outros poderiam ser escolhidos, que tratam diretamente das fases do Processo de Engenharia de Requisitos, mas o objetivo é apresentar uma comparação entre processos que utilizem divisões em fases, contextos e perspectivas para melhorar o tratamento que os requisitos recebem durante sua especificação.

Na comparação feita, foram escolhidos 10 itens que, de maneira geral, demonstram como os processos citados asseguram a qualidade dos requisitos. Os quatro primeiros itens demonstram como os processos comparados tratam as fases da Engenharia de Requisitos, e para o efeito de verificação do processo proposto, os seis últimos itens demonstram características para garantir a qualidade dos requisitos.

A seguir mostram-se os objetivos de avaliação de cada um destes itens.

1. Identificação dos Requisitos: mostra como os trabalhos selecionados tratam a identificação dos requisitos, quem e como são identificados.

2. Análise de Requisitos: neste item apresenta-se o tratamento da fase de Análise de Requisitos, mostrando como Engenheiros e Analistas trabalham com os requisitos antes de descrever sua especificação.

3. Especificação de Requisitos: este item representa a fase de Especificação de Requisitos de cada processo, mostrando de que maneira os requisitos são especificados, como são tratados e onde são armazenados.

4. Validação de Requisitos: aqui apresenta-se como os trabalhos selecionados tratam a validação de requisitos, qual o foco desta validação e de que maneira ela é feita.

5. Utiliza grupamento de Requisitos: neste item demonstra-se se os trabalhos utilizam grupamento de requisitos para melhorar sua separação e organização.

6. Utiliza padronização de Requisitos: a padronização de requisitos, sendo um dos objetivos deste trabalho, é exemplificada nos trabalhos selecionados.

7. Qualidade dos Requisitos: este item apresenta como e de que maneira os trabalhos tratam a garantia de qualidade dos requisitos

8. Nível de Participação do Cliente: a participação do cliente também foi escolhida como um item de comparação entre os trabalhos, pois um maior nível de participação do cliente dentro do processo de Engenharia de Requisitos corresponde a um maior nível de qualidade na especificação.

9. Nível de Interação manual do modelo: este item apresenta se os modelos de processos selecionados permitem ou possibilitam ajuda computacional ou automação para melhoria e aumento de qualidade.

10. Reutilização Parcial ou Total dos Requisitos: a reutilização de requisitos também foi escolhida como um item de comparação, mostrando como os trabalhos selecionados tratam este item.

No Quadro 4.1 mostra-se o comparativo entre processos.

Quadro 4.1 - Comparativo entre Trabalhos

	Chen et al.	Cabral et al.	Pandey et al.	Processo Proposto
Identificação dos Requisitos	Cliente cria os requisitos.	Não especificado no trabalho.	São identificados requisitos imaturos do ponto de vista do cliente.	Cliente e ER durante reuniões identificam os requisitos.
Análise de Requisitos	Não é feita uma análise dos requisitos.	Não é especificado no trabalho quem faz a análise dos requisitos.	Um Analista analisa os requisitos imaturos e cria os requisitos de sistema.	O ER faz a análise dos requisitos baseado na sua experiência e nos contextos.
Especificação de Requisitos	A especificação vem da identificação dos requisitos feita pelo cliente, escrita em LN.	Os requisitos já estão especificados, contidos no Documento de Requisitos.	A especificação é feita através do SRS.	A especificação é feita pelo ER, utilizando LN, com a opção de escolha de palavras validadas.
Validação de Requisitos	Não é objetivo do trabalho a validação os requisitos.	Foca na validação dos requisitos utilizando para isso técnicas de leitura e revisão de requisitos.	A validação é feita analisando o SRS juntamente com o cliente.	Os requisitos são validados pelo Cliente e ER através de níveis de rigorosidade.
Utiliza grupamento de Requisitos	Utiliza Domínios Gerais e Específicos.	Não utiliza grupamentos	Grupamentos em níveis do sistema.	Utilizam-se dois níveis: Contextos e Tipos.
Utiliza padronização de Requisitos	Através de "palavras-objetivos" dentro dos requisitos.	Através da identificação de erros nos requisitos.	Através da fase de Planejamento do requisito.	Através de palavras validadas.
Qualidade dos Requisitos	Busca o aumento da qualidade dos próprios requisitos.	Busca o aumento da qualidade da validação dos requisitos.	Busca o aumento da qualidade do processo todo.	Busca o aumento da qualidade na validação e na descrição dos requisitos.
Nível de Participação do Cliente	Alto, identificando e especificando os requisitos.	O trabalho não cita como foi a participação do cliente.	A participação do cliente deve ser constante.	Alto, identificando e validando os requisitos.
Nível de Interação manual do modelo	Médio. O modelo utiliza ajuda computacional para o tratamento dos requisitos.	Alto. O modelo utiliza técnicas que necessitam de muita interpretação humana.	Alto. O modelo não cita possibilidades de automação do processo.	Médio. O modelo apresenta possibilidade de automação para o tratamento dos requisitos.
Reutilização Parcial ou Total de Requisitos	Parcial, dependendo da experiência do ER.	Parcial, dependendo da quantidade de erros encontrados.	Parcial ou Total, dependendo do nível de Gerenciamento dos requisitos.	Parcial ou Total, dependendo da quantidade de palavras reutilizadas.

Esta análise sobre trabalhos relacionados ajuda-nos a perceber que a padronização da escrita dos requisitos, bem como a sua reutilização total ou parcial são temas significativos na busca pelo aumento da qualidade dos requisitos.

Mesmo quando os trabalhos tratam da garantia de qualidade dos requisitos dentro de uma especificação de software estes itens apresentam uma área muito propícia para pesquisa e desenvolvimento.

Com este comparativo reforçam-se as ideias já apontadas por [34] e que serviram como base para a criação do modelo de processo apresentado. Elas dizem respeito **à utilização de domínios específicos** para tratar e agrupar requisitos, utilizar contextos explícitos para prevenir inconsistências e diminuir redundâncias em requisitos conflitantes e por fim, **apresentar a padronização da escrita através do reuso de palavras** que formarão os requisitos.

Mais uma vez, é importante salientar que na comparação feita buscaram-se trabalhos que tratam da garantia da qualidade dos requisitos através de processos e técnicas para alcançar uma melhor qualidade na especificação.

O modelo de processo proposto diferencia-se dos trabalhos apresentados principalmente:

- **Na especificação de requisitos**, quando se propõe a sugestão de palavras validadas na descrição de requisitos;
- **Na validação de requisitos**, com o cliente e o ER validando os requisitos criados através de níveis de rigorosidade para que as palavras usadas e também os requisitos criados possam ser reutilizados de maneira total ou parcial;
- **No grupamento de requisitos**, oferecendo a divisão em contextos gerais e específicos e também na funcionalidade dos requisitos;
- **Na padronização da escrita**, através do reuso de palavras, sempre buscando alcançar um aumento na qualidade dos requisitos.

As normas [6][7] também são responsáveis por nortear a comparação aqui exposta, quando afirmam as propriedades de **isenção de ambiguidade**, o **uso da linguagem natural para descrição dos requisitos** e a **possibilidade da verificação de conformidade dos requisitos** para que a especificação de requisitos de software obtenha um bom nível de qualidade.

5 ESTUDO DE CASO

Para a criação deste estudo de caso foram selecionados dois módulos de um projeto acadêmico da Fábrica de Projetos de TIC - GAIA[1] situada no Departamento de Computação na Universidade Estadual de Londrina. Os módulos foram desenvolvidos em parceria com a Clínica Odontológica Universitária (COU), órgão pertencente à Universidade, com o objetivo de implantar um Prontuário Eletrônico que pudesse auxiliar o desenvolvimento das atividades dentro do Departamento de Odontologia.

O objetivo deste estudo de caso é **avaliar o aumento na qualidade da descrição dos requisitos** de um projeto de software, utilizando o modelo de processo proposto, buscando **minimizar a ocorrência de requisitos despadronizados**.

Utilizaram-se os conceitos apresentados por [8][9] com relação a reportar dados quantitativos e qualitativos dentro de um estudo de caso avaliando os resultados obtidos durante este estudo.

