UNIVERSIDADE ESTADUAL DA PARAÍBA
PRÓ-REITORIA DE PÓS-GRADUAÇÃO E PESQUISA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIAS E TECNOLOGIA EM SAÚDE

JULIA CIBELLE FREIRE DE QUEIROZ ARNAUD

RE4CH: Engenharia de Requisitos para Saúde Conectada: Lidando com Práticas Ágeis e Rastreabilidade

Campina Grande/PB
Março de 2017

JULIA CIBELLE FREIRE DE QUEIROZ ARNAUD

RE4CH: Engenharia de Requisitos para Saúde Conectada: Lidando com Práticas Ágeis e Rastreabilidade

Dissertação de Mestrado submetida à banca de avaliação do Programa de Pós-Graduação em Ciência e Tecnologia em Saúde da Universidade Estadual da Paraíba.

Orientador: Prof. Dr. Paulo Eduardo e Silva Barbosa

Campina Grande/PB

Março de 2017

É expressamente proibida a comercialização deste documento, tanto na forma impressa como eletrônica. Sua reprodução total ou parcial é permitida exclusivamente para fins acadêmicos e científicos, desde que na reprodução figure a identificação do autor, título, instituição e ano da dissertação.

A744r Arnaud, Julia Cibelle Freire de Queiroz.
Re4ch: Engenharia de requisitos para saúde conectada [manuscrito] : lidando com práticas ágeis e rastreabilidade / Julia Cibelle Freire de Queiroz Arnaud. - 2017.
85 p. : il.

Digitado.
Dissertação (Mestrado Profissional em Ciência e Tecnologia em Saúde) - Universidade Estadual da Paraíba, Pró-Reitoria de Pós-Graduação e Pesquisa, 2017.
"Orientação: Prof. Dr. Paulo Eduardo e Silva Barbosa, Pró-Reitoria de Pós-Graduação e Pesquisa".

1. Sistemas de Informação em Saúde. 2. Engenharia de requisitos. 3. Rastreabilidade. 4. Processo Re4ch. I. Título.
21. ed. CDD 005.3

JULIA CIBELLE FREIRE DE QUEIROZ ARNAUD

RE4CH: Engenharia de Requisitos para Saúde Conectada: Lidando com Práticas Ágeis e Rastreabilidade

Dissertação de Mestrado submetida à banca de avaliação do Programa de Pós-Graduação em Ciência e Tecnologia em Saúde da Universidade Estadual da Paraíba.

Aprovado em 31/03/2017

BANCA EXAMINADORA

Orientador: Prof. Dr. Paulo Eduardo e Silva Barbosa

Orientador UEPB

Orientador: Prof. Dr. Alisson Vasconcelos de Brito

Examinador UFPB

Orientador: Prof. Dr. Fernando Menezes Matos

Examinador UFPB

AGRADECIMENTOS

Agradeço e dedico este trabalho a Deus por me dar força, fé e perseverança para superar todos os obstáculos para realização deste trabalho. A todo instante ele estava comigo!

Agradeço ao meu esposo, José Arnaud, por sempre estar ao meu lado, me apoiando incansavelmente em todos os momentos, por todo o amor e carinho tornando meus dias mais leves.

A minha família que mesmo passando por momentos tão difíceis, não me deixou faltar atenção e paciência. Em especial a minha mãe por todas as preces.

Agradeço ao professor Dr. Paulo Eduardo e Silva Barbosa, pela oportunidade, além de me motivar e apoiar na realização deste trabalho.

Ao Núcleo de Tecnologias Estratégicas em Saúde (NUTES), pelo ambiente oferecido para a realização deste trabalho.

Agradeço a dedicação da equipe do Laboratório de Instrumentação Biomédica e Ensaios Eletrônicos (LIBE), que participou da avaliação deste trabalho. A Alex Figueiredo por toda colaboração.

Ao meu amigo Túlio Henriques que tanto contribuiu ao longo dessa jornada com críticas e sugestões que foram muito importantes.

Aos queridos amigos Renata, Nívian, Fernanda, Joanna, Eliclenes pela atenção e palavras de ânimo. Obrigada pela amizade de vocês! Aos demais amigos que fiz durante esta caminhada.