A equipe de desenvolvimento foi composta por 4 alunos de graduação desempenhando o papel de equipe de implementação e 2 alunos de mestrado desempenhando o papel de ER, juntamente com professores responsáveis pelas clínicas (disciplinas) de Odontopediatria e Odontogeriatria atendidas pela COU, e que neste cenário, representaram o papel de Cliente do projeto.

Os dois módulos foram tratados separadamente, para que os resultados da especificação de requisitos pudessem ser comparados. No módulo de Odontopediatria foi utilizado o modelo de processo aqui proposto, com as fases desenvolvidas de acordo com as ideias apresentadas, obtendo os artefatos de entrada e saída para cada fase. Já o módulo de Odontogeriatria foi desenvolvido sem a utilização de nenhum modelo de processo.

5.1 Análise Quantitativa dos Dados

Na análise quantitativa foram escolhidos 5 itens, refletindo as métricas que descrevem os dados referentes aos dois módulos utilizados no estudo de caso e avaliando a efetividade do processo proposto. No Quadro 5.1 apresentam-se estas métricas.

[1] Mais informações podem ser encontradas em: www.gaia.uel.br

Quadro 5.1 - Análise Quantitativa dos Dados

		Módulo Odontopediatria	Módulo Odontogeriatria
	Total de Requisitos especificados	35	34
	Total de Palavras utilizadas	853	633
1	Aprovados pelo cliente (1ª Iteração)	28	20
	Aprovados pelo cliente (2ª Iteração)	35	24
	Aprovados pelo cliente (3ª Iteração)	-	34
2	Problemas de descrição	5	12
3	Ambiguidade de contexto	0	6
4	Número de palavras reutilizadas	777	530
5	Número de palavras não reutilizadas	76	103

Descrevem-se agora como esses resultados foram alcançados e apresentam-se algumas projeções gráficas que os demonstram.

5.1.1 Requisitos Aprovados Pelo Cliente

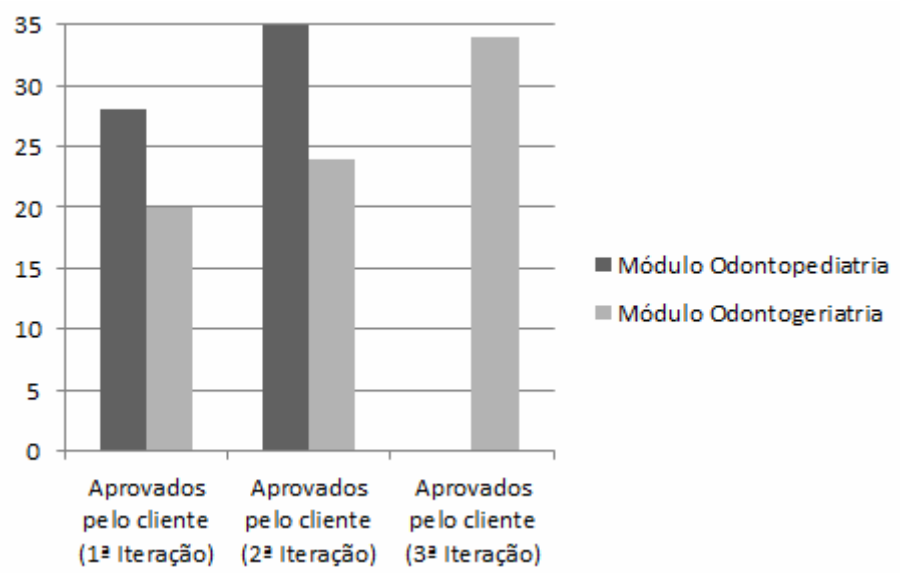
Essa métrica consistiu na análise de toda a descrição das necessidades do cliente, especificação destas necessidades em forma de requisito e, por fim, a validação do requisito criado pelo ER juntamente com o cliente. Neste estudo de caso foram necessárias 2 reuniões para que todos os requisitos fossem validados, sendo que a 1ª reunião consistiu em uma iteração e a 2ª reunião em duas iterações, totalizando três iterações conforme apontado no Quadro 5.1.

Analisando os dados apresentados podemos perceber que de um total de 35 requisitos especificados, utilizando o modelo de processo proposto no módulo Odontopediatria, 28 deles foram validados na 1ª iteração, isso corresponde a 80% de aprovação. No módulo Odontogeriatria com um total de 34 requisitos especificados, apenas 20 foram validados na 1ª iteração correspondendo a 58% de aprovação.

Na 2ª iteração o modelo de processo proposto alcançou o índice de 100% de requisitos aprovados, contra 70% de requisitos aprovados no módulo sem a utilização do processo proposto, sendo necessário uma 3ª iteração, somente para este módulo, para que fossem alcançados os 100% de requisitos aprovados.

Apresenta-se na Figura 5.1 um gráfico que demonstra os resultados da métrica utilizada.

Figura 5.1 - Análise de Requisitos aprovados



Por fim, a análise desta métrica nos permite identificar que utilizando o modelo de processo proposto houve um aumento de 22% na quantidade de requisitos aprovados na 1ª iteração e 30% na 2ª iteração.

5.1.2 Problemas de Descrição

Nesta seção busca-se apresentar a quantidade de requisitos em que ocorreram problemas na sua descrição, sejam erros de escrita ou requisitos que não representaram coerentemente a necessidade descrita pelo cliente.

Dos 35 requisitos especificados no módulo Odontopediatria foram identificados 5 requisitos com algum dos tipos de problemas citados, enquanto no módulo Odontogeriatria foram identificados 12 num total de 34 requisitos. Esses dados apontam uma diminuição na quantidade de requisitos com problemas de descrição de 21% quando utilizado o modelo de processo proposto.

5.1.3 Ambiguidade de Contexto

Esta métrica foi utilizada para avaliar a efetividade da 2ª Fase do modelo de processo proposto neste artigo. Com a identificação e atribuição de contextos Gerais e

Específicos para o requisito especificado, este se tornou distinguível de outros requisitos semelhantes, deixando claro em que lugar do projeto o requisito especificado está contido.

Não ocorreram ambiguidades de contexto com os requisitos especificados utilizando o modelo de processo proposto. Ocorreram 6 inconsistências relacionadas com a ambiguidade de contexto nos requisitos especificados no módulo Odontogeriatria. Esses dados nos permitem identificar uma melhora de 17% com relação a problemas de ambiguidade de contexto nos requisitos especificados.

Para esta métrica, na Análise Qualitativa de Dados, serão apresentados exemplos de requisitos identificados que comprovam a análise aqui apresentada.

5.1.4 Número de Palavras Reutilizadas

A quantidade de palavras reutilizadas na descrição dos requisitos também foi escolhida como métrica para avaliação da efetividade do modelo de processo proposto.

Todas as palavras que foram reutilizadas **pelo menos uma vez** na especificação dos requisitos deste Estudo de Caso foram identificadas nesta análise. Existiam ainda palavras inicialmente contidas no BDHR, assim caso essas palavras fossem utilizadas elas também seriam identificadas como reutilizadas.

Para ambos os módulos existiam 187 palavras disponíveis inicialmente no BDHR, que são apresentadas no Quadro 5.2.

Quadro 5.2 - Palavras inicialmente disponíveis no BDHR

Letras	Palavras
#	1, 2, 3, 4, 5, 6, 7, 8, 9, 0.
A	a, as, ao, aos, apresentar, alcançar, acaso, absorver, atender, atendente, através, atravessar.
B	baseado, basear, bloquear, bloqueado, base, blindado, blindagem.
C	com, casa, característica, como, cada, cliente, caso, correto, correta, corretamente, conseguir.
D	da, das, de, desde, dever, devem, do, dos, deve, deverá, deveria, dente, dentista, digam, diga.
E	e, é, em, esse, este, ele, ela, especificar, especificação, escrever, escrita, executar, especificados.
F	faz, fez, feita, fazer, foi, fim, fratura, fraturar.
G	gerar, garantir, gasto, gesto, ganhar, ganhador.
H	hoje, houve, haveria, histórico.

Letras	Palavras
I	irá, item, índice, indicado, indicação.
L	ler, leva, levará, lista, listar, lugar, lido, lendo.
M	mas, mesmo, módulo, modalidade, moda, mistura, mudança, mais, menos, memória, meses.
N	no, na, nas, nos, num, novo, nova, norma, nome.
O	o, os, ostentar, oscilar, outros, opção, opções, objeto, objetos.
P	para, parar, permanecer, permitido, página, paciente, próprio, palavras, pontos, possibilidade.
Q	que, qual, quantos, querer, quiser.
R	resto, respostas, riscos, resultado, resultando, responsável, responsabilidade, reservar, reserva, ruído.
S	se, seu, sua, suas, seja, sejam, sobra, substância, sobrar, sistema.
T	todo, toda, teu, todos, tem, tido, texto, teste, ter, teriam.
U	um, uma, uns, umas, unidade, utiliza, utilizar.
V	virá, ver, visa, visto, visitar, visitante, volta.

Como os requisitos do módulo Odontopediatria foram especificados utilizando o modelo de processo proposto essas palavras foram utilizadas na 2ª Fase do processo. No caso do módulo Odontogeriatria que não fez uso de nenhum modelo essas palavras foram utilizadas na análise dos dados do Estudo de Caso, para verificar se elas foram reutilizadas ou não.

No módulo Odontopediatria foram utilizadas 853 palavras no total para especificar os 35 requisitos, sendo reutilizadas 777 palavras, gerando uma porcentagem de reutilização de 91%. No módulo Odontogeriatria foram utilizadas 633 palavras para especificar 34 requisitos, destas, foram reutilizadas 530 palavras, obtendo uma porcentagem de reutilização de 84%. Através destes dados podemos confirmar um aumento de 7% na reutilização de palavras que compõem os requisitos utilizando o modelo de processo proposto.

5.1.5 Número de Palavras Não Reutilizadas

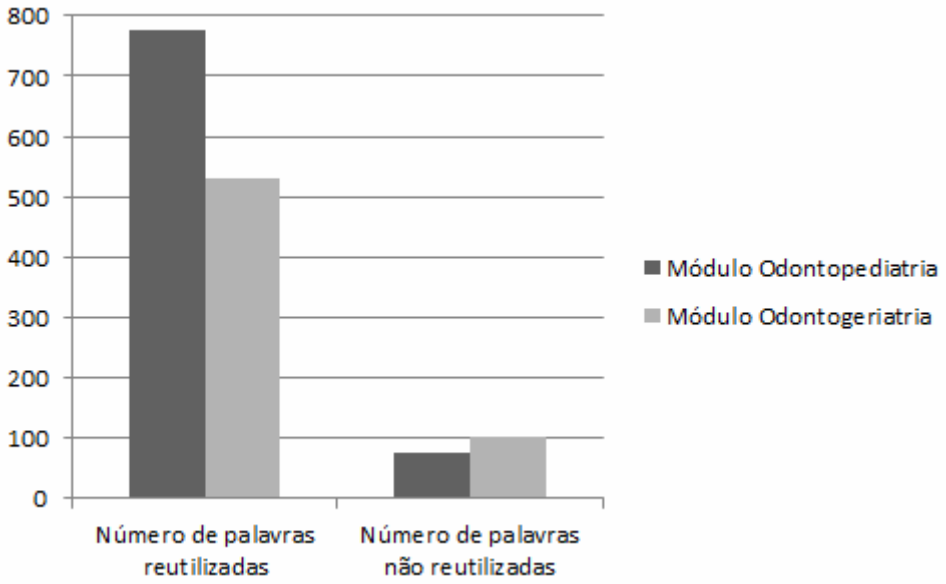
Utiliza-se também uma métrica para avaliar o número de palavras não reutilizadas identificadas. Assim pôde-se confirmar de que maneira a reutilização de palavras afeta a utilização das palavras dentro da especificação dos requisitos.

Para o cálculo desta métrica todas as palavras utilizadas na especificação dos requisitos foram verificadas e se a palavra foi utilizada **somente uma vez** ela foi identificada como não reutilizada.

No módulo Odontopediatria não foram reutilizadas 76 palavras, portanto 9% de palavras não reutilizadas no total de palavras. No módulo Odontogeriatria não foram reutilizadas 103 palavras novas, obtendo 16% de palavras não reutilizadas. Estes dados mostram que houve uma diminuição na quantidade de palavras não reutilizadas em 7%.

Os resultados destas duas últimas métricas são apresentados na Figura 5.2, demonstrando que a relação entre reutilização e a não reutilização de palavras para a descrição dos requisitos é correspondente a uma proporcionalidade inversa, com 7% de aumento na reutilização e 7% de diminuição na não reutilização.

Figura 5.2 - Análise do número de palavras reutilizadas e não reutilizadas



5.2 Análise Qualitativa dos Dados

Na análise qualitativa dos dados deste estudo de caso utilizam-se as mesmas métricas já apresentadas na análise quantitativa. No Quadro 5.3 apresentam-se todos os requisitos obtidos para o módulo Odontopediatria utilizando o modelo de processo proposto e no módulo Odontogeriatria sem a utilização de modelos de processo.

Quadro 5.3 - Requisitos obtidos no Estudo de Caso

Requisitos Obtidos	Módulo Odontopediatria	Módulo Odontogeriatria
01	O módulo de Odontopediatria deverá ser apresentado quando for selecionado dentro da página de Gerência de Prontuários.	A página deve aparecer quando clicar em Odontogeriatria nas fichas do paciente na página de Gerência de Prontuários.
02	O módulo deve incluir no início de sua página o Histórico de todos os atendimentos realizados pelo paciente.	O Histórico das consultas irá aparecer no início da página, com todos os atendimentos do paciente.
03	A cada novo atendimento realizado pelo paciente no COU o histórico deverá ser incrementado e não deverão ser apagadas informações armazenadas anteriormente.	Não deve ser apagado nada, toda nova consulta do paciente deve ser armazenada separadamente.
04	O módulo deverá incluir, como 1º item da página, a Avaliação de doenças ocorridas na Infância do paciente.	O sistema deve armazenar os itens para Orientação de Data do paciente.
05	O módulo deve apresentar todos os campos referentes ao modelo de Avaliação de Infância com as opções de “Sim” ou “Não”.	Deve existir um campo para que o paciente diga qual é o dia da semana e qual é o horário do atendimento feito.
06	Caso a resposta de algum item da Avaliação de Infância seja “Sim”, o módulo deverá mostrar um campo para a descrição de quantos meses o paciente tinha quando a doença ocorreu.	O cliente deseja que sejam feitos campos para a Orientação Local do paciente, para que ele diga onde está.
07	O módulo deverá incluir, como 2º item da página, a Avaliação de Hábitos que o paciente tem durante o dia a dia.	Caso o paciente esqueça algum item da Orientação Local o dentista deve perguntar a ele para que os itens estejam completos.
08	O módulo deve apresentar todos os campos referentes à Avaliação de Hábitos do paciente com as opções de “Sim” ou “Não”.	O sistema deve permitir que o dentista avalie a Memória Imediata do paciente, para que ele escute e repita o nome de três objetos.
09	Caso a resposta de algum item da Avaliação de Hábitos seja “Sim”, o módulo deverá mostrar um campo com duas opções “Parou” e “Ainda usa”.	Devem existir três campos de texto para que o dentista armazene os nomes dos objetos utilizados neste item.
10	Caso a resposta de algum item da Avaliação de Hábitos seja “Sim”, e a opção “Ainda usa” seja escolhida, o módulo deve apresentar um campo para a descrição de quantos meses o paciente tem esse hábito.	O sistema irá permitir que o dentista avalie a atenção do paciente através de um teste.
11	Caso a resposta de algum item da Avaliação de Hábitos seja “Sim”, e a opção “Parou” seja escolhido, o módulo não deverá mostrar nenhum campo.	O teste de Cálculo deve fazer com que o paciente recorde a subtração começando com 100 e diminuindo de 7 em 7 unidades.