“O êxito da vida não se mede pelo caminho que você conquistou, mas sim pelas dificuldades que superou no caminho”

Abraham Lincoln

RESUMO

O número de sistemas de informação em saúde está aumentando em quantidade e complexidade, e sua qualidade é essencial para melhorar a qualidade dos serviços prestados à população. O desenvolvimento de software para dispositivo médico apresenta requisitos que normalmente não são abordados pelo desenvolvimento de software tradicional. Neste sentido, os projetos de sistemas de informação de saúde começam a enfrentar novos desafios impostos pelos reguladores, sociedade e fornecedores, uma vez que começam a interagir com os dispositivos médicos, dispositivos pessoais de saúde e soluções de diagnóstico. Os processos de engenharia de requisitos são muito importantes para a redução dos riscos envolvidos no desenvolvimento destes sistemas. Muitos dos processos ágeis utilizados por várias empresas, apesar de seguir um ciclo de atividades para o desenvolvimento do software, apresentam algumas lacunas nas atividades da engenharia de requisitos, tornando a forma com que os requisitos são desenvolvidos no contexto para saúde muitas vezes incompleta. O desenvolvimento de sistemas na área da saúde deve ser apoiado por um processo de engenharia de requisitos apropriado, que além de guiar processo de desenvolvimento, também se encaixe no tratamento de requisitos específicos que estes produtos de software necessitam. Este trabalho propõe a definição de um processo de engenharia de requisitos que possibilite o desenvolvimento e gerenciamento dos requisitos de maneira rastreável, ao mesmo tempo que se alinha ao processo de desenvolvimento ágil. O processo define atividades, entradas, saídas e diretrizes que de forma integrada, orientam a engenharia de requisitos de maneira eficaz. Finalmente é apresentado um estudo experimental no contexto de sistemas de informação em saúde para avaliar a viabilidade do processo proposto.

Palavras-Chave: Sistemas de Informação em Saúde, Engenharia de Requisitos, Rastreabilidade, Ágil.

ABSTRACT

The number of health information systems is increasing in quantity and complexity, and their quality is essential to improve the quality of the services delivered to the population. Normally, the development of medical device software presents requirements that normally are not addressed by the traditional software development of non-safety critical industry. In this sense, projects of health information systems start to face new challenges imposed by regulators, society and suppliers since they start to interact with medical devices, personal health devices and diagnostic solutions. Requirements engineering processes are very important for the reduction of the involved risks in the development of these systems. Most agile processes used by several companies, although following well established activities for software development, have some gaps in the requirements engineering activities, turning the way in which requirements are developed in the context for health often incomplete. Health system development should be supported by an appropriate requirements engineering process, which in addition to guiding the development process, also fits in the treatment of specific requirements that these software products need. This work proposes the definition of a requirements engineering process that enables the development and management of requirements in a traceable way, while at the same time aligning with the agile development process. The process defines activities, inputs, outputs and guidelines that, in an integrated way and guides the requirements engineering in an effective way. Finally, an experimental study is presented in the context of health information systems to evaluate the viability of the proposed process.

Keywords: Health Information Systems; Requirements Engineering; Traceability; Agile.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

São cada vez mais crescentes os investimentos em desenvolvimento de software para saúde em todo o mundo. Na última década, o uso da Tecnologia da Informação (TI) nesta área tem assumido um papel cada vez maior com a tendência dos sistemas de informação, atuando principalmente na automatização de processos clínicos e administrativos hospitalares. Dessa maneira, os benefícios são presumíveis desde o atendimento ao paciente, como também, permitindo um acesso mais fácil aos recursos sociais e de saúde [COO08].

Com o crescimento do número de aplicações na área de saúde aumentou-se a complexidade informacional, pois cada vez mais registros são distribuídos em diferentes bases de dados abrindo a possibilidade para a pesquisa e proposta de novas tecnologias de informática em saúde [SAN10].