Requisitos Obtidos	Módulo Odontopediatria	Módulo Odontogeriatria
12	O módulo deverá incluir, como 3º item da página, como o paciente faz o Consumo de Água no seu dia a dia.	O teste de Atenção deve fazer com que o paciente recorde como soletrar a palavra “Mundo” ao contrário.
13	O módulo deve apresentar todos os campos referentes ao Consumo de Água do paciente com as opções de “Sim” ou “Não”.	A página deverá apresentar campos para que as respostas dos testes de Cálculo e Atenção sejam armazenadas.
14	O módulo deverá incluir, como 4º item da página, o Histórico de Alimentação, que indicará com opções, quantas vezes no dia o paciente se alimenta de cada item.	Na página de Odontogeriatria o dentista vai anotar o resultado do teste de Evocação para Recordar o nome dos três objetos usados no teste de Memória imediata.
15	Caso a resposta de algum item do Histórico de Alimentação seja “Sim”, o módulo deverá mostrar um campo com quatro opções “3x por dia”, “4x por dia”, “5x por dia” e “6x por dia”.	Devem existir três campos para armazenagem das respostas que o paciente deu no teste de Evocação.
16	O módulo deverá incluir, como 5º item da página, a Avaliação de Higienização do dia a dia do paciente.	A página irá apresentar vários testes de Linguagem para a avaliação do conhecimento do paciente.
17	Caso a resposta de algum item da Avaliação de Higienização seja “Sim”, o módulo deverá mostrar um campo para que possa ser especificada a frequência com que o paciente faz a higienização.	No teste de Linguagem Denominação a página deve permitir que o dentista anote o resultado da resposta do paciente através de *checkboxes*.
18	O módulo deve mostrar na Avaliação de Higienização um campo para que seja especificada como é a colaboração do paciente quando a higienização é feita, com as opções de “Colaborador” e “Não Colaborador”.	No teste de Linguagem Repetição o dentista irá dizer uma frase e pedir para o paciente repetir. O sistema deve ter um campo para anotar se a resposta do paciente foi correta.
19	O módulo deverá incluir, como 6º item da página, os itens para avaliação do Risco de Cárie nos dentes do paciente.	No teste de Linguagem Comando Verbal o dentista irá passar três comandos para o paciente que deverá realiza-los.
20	Caso a resposta de algum item da avaliação de Risco de Cárie seja “Sim”, o módulo não deve mostrar outros campos para mais informações.	Na página deverão existir três campos separados para que o dentista marque se o paciente fez todos os comandos corretamente.
21	O módulo deve mostrar na avaliação de Risco de Cárie um campo para que sejam escolhidas as características dentárias do paciente, também com a opção de “Outros” para a especificação de outras características.	O teste de Linguagem Comando Escrito irá observar se o paciente responde corretamente a um comando escrito num papel.

Requisitos Obtidos	Módulo Odontopediatria	Módulo Odontogeriatria
22	O módulo deverá incluir, como 7º item da página, a avaliação de Experiência de Cárie Dentária do paciente.	O sistema deve ter um campo para armazenar se o paciente executou corretamente o teste de Comando Escrito.
23	Caso a resposta de algum item da avaliação de Experiência de Cárie Dentária seja “Sim”, o módulo não deve mostrar outros campos para mais informações.	No teste de Linguagem Escrita o paciente deve escrever corretamente uma frase em língua portuguesa.
24	O módulo deve apresentar na avaliação de Experiência de Cárie Dentária os itens de CPOD, CPOS e CEO. Cada um destes itens deverá apresentar campos de texto para descrição de suas características.	Deve ter um campo para armazenar se o paciente conseguiu escrever a frase corretamente no teste de Escrita.
25	O módulo deverá incluir, como 8º item da página, o Índice de Higiene Oral Simplificado (IHOS) do paciente.	No teste de Linguagem Desenho o paciente deve copiar um desenho passado pelo dentista.
26	O módulo deve apresentar no IHOS os itens de Dentição Mista e Dentição Permanente. Cada um destes itens deverá apresentar campos de texto para descrição de suas características.	Na página deverá existir um campo para que o dentista marque se o paciente desenhou certo.
27	O módulo deve apresentar no IHOS os itens de Higiene Oral, permitindo sua classificação.	O teste de Recordação Dentária deve permitir ao dentista avaliar se o paciente tem conhecimento o tratamento executado.
28	O módulo deve armazenar em uma tabela no final do IHOS todas as avaliações feitas pelo dentista correspondente a Higiene Oral do paciente.	Na página de Odontogeriatria irão existir três campos de texto para armazenar as respostas do paciente sobre o tratamento executado.
29	O módulo deverá manter listado todo o histórico do paciente, como 5º item, no Índice de Higiene Oral.	Todos os campos da página devem ter uma resposta padrão para facilitar o atendimento do dentista.
30	O módulo deverá incluir, como 9º item da página, a avaliação de Traumatismo do paciente.	Os pacientes da Odontogeriatria poderão ter uma reavaliação dos itens, assim o sistema deve permitir que ocorram mudanças.
31	O módulo deve apresentar no item de avaliação de Traumatismo os campos para especificação das características do traumatismo dentário sofrido pelo paciente.	O cliente deseja todos os testes feitos com o paciente na página sejam avaliados com pontos armazenados em campos próprios e atribuídos pelo dentista.

Requisitos Obtidos	Módulo Odontopediatria	Módulo Odontogeriatria
32	No item de Traumatismo o módulo deve apresentar os campos de Fratura Coronária e Fratura Redicular, para que o dentista faça a avaliação específica da fratura.	Deverá existir na página um local para armazenar o total de pontos obtidos pelo paciente.
33	O módulo deve incluir no final da página, dois campos para que sejam especificados os resultados do tratamento e da evolução do paciente.	Os itens da página devem ser idênticos ao Prontuário Dentário Único (PDU) que é utilizado no COU.
34	Para o módulo de Odontopediatria todos os itens devem obedecer ao Prontuário Dentário Único (PDU) que ainda é utilizado nos atendimentos do COU.	No final haverá os botões de “Salvar” e “Cancelar” para voltar à página inicial.
35	No final da página, o módulo deve apresentar os botões de “Salvar” para que as informações descritas sejam salvas no Banco de Dados e “Cancelar” que voltará para a página de Gerência de Prontuários.	

Esses requisitos foram especificados nas primeiras iterações de cada módulo, ou seja, alguns deles possuem inconsistências que serão discutidas a seguir.

5.2.1 Requisitos Aprovados Pelo Cliente e Problemas de Descrição

Para facilitar o relato dos resultados apresentam-se estas duas métricas juntas. Como já citado, essas métricas foram utilizadas para avaliar a aprovação da especificação dos requisitos e identificar problemas ou inconsistências nos mesmos.

Importante salientar que estes problemas na descrição dos requisitos foram corrigidos nas iterações seguintes de cada módulo, conforme apresentado no Quadro 5.1.

No Quadro 5.4 estão representados os requisitos obtidos neste Estudo de Caso, para o módulo Odontogeriatria, e que apresentaram problemas na sua descrição.

Quadro 5.4 - Problemas de Descrição Odontogeriatria

Número de Requisitos com Problemas de Descrição: 12		
Requisito	**Descrição**	**Real Necessidade do Cliente**
05	Deve existir um campo para que o paciente diga qual é o dia da semana e qual é o horário do atendimento feito.	No módulo Odontogeriatria devem existir dois campos para que o dentista anote a resposta do paciente para as perguntas: “Qual o dia da semana?” e “Qual o Horário do Atendimento?”.
06	O cliente deseja que sejam feitos campos para a Orientação Local do paciente, para que ele diga onde está.	O módulo Odontogeriatria deve apresentar um campo de texto para que o dentista anote a resposta do paciente para a pergunta: “Onde você está?”, no item Orientação Local.
08	O sistema deve permitir que o dentista avalie a Memória Imediata do paciente, para que ele escute e repita o nome de três objetos.	O módulo Odontogeriatria deve apresentar três campos de texto para que o dentista anote a resposta do paciente, no item Memória Imediata, onde ele ouve e repete o nome de três objetos.
09	Devem existir três campos de texto para que o dentista armazene os nomes dos objetos utilizados neste item.	Na 2º iteração do módulo Odontogeriatria este requisito foi incluído no requisito 08.
10	O sistema irá permitir que o dentista avalie a atenção do paciente através de um teste.	O módulo Odontogeriatria deve permitir ao dentista avaliar e anotar, através de um campo, o grau de atenção do paciente quando pedido para relembrar o nome de um objeto mostrado.
13	A página deverá apresentar campos para que as respostas dos testes de Cálculo e Atenção sejam armazenadas.	O módulo Odontogeriatria deve apresentar dez campos do tipo *checkbox* para que o dentista anote a resposta do paciente para os testes de Cálculo e Atenção.
19	No teste de Linguagem Comando Verbal o dentista irá passar três comandos para o paciente que deverá realiza-los.	O módulo Odontogeriatria deve apresentar três campos de texto, assim o dentista anote a resposta do paciente para o teste de Linguagem Comando, onde o paciente ouve e executa comandos.
20	Na página deverão existir três campos separados para que o dentista marque se o paciente fez todos os comandos corretamente.	Na 2º iteração do módulo Odontogeriatria este requisito foi incluído no requisito 19.
21	O teste de Linguagem Comando Escrito irá observar se o paciente responde corretamente a um comando escrito num papel.	O módulo Odontogeriatria deve permitir que dentista anote em um campo de texto a resposta do paciente, para um comando escrito num papel, no teste Linguagem Comando Escrito.