Ao mesmo tempo, esta tendência também deu origem a novos desafios relacionados com a implementação de novas tecnologias utilizadas nos cuidados da saúde, como é o caso dos dispositivos médicos, que estão desempenhando um papel cada vez mais importante [ZEM15]. A indústria de dispositivos médicos também é uma área de crescimento constante [DEN07]. Por sua natureza crítica, o desenvolvimento de software para dispositivos médicos requer exigências que normalmente não são enfrentados pelos desenvolvedores de software tradicional, como atender às regulamentações específicas, tornando-se essencial para organizações alcançarem a conformidade e aprovação pelo órgão regulador do país onde desejam comercializar os seus dispositivos médicos [AXE11]. Muitos dispositivos médicos devem fazer interface com outros equipamentos, como também conectar a sistemas de informação hospitalares e laboratoriais e isso tem aumentado a complexidade desses sistemas fazendo com que desenvolvedores enfrentem crescente pressão para entregar software de alta qualidade com funcionalidades adicionais, em curtos prazos e economicamente viáveis.

Com o advento dos atuais sistemas, denominados sistemas ciber-físicos, que são formados por soluções que vão desde sistemas de informação, tecnologias móveis e sistemas embarcados vêm propiciando valor agregado na área de saúde [HAQ14]. As empresas que tradicionalmente desenvolvem sistemas de informação e fazem uso de metodologias ágeis, passam a enfrentar novos desafios impostos por reguladores, sociedade, fornecedores. Isto se dá devido a necessidade de inovação, gerando demandas por atributos de qualidade, tais como segurança, privacidade, disponibilidade, rastreabilidade entre outros. Estes atributos só podem

ser inseridos com maior rigor mediante a adoção de práticas existentes em processos mais pesados, usados pela indústria de sistemas críticos, tal como a saúde.

É muito comum em empresas e projetos que desenvolvem softwares para dispositivo médico seguirem planos sequenciais para a gestão do ciclo de vida do software, tais como o modelo V, a fim de mostrar a garantia dos requisitos a agências reguladoras. A idéia principal desse tipo de modelo é seguir uma sequência rigorosa de passos definidos para a análise e especificação de requisitos, arquitetura de software e design, codificação e teste [MCC05]. No entanto, verificou-se em muitos cenários que as empresas estão enfrentando dificuldades para seguir modelos sequenciais, especialmente na área de engenharia de requisitos. Normalmente quando uma mudança é introduzida uma série de etapas precisam ser revistas para acomodar uma alteração, exigindo uma grande quantidade de retrabalho, aumentando tempo e custo de desenvolvimento [MCH13].

Por outro lado, as metodologias ágeis são alternativas ao gerenciamento de projetos tradicional, bastante popular entre desenvolvedores de sistemas de informação convencionais por ajudar a lidar com as incertezas por parte dos clientes e os recursos reduzidos das empresas através de iterações incrementais [SER15]. No entanto, esse tipo de metodologia, embora assegure a agilidade nas suas atividades, apresentam algumas lacunas durante as atividades de Engenharia de Requisitos (ER), se usado para áreas complexas, tais como sistemas críticos de segurança. Isto torna a forma com que os requisitos são desenvolvidos e gerenciados ainda bastante incompletos [INA15].

Por exemplo, tomemos a questão da rastreabilidade, que é um requisito de qualidade reconhecidamente importante na comunidade de engenharia de software e por várias indústrias onde a qualidade é fundamental para garantia de integridade e segurança de seus usuários. Muitas agências reguladoras têm reconhecido a sua importância e têm incorporado rastreabilidade em suas diretrizes e normas. Por exemplo, a norma IEC 62304, que lida com processos de software para dispositivos médicos, requer “rastreabilidade entre os requisitos do sistema, requisitos de software, testes do sistema e teste do software, e medições de controle de riscos implementadas no software” [IEC62304-2006]. Além disso, a norma requisita que “o fabricante deve verificar e documentar que os requisitos do software são rastreáveis aos requisitos de sistema ou outras fontes”.

1.1 Relevância

Nesse contexto, o desenvolvimento de sistemas na área da saúde deve ser apoiado por um processo de ER apropriado, que além de guiar o processo de desenvolvimento, também se encaixe no tratamento de requisitos específicos que estes produtos de software necessitam. Desse modo, mostra-se necessária uma maneira de além de identificar, especificar e validar os objetivos dos clientes de acordo com suas necessidades, dar suporte ao gerenciamento de requisitos. Neste sentido, deve ser levado em consideração o atributo rastreabilidade, o qual apoia o gerenciamento de mudanças corroborando na conformidade do software com os seus requisitos. Manter a rastreabilidade dos requisitos além de ser de grande utilidade para o gerenciamento dos requisitos, é também exigida por padrões de requisitos, normas de segurança e modelos de maturidade [REM15].