Requisito	Descrição	Real Necessidade do Cliente
22	O sistema deve ter um campo para armazenar se o paciente executou corretamente o teste de Comando Escrito.	Na 2º iteração do módulo Odontogeriatria este requisito foi incluído no requisito 21.
23	No teste de Linguagem Escrita o paciente deve escrever corretamente uma frase em língua portuguesa.	O módulo Odontogeriatria deve apresentar um campo do tipo *checkbox*, para que o dentista anote se o paciente escreveu corretamente uma frase em língua portuguesa no teste Linguagem Escrita.
24	Deve ter um campo para armazenar se o paciente conseguiu escrever a frase corretamente no teste de Escrita.	Na 2º iteração do módulo Odontogeriatria este requisito foi incluído no requisito 23.

Da mesma maneira, no Quadro 5.5 estão representados os requisitos obtidos para o módulo Odontopediatria, e que apresentaram problemas na sua descrição.

Quadro 5.5 - Problemas de Descrição Odontopediatria

Número de Requisitos com Problemas de Descrição: 05		
Requisito	**Descrição**	**Real Necessidade do Cliente**
01	O módulo de Odontopediatria deverá ser apresentado quando for selecionado dentro da página de Gerência de Prontuários.	O módulo Odontopediatria deve ser iniciado quando for selecionado no item Fichas, dentro da página de Gerência de Prontuários.
02	O módulo deve incluir no início de sua página o Histórico de todos os atendimentos realizados pelo paciente.	O módulo Odontopediatria deve incluir no início de sua página um campo do tipo *drop-down* armazenando todos os atendimentos realizados pelo paciente.
27	O módulo deve apresentar no IHOS os itens de Higiene Oral, permitindo sua classificação.	O módulo Odontopediatria deve permitir ao dentista classificar os itens da seção Higiene Oral dentro do item IHOS.
28	O módulo deve armazenar em uma tabela no final do IHOS todas as avaliações feitas pelo dentista correspondente a Higiene Oral do paciente.	O módulo Odontopediatria deve apresentar todas as avaliações feitas pelo paciente na seção Higiene Oral, e deve mostrá-las em uma tabela no final do item IHOS, permitindo a inclusão de novas avaliações.
33	O módulo deve incluir no final da página, dois campos para que sejam especificados os resultados do tratamento e da evolução do paciente.	O módulo Odontopediatria deve incluir dois campos de texto, depois de todos os itens da página, para que o dentista anote os resultados do tratamento e da evolução do paciente.

Outro fato importante a se destacar, é a redução do número de iterações nas reuniões com o cliente. Essa redução foi alcançada através da reutilização de palavras baseadas no BDHR, proposto pelo modelo de processo, e também da experiência do ER, possibilitando traduzir as necessidades do cliente de forma mais rápida, consequentemente aumentando a qualidade da especificação.

5.2.2 Ambiguidade de Contexto

Esta métrica avaliou a ambiguidade dos requisitos obtidos no Estudo de Caso com relação ao seu contexto, ou seja, cada requisito deve ser único e endereçar somente uma área/função/conceito específico do sistema. Quando isso não ocorreu, identificou-se uma ambiguidade de contexto.

No Quadro 5.6 apresentam-se os requisitos com ambiguidades de contexto do módulo Odontogeriatria. Também mostram-se quais foram os motivos das ambiguidade identificadas.

Quadro 5.6 - Ambiguidade de Contexto Odontogeriatria

Número de Requisitos com Ambiguidade de Contexto: 06		
Requisito	**Descrição**	**Ambiguidade Identificada**
02	O Histórico das consultas irá aparecer no início da página, com todos os atendimentos do paciente.	O requisito não especifica como o Histórico do paciente irá aparecer na página, nem como os atendimentos do paciente estarão contidos no histórico. Outros módulos também usam o histórico.
04	O sistema deve armazenar os itens para Orientação de Data do paciente.	Este requisito não especifica de que maneira os “itens” serão armazenados, muito menos quais são os itens descritos. Também não identifica a qual módulo, ou parte do sistema pertence.
15	Devem existir três campos para armazenagem das respostas que o paciente deu no teste de Evocação.	O requisito não apresenta a qual módulo ou parte do sistema pertence, que tipo de campos serão necessários para estas respostas ou onde estes campos estarão dentro da página.
16	A página irá apresentar vários testes de Linguagem para a avaliação do conhecimento do paciente.	O requisito não identifica quais são os Testes de Linguagem, como eles serão avaliados ou ainda de que maneira o dentista irá realizar esta avaliação.

Requisito	Descrição	Ambiguidade Identificada
26	Na página deverá existir um campo para que o dentista marque se o paciente desenhou certo.	Este requisito não deixa claro qual a sua função dentro do sistema, por trazer mais perguntas do que descrever o sistema: “O que o paciente desenhou?”, “Qual o motivo desse desenho?”, “Onde este campo deve ficar na página?”.
32	Deverá existir na página um local para armazenar o total de pontos obtidos pelo paciente.	Não está claro onde este requisito está inserido dentro do projeto, muito menos à que módulo ele pertence ou qual seria este “local” que o cliente espera que o requisito esteja.

No módulo Odontopediatria não houveram ambiguidades de contexto identificadas nos requisitos deste Estudo de Caso, pois os requisitos foram especificados utilizando o modelo de processo proposto, que previa em sua 2ª Fase a utilização de Contextos Gerais e Específicos para o tratamento dos requisitos e também a classificação por funcionalidade, que ajudaram a **minimizar a ocorrência de ambiguidade nos requisitos**.

Para demonstrar os resultados obtidos com a utilização da 2ª Fase do processo proposto apresentamos no Quadro 5.7 os requisitos do módulo Odontopediatria juntamente com suas respectivas classificações.

Quadro 5.7 - Utilização dos Contextos para requisitos

Nº	Contexto Geral	Contexto Específico	Descrição	Classificação
01	Gerência de Prontuários	Seleção de fichas de atendimento do paciente	O módulo de Odontopediatria deverá ser apresentado quando for selecionado dentro da página de Gerência de Prontuários.	Requisito Não Funcional
02	Gerência de Prontuários	Histórico de atendimento do paciente no módulo Odontopediatria	O módulo deve incluir no início de sua página o Histórico de todos os atendimentos realizados pelo paciente.	Requisito Não Funcional
03	Gerência de Prontuários	Histórico de atendimento do paciente no módulo Odontopediatria	A cada novo atendimento realizado pelo paciente no COU o histórico deverá ser incrementado e não deverão ser apagadas informações armazenadas anteriormente.	Requisito Funcional
04	Gerência de Prontuários	Item 1 no módulo Odontopediatria	O módulo deverá incluir, como 1º item da página, a Avaliação de doenças ocorridas na Infância do paciente.	Requisito Funcional