A adoção de práticas ágeis aplicadas na ER podem compensar algumas deficiências dos métodos tradicionais de ER. Uma delas está relacionada com a flexibilidade no gerenciamento de mudança dos requisitos, que podem surgir durante o ciclo de vida do desenvolvimento, de uma maneira que seja possível acomodar alterações sem aumento de custo e retrabalho demasiado [INA15]. O feedback constante do cliente, a forma como os requisitos são detalhados e especificados gradualmente na interação entre o cliente e a equipe de desenvolvimento, faz com que os requisitos possam ser identificados e acordados de forma mais eficiente [BJA11].

Diante dos problemas supracitados faz-se necessário um processo que se adeque a este contexto, mantendo um equilíbrio adequado entre as características de uma documentação mínima destinada a processos ágeis e as características de uma documentação mais rigorosa dos processos de engenharia de requisitos sequenciais.

1.2 Objetivo do Trabalho

Este trabalho tem como principal objetivo definir um processo de engenharia de requisitos, com o propósito de alcançar uma documentação mais completa dos requisitos, a fim de lidar com a rastreabilidade dos artefatos gerados, ao mesmo tempo que se alinha ao processo de desenvolvimento ágil. O processo é composto por atividades, entradas e saídas que de forma integrada, orientam a engenharia de requisitos de maneira eficaz. Para alcançar o objetivo proposto as seguintes etapas foram definidas:

1. Analisar técnicas e metódos existentes utilizadas na engenharia de requisitos tradicional e ágil;

2. Definir um processo de engenharia de requisitos que possibilite o desenvolvimento e gerenciamento dos requisitos de maneira rastreável;
3. Planejar e definir um estudo experimental aplicando o processo proposto;
4. Avaliar os resultados do experimento realizado.

1.3 Estrutura da Dissertação

As próximas partes da dissertação estão estruturadas da seguinte forma:

Capítulo 2: Este capítulo apresenta os conceitos básicos necessários para melhor compreensão deste trabalho. Os conceitos descritos estão relacionados com sistemas de informação em saúde, engenharia de requisitos tradicional e ágil, como também a rastreabilidade de requisitos.

Capítulo 3: Este capítulo descreve o processo de engenharia de requisitos proposto, apresentando as atividades, entradas, saídas e diretrizes para uso do processo.

Capítulo 4: Este capítulo apresenta um estudo experimental realizado através de um estudo de caso utilizando o processo proposto no desenvolvimento de um sistema de informação para saúde.

Capítulo 5: Este capítulo apresenta a avaliação experimental do processo proposto.

Capítulo 6: Este capítulo apresenta os principais trabalhos relacionados à esta pesquisa.

Capítulo 7: Este capítulo apresenta a conclusão deste trabalho através dos resultados obtidos e propõe trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem como objetivo fornecer embasamento teórico sobre conceitos necessários para melhor compreensão deste trabalho. Serão introduzidos os princípios de Sistemas de Informação em Saúde, Engenharia de Requisitos e Rastreabilidade de Requisitos, assuntos relevantes para esta pesquisa.

2.1 Sistemas de Informação em Saúde

Sistemas de Informação em Saúde (SIS) são essenciais para a tomada de decisão em todo o setor da saúde, quando estes dispõem de informação precisa e acessível, permitem gerir recursos humanos e financeiros, priorizar os problemas de saúde, formular políticas de saúde entre outros benefícios [CAS09].