Nº	Contexto Geral	Contexto Específico	Descrição	Classificação
05	Gerência de Prontuários	Item 1 no módulo Odontopediatria	O módulo deve apresentar todos os campos referentes ao modelo de Avaliação de Infância com as opções de “Sim” ou “Não”.	Requisito Funcional
06	Gerência de Prontuários	Item 1 no módulo Odontopediatria	Caso a resposta de algum item da Avaliação de Infância seja “Sim”, o módulo deverá mostrar um campo para a descrição de quantos meses o paciente tinha quando a doença ocorreu.	Requisito Funcional
07	Gerência de Prontuários	Item 2 no módulo Odontopediatria	O módulo deverá incluir, como 2º item da página, a Avaliação de Hábitos que o paciente tem durante o dia a dia.	Requisito Funcional
08	Gerência de Prontuários	Item 2 no módulo Odontopediatria	O módulo deve apresentar todos os campos referentes à Avaliação de Hábitos do paciente com as opções de “Sim” ou “Não”.	Requisito Funcional
09	Gerência de Prontuários	Item 2 no módulo Odontopediatria	Caso a resposta de algum item da Avaliação de Hábitos seja “Sim”, o módulo deverá mostrar um campo com duas opções “Parou” e “Ainda usa”.	Requisito Funcional
10	Gerência de Prontuários	Item 2 no módulo Odontopediatria	Caso a resposta de algum item da Avaliação de Hábitos seja “Sim”, e a opção “Ainda usa” seja escolhida, o módulo deve apresentar um campo para a descrição de quantos meses o paciente tem esse hábito.	Requisito Funcional
11	Gerência de Prontuários	Item 2 no módulo Odontopediatria	Caso a resposta de algum item da Avaliação de Hábitos seja “Sim”, e a opção “Parou” seja escolhido, o módulo não deverá mostrar nenhum campo.	Requisito Funcional
12	Gerência de Prontuários	Item 3 no módulo Odontopediatria	O módulo deverá incluir, como 3º item da página, como o paciente faz o Consumo de Água no seu dia a dia.	Requisito Funcional
13	Gerência de Prontuários	Item 3 no módulo Odontopediatria	O módulo deve apresentar todos os campos referentes ao Consumo de Água do paciente com as opções de “Sim” ou “Não”.	Requisito Funcional
14	Gerência de Prontuários	Item 4 no módulo Odontopediatria	O módulo deverá incluir, como 4º item da página, o Histórico de Alimentação, que indicará com opções, quantas vezes no dia o paciente se alimenta de cada item.	Requisito Funcional

Nº	Contexto Geral	Contexto Específico	Descrição	Classificação
15	Gerência de Prontuários	Item 4 no módulo Odontopediatria	Caso a resposta de algum item do Histórico de Alimentação seja “Sim”, o módulo deverá mostrar um campo com quatro opções “3x por dia”, “4x por dia”, “5x por dia” e “6x por dia”.	Requisito Funcional
16	Gerência de Prontuários	Item 5 no módulo Odontopediatria	O módulo deverá incluir, como 5º item da página, a Avaliação de Higienização do dia a dia do paciente.	Requisito Funcional
17	Gerência de Prontuários	Item 5 no módulo Odontopediatria	Caso a resposta de algum item da Avaliação de Higienização seja “Sim”, o módulo deverá mostrar um campo para que possa ser especificada a frequência com que o paciente faz a higienização.	Requisito Funcional
18	Gerência de Prontuários	Item 5 no módulo Odontopediatria	O módulo deve mostrar na Avaliação de Higienização um campo para que seja especificada como é a colaboração do paciente quando a higienização é feita, com as opções de “Colaborador” e “Não Colaborador”.	Requisito Funcional
19	Gerência de Prontuários	Item 6 no módulo Odontopediatria	O módulo deverá incluir, como 6º item da página, os itens para avaliação do Risco de Cárie nos dentes do paciente.	Requisito Funcional
20	Gerência de Prontuários	Item 6 no módulo Odontopediatria	Caso a resposta de algum item da avaliação de Risco de Cárie seja “Sim”, o módulo não deve mostrar outros campos para mais informações.	Requisito Funcional
21	Gerência de Prontuários	Item 6 no módulo Odontopediatria	O módulo deve mostrar na avaliação de Risco de Cárie um campo para que sejam escolhidas as características dentárias do paciente, também com a opção de “Outros” para a especificação de outras características.	Requisito Funcional
22	Gerência de Prontuários	Item 7 no módulo Odontopediatria	O módulo deverá incluir, como 7º item da página, a avaliação de Experiência de Cárie Dentária do paciente.	Requisito Funcional
23	Gerência de Prontuários	Item 7 no módulo Odontopediatria	Caso a resposta de algum item da avaliação de Experiência de Cárie Dentária seja “Sim”, o módulo não deve mostrar outros campos para mais informações.	Requisito Funcional

Nº	Contexto Geral	Contexto Específico	Descrição	Classificação
24	Gerência de Prontuários	Item 7 no módulo Odontopediatria	O módulo deve apresentar na avaliação de Experiência de Cárie Dentária os itens de CPOD, CPOS e CEO. Cada um destes itens deverá apresentar campos de texto para descrição de suas características.	Requisito Funcional
25	Gerência de Prontuários	Item 8 no módulo Odontopediatria	O módulo deverá incluir, como 8º item da página, o Índice de Higiene Oral Simplificado (IHOS) do paciente.	Requisito Funcional
26	Gerência de Prontuários	Item 8 no módulo Odontopediatria	O módulo deve apresentar no IHOS os itens de Dentição Mista e Dentição Permanente. Cada um destes itens deverá apresentar campos de texto para descrição de suas características.	Requisito Funcional
27	Gerência de Prontuários	Item 8 no módulo Odontopediatria	O módulo deve apresentar no IHOS os itens de Higiene Oral, permitindo sua classificação.	Requisito Funcional
28	Gerência de Prontuários	Item 8 no módulo Odontopediatria	O módulo deve armazenar em uma tabela no final do IHOS todas as avaliações feitas pelo dentista correspondente a Higiene Oral do paciente.	Requisito Funcional
29	Gerência de Prontuários	Item 5 no Índice de Higiene Oral do módulo Odontopediatria	O módulo deverá manter listado todo o histórico do paciente, como 5º item, no Índice de Higiene Oral.	Requisito Funcional
30	Gerência de Prontuários	Item 9 no módulo Odontopediatria	O módulo deverá incluir, como 9º item da página, a avaliação de Traumatismo do paciente.	Requisito Funcional
31	Gerência de Prontuários	Item 9 no módulo Odontopediatria	O módulo deve apresentar no item de avaliação de Traumatismo os campos para especificação das características do traumatismo dentário sofrido pelo paciente.	Requisito Funcional
32	Gerência de Prontuários	Item 9 no módulo Odontopediatria	No item de Traumatismo o módulo deve apresentar os campos de Fratura Coronária e Fratura Redicular, para que o dentista faça a avaliação específica da fratura.	Requisito Funcional
33	Gerência de Prontuários	Resultados finais no módulo Odontopediatria	O módulo deve incluir no final da página, dois campos para que sejam especificados os resultados do tratamento e da evolução do paciente.	Requisito Funcional

Nº	Contexto Geral	Contexto Específico	Descrição	Classificação
34	Gerência de Prontuários	Opções de conformidade no módulo Odontopediatria	Para o módulo de Odontopediatria todos os itens devem obedecer ao Prontuário Dentário Único (PDU) que ainda é utilizado nos atendimentos do COU.	Requisito Não Funcional
35	Gerência de Prontuários	Opções de salvamento no módulo Odontopediatria	No final da página, o módulo deve apresentar os botões de “Salvar” para que as informações descritas sejam salvas no Banco de Dados e “Cancelar” que voltará para a página de Gerência de Prontuários.	Requisito Funcional

Aponta-se também que, em seu trabalho, [34] já havia identificado o resultado aqui exposto, assim pudemos reproduzi-lo neste estudo de caso.

5.2.3 Número de Palavras Reutilizadas

Na análise qualitativa desta métrica apresenta-se no Quadro 5.8 um comparativo entre as palavras reutilizadas nos módulos Odontopediatria e Odontogeriatria. Ressalta-se que o significado de cada palavra aqui mostrada, é que foi utilizada pelo menos uma vez em algum outro requisito.

Quadro 5.8 - Palavras reutilizadas na descrição

Letras	Módulo Odontopediatria	Módulo Odontogeriatria
#	3, 4, 5, 6.	7.
A	apresentar, algum, a, as, ao, avaliação, atendimentos, ainda, água, alimentação.	a, apresentar, aparecer, apagado, armazenar, atendimento, atenção, ao, avalie, através, as, anotar.
C	como, cada, cárie, caso, COU, campos, com, campo, consumo, colaborador, características.	com, cálculo, campo, cliente, campos, comando, corretamente, comandos, conhecimento.
D	de, deverá, dia, da, deve, dentária, do, descrição, destes, dentição, dentista, dentário.	de, deverá, deve, do, das, da, devem, diga, deseja, desenho, dentista, dos.
E	e, é, experiência, especificar, especificação.	em, existir, escrita, é, e, ele, escrito, escrever, evocação, executado.

Letras	Módulo Odontopediatria	Módulo Odontogeriatria
F	faz, final, fratura.	final, feitos, frase, fez.
G	gerência.	
H	histórico, hábitos, higienização, higiene.	
I	incluir, item, informações, infância, itens, IHOS, índice.	irá, itens, item, imediata.
L		linguagem, local.
M	módulo, mostrar, meses, mais.	memória, marque.
N	no, não, na, nos.	na, no, nome.
O	o, os, ou, opção, odontopediatria, outros, opções, oral.	o, os, odontogeriatria, orientação, objetos.
P	página, para, prontuários, paciente, pelo, por.	página, paciente, para, que, permitir, pontos, pelo.
Q	quando, quantos, que.	qual.
R	reposta, referentes, risco.	resposta, respostas, resultado, recorde.
S	ser, sua, sim, seja, sejam, suas.	sistema, ser, sejam, se.
T	todos, tem, texto, traumatismo.	teste, todos, toda, testes, três, ter, texto.
U	um, usa.	um, uma.

Algumas destas palavras foram reutilizadas muitas vezes dentro dos requisitos especificados, enquanto outras tiveram duas utilizações apenas, mas nos dois casos considera-se a reutilização, pois atingiu-se o objetivo de fornecer ao auxílio ao ER na especificação dos requisitos.

5.2.4 Número de Palavras Não Reutilizadas

Esta métrica é responsável por demonstrar a quantidade de palavras não reutilizadas, isto é, as palavras aqui apresentadas foram utilizadas **somente uma vez**, quando foram descritas, e foram incluídas no BDHR quando estes requisitos foram especificados. No Quadro 5.9 apresenta-se um comparativo com todas as palavras não reutilizadas na descrição dos requisitos.

Quadro 5.9 - Palavras não reutilizadas na descrição

Letras	Módulo Odontopediatria	Módulo Odontogeriatria
#		100.
A	apresentado, à, atendimento, apagadas, avaliações, armazenadas, anteriormente, alimenta, armazenar.	atendimentos, armazenada, algum, armazenadas, à, avaliados, armazenados, atribuídos, assim, armazene, avaliar, anote, armazenagem, avaliação.
B	botões, banco.	botões.
C	colaboração, CPOD, CPOS, CEO, classificação, correspondente, coronária, cancelar.	clicar, consultas, consulta, cancelar, conseguiu, copiar, COU, completos, certo, checkboxes, começando, contrário.
D	doença, duas, dentro, deverão, doenças, durante, dentes, dentárias, dois, descritas, dados.	data, deu, dia, dizer, dentário, deverão, desenhou, dentária, denominação, diminuindo.
E	escolhida, escolhido, escolhidas, específica, evolução.	está, esqueça, executou, estejam, escute.
F	frequência, feitas, faça.	fichas, feito, facilitar.
G		gerência.
H	hábito.	horário, haverá.
I	início, incrementado, indicará.	início, inicial, idênticos, irão.
L	listado.	língua.
M	modelo, mista, manter.	mudanças, mundo.
N	nenhum.	não, nada, nomes, neste.
O	ocorridas, ocorreu, outras, obedecer.	onde, obtidos, ocorram, observar.
P	parou, possa, PDU, permanente, permitindo, prontuário.	prontuários, prontuário, passado, padrão, pacientes, poderão, PDU, pedir, perguntar, portuguesa, próprios, palavra, papel, passar.
Q	quantas, quatro.	quando.
R	realizados, realizado, redicular, resultados.	repetição, recordação, reavaliação, repetir, responde, recordar, repita, realiza-los.
S	selecionado, simplificado, sofrido, salvar, salvas.	separadamente, semana, salvar, separados, subtração, soletrar, sobre.
T	tinha, também, tabela, todas, tratamento.	total.
U	único.	único, utilizado, usados, utilizados, unidades.
V	vezes, voltará.	vai, vários, verbal.

Como apontado nas análises, percebe-se que, quando utilizado o modelo de processo proposto, ocorreu um aumento na quantidade de palavras reutilizadas e uma diminuição nas palavras não reutilizadas. Em contrapartida, quando o modelo de processo não foi utilizado essa situação se inverteu.

5.3 Considerações Finais

Por fim reforça-se que, o objetivo deste estudo de caso era verificar o aumento na qualidade da descrição dos requisitos através da utilização do modelo de processo proposto e a diminuição da ocorrência de requisitos despadronizados.

Os requisitos especificados utilizando o modelo de processo proposto obtiveram uma melhor descrição, pois estavam baseados em palavras já utilizadas e validadas pelo cliente, possibilitando sua padronização, enquanto os requisitos especificados sem a ajuda do modelo proposto ficaram dependentes do conhecimento e experiência do ER responsável, confirmando o que já havia sido citado em [1][3][4].

A seguir apresenta-se um resumo com os dados obtidos pela análise quantitativa de dados, demonstrando alguns benefícios obtidos pela utilização do modelo de processo proposto neste trabalho.

1. **Requisitos aprovados pelo Cliente**: Aumento de 22% de aprovação de requisitos na 1ª iteração e 30% na 2ª Iteração;
2. **Problemas de Descrição**: diminuição de 21% na quantidade de requisitos com problemas de descrição;
3. **Ambiguidade de Contexto**: diminuição de 17% com relação a problemas de ambiguidade de contexto nos requisitos;
4. **Número de Palavras Reutilizadas**: aumento de 7% na reutilização de palavras que compõem os requisitos;
5. **Número de Palavras Não Reutilizadas**: diminuição de 7% na quantidade de palavras não reutilizadas.

6 CONCLUSÕES

Este trabalho apresentou o esforço existente para que a descrição, documentação e o reuso dos requisitos estejam com maior nível de qualidade, utilizando processos e técnicas para que cliente e Engenheiro de Requisitos possam ficar seguros de que o requisito criado corresponde exatamente ao requisito desejado.

A Engenharia de Requisitos é uma das primeiras fases do desenvolvimento de software, talvez a mais importante, pois através dela o software é definido. Portanto, todo o auxílio para que os requisitos estejam em conformidade com o desejo do cliente é de grande valia tanto para analistas, engenheiros e desenvolvedores quanto para a organização em si.

Dentro da literatura de Engenharia de Requisitos existem diversos processos e técnicas que buscam prover auxílio e aumento da qualidade na Especificação de Requisitos. Assim, utilizou-se uma comparação entre alguns trabalhos que tratavam da qualidade de requisitos para reforçar que a padronização da escrita de requisitos e a utilização de contextos para a separação e grupamento de requisitos são itens que podem ser explorados em pesquisa e no desenvolvimento de software.

O modelo de processo proposto apresentado visa atender estes itens, provendo os benefícios citados, melhorando a padronização da escrita dos requisitos, diminuindo a quantidade de retrabalho necessário nas fases posteriores do desenvolvimento e minimizando a chance de inconsistências na descrição.

O estudo de caso apresentado mostrou que é possível obter-se melhorias qualitativas no tempo de Especificação de Requisitos, diminuição de problemas na descrição e ambiguidade dos requisitos, e na reutilização de palavras que compõem os requisitos. Conseguiu-se demonstrar também a possibilidade de melhorias quantitativas significativas nos mesmos aspectos utilizados no estudo de caso.

6.1 TRABALHOS FUTUROS

Como continuação do trabalho, existem algumas frentes que poderão ser desenvolvidas para que ocorra a melhoria e evolução do estudo feito até agora.

O desenvolvimento e a inclusão no processo de um DRS padrão para armazenar a especificação, melhorando a documentação de requisitos proposta pelo processo, bem como prover um alinhamento entre o modelo de processo e a Gerência de Requisitos,

para que, além de tratados, os requisitos possam ser gerenciados de acordo com os níveis de qualidade esperados.

Para a avaliação contínua do modelo de processo, a criação e o desenvolvimento de outros estudos de caso seria algo de extrema importância. Não somente pelo fato de aumentar a validação do modelo de processo, mas também para obter outros dados, em outras situações, que poderiam influenciar determinadas fases ou passos específicos do processo, assim conseguiríamos trabalhar e evoluir estes pontos específicos.