Conceitualmente, um SIS é um conjunto de informações (dados), processos, pessoas e tecnologia da informação que interagem para coletar, processar, armazenar e fornecer como saída a informação precisa para apoiar as organizações de cuidados com a saúde. Os Sistemas de Informação em Saúde são reconhecidos como instrumentos que além de aumentar a efetividade dos profissionais, reduzem os custos com saúde, e promovem a padronização do cuidado com o paciente através da informação gerida [FAT10]. Eles estão divididos em duas categorias principais, as administrativas e clínicas [WAG09]. Um sistema de informação administrativo é geralmente utilizado para apoiar as funções de gerenciamento e operações gerais da organização de saúde, eles podem conter informações utilizadas para gerir pessoal, finanças, suprimentos ou equipamentos. Já um sistema de informação clínico contém informações clínicas relacionadas com a saúde do paciente que são normalmente utilizados por profissionais da saúde para diagnosticar, tratar, monitorar e gerenciar o atendimento do paciente. Exemplos de sistemas clínicos são: sistemas de apoio à decisão clínica, sistemas de administração de medicamentos, sistemas de registros médicos, como prontuário eletrônico do paciente (pep), telemedicina, entre outros [WAG09, CAR08].

Nos últimos anos as aplicações das tecnologias da informação e comunicação (TICs) no setor da saúde tornaram-se parte integrante dos cuidados de saúde com a finalidade de reestruturar e melhorar os seus custos e eficiência. A telemedicina por exemplo, é definida como a utilização de telecomunicações para diagnosticar e tratar doenças, através dela é possível, beneficiar os pacientes agindo prontamente, diminuindo custos e minimizando riscos com suas locomoções [JOR11]. O pronto atendimento em locais remotos, a transmissão de imagens e resultados de exames transmitidos que possibilitam a avaliação à distância, em áreas como radiologia, patologia, cardiologia, neurologia, entre outras, são exemplos de sua aplicação. As principais organizações internacionais como, Organização Mundial da Saúde (OMS), União Européia (UE), União Internacional das Telecomunicações (UIT) e European Space Agency (ESA) adotaram oficialmente a denominação "*e-health*" para referir-se à utilização das modernas TICs para antender uma série de aspectos dos cuidados de saúde [JOR11]. Estes aspectos vão desde registros eletrônicos médicos, prescrições eletrônicas, monitorização remota e gestão do conhecimento em saúde. A adoção de *e-health* é considerada mundialmente uma prática bem-sucedida no desenvolvimento desses cuidados por ser eficiente e centrada no paciente [BOR14]. O termo "*m-health*" também foi introduzido para incluir o uso de dispositivos móveis no acompanhamento do paciente, e transmissão de dados.

Com o avanço das tecnologias de comunicação propiciando o advento da Internet das Coisas, onde é possível interconexões inteligentes de objetos e pessoas no mundo físico interagindo e compartilhando informações, surgem muitas oportunidades de usar essa tecnologia para melhorar os resultados dos cuidados de saúde e reduzir os custos [BUI11]. Através dela é possível obter uma perspectiva apropriada para a comunicação e integração entre pessoas, dispositivos médicos, equipamentos, como sensores, tecnologias móveis, como também, a conexão com sistemas de informação hospitalares, formando os ecossitemas da saúde.

O termo ecossistema refere-se a uma comunidade composta por um conjunto de seres vivos interrelacionados e pelo ambiente em que vivem [ROJ14]. Segundo os autores de [ROJ13], em um ecossitema de saúde, devem ser incluídas características de agentes humanos, tecnológicos e socioeconômicos, bem como a interação entre eles. Os agentes humanos representam todos os envolvidos no serviço da saúde, como profissionais de saúde, pacientes, gestores, entre outros. Todos eles desempenham um papel importante no

ecossitema. Os agentes tecnológicos referem-se aos vários dispositivos médicos, softwares para a saúde, meios de transmissão e armazenamento, entre outros.

Através destes equipamentos é possível fornecer aos agentes humanos serviços de saúde mais eficientes, possibilitando melhoraria na qualidade de vida de pessoas que sofrem de doenças crônicas por exemplo, e das que precisam ser monitoradas remotamente ou até mesmo durante emergências. Os agentes socioeconômicos referem-se a relação entre os fatores humanos e econômicos que incluí a análise da infra-estrutura, da qualidade de vida e das condições de vida da população. O aspecto econômico está relacionado com o fornecimento e implementação de serviços médicos tanto em instâncias de saúde pública quanto privada, levando em conta os aspectos orçamentais da execução do serviço.