Os estudos de caso poderiam abordar agora algum processo já difundido na literatura de Engenharia de Requisitos, para que quando comparados, o modelo de processo proposto possa ter suas características comparadas não somente na teoria, mas também na prática, dentro da indústria de software.

Outra frente de desenvolvimento muito importante diz respeito à criação e desenvolvimento de uma ferramenta CASE que implemente os conceitos apresentados neste trabalho, utilizando ajuda computacional para aumentar os benefícios obtidos. Esta ferramenta ainda auxiliaria no Gerenciamento de Requisitos para que o tratamento oferecido seja ampliado e com isso consigamos aumentar ainda mais a qualidade com que os requisitos são tratados.

Neste sentido, a continuação do trabalho deve abranger a área de Engenharia de Requisitos, utilizando o modelo de processo, buscando melhorar o que já é oferecido e preencher as lacunas deixadas pelas outras ferramentas.

REFERÊNCIAS

[1] PRESSMAN, R. S. *Engenharia de Software - Uma Abordagem Profissional*. 7 ed. McGraw-Hill, 2011.

[2] SOMMERVILLE, I. *Engenharia de Software*. 8 ed. Addison Wesley, 2007.

[3] SOMMERVILLE, I.; SAWYER P. *Requirements Engineering: A Good Practice Guide*. 1 ed. Wiley, 1997.

[4] ROBERTSON, S.; ROBERTSON, J. *Mastering the Requirements Process*. 2 ed. Addison Wesley, 2006.

[5] WIEGERS, K. E. *Software Requirements*. 2 ed. Microsoft Press, 2003.

[6] IEEE 830: Recommended Practice for Software Requirements Specification, http://ieeexplore.ieee.org/xpls/abs_all.jsp?isnumber=15571&arnumber=720574&count=1&index=0, último acesso: Julho de 2012, 1998 (R2009).

[7] IEEE 1233a-1998: IEEE Guide for Developing System Requirements Specifications, http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=741940&c ontentType=Standards, último acesso: Julho de 2012, 1998.

[8] RUNESON, P.; HÖST, M. Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering 14* (2), April, pp 131-164. doi=10.1007/s10664-008-9102-8, 2009.

[9] KITCHENHAM, B.; et al. Evaluating guidelines for reporting empirical software engineering studies. *Empirical Software Engineering 13* (1), pp 97-121 doi:10.1007/s10664-007-9053-5, 2008.

[10] POHL, K. *Requirements Engineering: Fundamentals, Principles, and Techniques*. 1 ed. Springer, 2010.

[11] LAMSWEERDE, A. van. *Requirements Engineering: From System Goals to UML Models to Software Specifications*. 1 ed. Wiley, 2009.

[12] THAYER, R. H.; DUNCAN, M.; DAVIS, A. M. *Software Requirements Engineering*. 2 ed. IEEE Press, 2000.

[13] LUTOWSKI, R. *Software Requirements: Encapsulation, Quality, and Reuse*. 1 ed. Auerbach Publications, 2005.

[14] LAUESEN, S. *Software Requirements: Styles & Techniques*. 1 ed. Addison-Wesley, 2002.

[15] AURUM, A. (E.); WOHLIN, C. (E.). *Engineering and Managing Software Requirements*. 1 ed. Springer, 2005.

[16] BOOCH, G.; RUMBAUGH, J.; JACOBSON, I. *UML: Guia do Usuário*. 2 ed. Elsevier, Rio de Janeiro, 2005.

[17] SAYÃO, M. *Verificação e Validação em Requisitos: Processamento da Linguagem Natural e Agentes*. 2007. 205 f. Tese (Doutorado em Informática) – Pontifícia Universidade Católica do Rio de Janeiro (PUC-Rio), Rio de Janeiro, 2007.

[18] POHL, K.; RUPP, C. *Requirements Engineering Fundamentals: A Study Guide for the Certified Professional for Requirements Engineering Exam*. 1 ed. Rocky Nook Computing, 2011

[19] RENAULT, S.; BONILLA, O. M.; FRACH, X. PABRE: Pattern-Based Requirements Elicitation. *Third International Conference On Research Challenges in Information Science*, Fez, Morocco, April 22-24 2009.

[20] KNETHEN, A. V.; et al. Systematic Requirements Recycling through Abstraction and Traceability. *RE - Requirements Engineering*. pp 273-281, Essen: Germany, 2002.

[21] SILVA, R. C. *Uma Abordagem para o Reuso de Requisitos baseada em Padrões e Rastreabilidade*. 2011. 247 f. Dissertação (Mestrado em Computação Aplicada) – Universidade do Vale do Itajaí, Santa Catarina, 2011.

[22] SHEHATA, M. S.; EBERLEIN, A.; HOOVER, H. J. Requirements Reuse and Feature Interaction Management. *ICSSEA - International Conference on Software \& Systems Engineering and their Applications*. Paris: France, 2002.

[23] KOTONYA G.; SOMMERVILLE, I. *Requirements Engineering: Processes and Techniques*. 1 ed. Wiley, 1998.

[24] DAVIS, A. M. *Just Enough Requirements Management: Where Software Development Meets Marketing*. 1 ed. Dorset House, 2005.

[25] BERENBACH, B.; PAULISH D. J.; KAZNEIER J.; RUDORFER A. *Software & Systems Requirements Engineering: In Practice*. 1 ed. McGraw-Hill, 2009.

[26] CYBULSKY, J.; REED, K. Requirements Classification and Reuse: Crossing Domains Boundaries. *6th International Conference on Software Reuse*. pp 190-210, Viena: Italy, 2000.

[27] NUSEIBEH, B. Weaving Together Requirements and Architectures. *Computer* (IEEE) 34 (3), pp 115-117, 2001.

[28] DAVIS, A. M.; HICKEY A. M. A New Paradigm for Planning and Evaluating Requirements Engineering Research. *Second International Workshop on Comparative Evaluation in Requirements Engineering*, Kyoto, Japan, pp 7–16, 2004.

[29] FERREIRA, D. A.; SILVA, A. R. A Controlled Natural Language Approach for Integrating Requirements and Model-Driven Engineering. *ICSEA - International Conference on Software Engineering Advances*, pp. 518-523, Porto – Portugal, 20-25 Setembro, 2009.

[30] MOROS, B.; VICENTE-CHICOTE, C.; TOVAL, A. Metamodeling Variability to Enable Requirements Reuse. *EMMSAD - Exploring Modeling Methods for Systems Analysis and Design*, Montpellier: France, June 16-17 2008.

[31] CHEN, H.; et al. Text-based requirements preprocessing using nature language processing techniques. *International Conference on Computer Design and Applications (ICCDA)*, pp 14-18, Qinhuangdao, Hebei, China, July 25-27 2010.

[32] CABRAL, M. S.; et al. Aplicação de Técnicas de Leitura durante a Análise de Requisitos. *WER - Workshop em Engenharia de Requisitos*, pp 193-204, Barcelona, Catalonia, Spain, September 12-13 2008.

[33] PANDEY, D.; RAMANI, A. K.; SUMAN, U. An Effective Requirement Engineering Process Model for Software Development and Requirements Management. *International Conference on Advances in Recent Technologies in Communication and Computing*, IEEE Press 2010.

[34] LAM, W.; MCDERMID, T. A.; VICKERS, A. J. Ten steps towards systematic requirements reuse. *Proceedings of the Third IEEE International Symposium on Requirements Engineering*, pp 6-15, 1997.

Trabalhos publicados pelo Autor

1. DORIGAN, J. A.; BARROS, R. M., **A model of Requirements Engineering Process for Standardization and Quality Increase**, International Conference on Applied Computing – IADIS, Madri: Espanha, Outubro de 2012, pp. 343-347, ISBN: 978-989-8533-14-2. (Qualis CC 2012 – B4)

2. DORIGAN, J. A.; BARROS, R. M., **Requirements Engineering: A Process Model and Case Study to Promote Standardization and Quality Increase**, International Conference on Software Engineering Advances – ICSEA, Lisboa: Portugal, Novembro de 2012, pp. 499-505, ISBN: 978-1-61208-230-1. (Qualis CC 2012 – B3)