Essas interações são de grande relevância para o setor de saúde porque serão cruciais para determinar o ciclo de vida do ecossistema. Além disso, diversos fatores como, regulações, patologias, fatores humanos, entre outros, influenciam os agentes propostos que também devem ser levados em consideração na hora de desenvolver um ecossistema para garantir o funcionamento contínuo do sistema de *e-health* [ROJ14].

2.2 Engenharia de Requisitos

A Engenharia de Requisitos (ER) é uma das fases mais importantes da engenharia de software por descobrir, analisar, documentar e verificar o que um sistema deve fazer e as restrições de seu funcionamento [SOM06].

2.2.1 Engenharia de Requisitos Tradicional

A Engenharia de Requisitos tradicional é definida pelo o autor de [PRE05] como um instrumento metodológico que permite, de forma sistemática, captar necessidades e objetivos do cliente, fornecendo meios que levem à análise e soluções coerentes, além de definir uma solução não ambígua, validar a especificação e gerenciar os requisitos, ao longo do processo evolutivo de constituição de um software funcional.

O documento de requisitos é o principal artefato trabalhado na ER, pois além de servir como declaração oficial dos requisitos para os stakeholders envolvidos, como clientes, equipe

de desenvolvimento, ou usuários finais deve servir de base para todas as outras atividades de desenvolvimento de sistema acordado [SOM05].

Os requisitos são frequentemente classificados como funcionais e não-funcionais [SOM06]. Um requisito funcional especifica os serviços que o sistema deve fornecer, ou seja, descreve o que o sistema deverá fazer sem levar em consideração as restrições físicas. Os Requisitos não funcionais especificam as propriedades do sistema, tais como restrições de implementação, restrições impostas por normas, desempenho, dependências de plataforma, manutenção, confiabilidade etc.

Segundo [HAS15], estudos revelam que a principal razão do fracasso nos projetos de desenvolvimento de software é oriunda de problemas relacionados com a engenharia de requisitos. Uma engenharia de requisitos conduzida de maneira incorreta pode causar um resultado negativo em um projeto de software, afetando diretamente os custos e prazos envolvidos devido a necessidade de um novo ciclo das atividades de especificação, projeto, codificação, teste entre outros.

De acordo com os autores de [SOM98], um processo de ER descrito de maneira completa deve incluir atividades que geram, validam e gerenciam o documento de requisitos bem como, a definição de suas entradas, saídas e ferramentas utilizadas para dar suporte a ER. Um bom processo de ER conduz ao sucesso do projeto. Não existe um modelo ideal a ser seguido devido a diversos fatores tais como, cultura organizacional da empresa, disciplina, maturidade técnica, domínio da aplicação [SOM98]. No entanto, o ciclo de vida da ER tradicional é sugerido pelos autores de [SOM98] como um modelo que contemple atividades de elicitação, análise, documentação, validação, e gerenciamento dos requisitos que serão descritas a seguir.

- **Elicitação dos Requisitos**: nesta fase é dado o primeiro passo do processo de engenharia de requisitos. O principal objetivo do levantamento de requisitos é obter conhecimento sobre o problema do cliente. Os requisitos são descobertos através de conversas com diferentes partes interessadas que informam, como por exemplo, sobre funcionalidades de sistemas existentes, sistemas que deve interagir com o sistema a ser desenvolvido, regulamentações que devem ser implementadas, políticas organizacionais que devem ser seguidas. Existem várias técnicas para elicitação de requistos e compreender bem as necessidades do cliente é determinante para o sucesso da elicitação de requisitos.

- **Análise e Negociação dos Requisitos**: após um levantamento inicial dos requisitos, estes devem ser analisados e acordados entre as partes envolvidas. Os requisitos devem ser analisados para garantir a sua completude e consistência. Os requisitos devem ser entendidos de maneira completa em todos os aspectos e conflitos de entendimentos entre eles devem ser resolvidos.
- **Documentação dos Resquisitos**: nesta fase o documento de requisitos é produzido. Os requisitos são documentados em um nível apropriado de detalhe. Em geral, é produzido um documento de especificação de requisitos, de forma que todos os envolvidos possam entendê-lo. Este documento ajuda a estimar o tempo, custo e esforço para o projeto.
- **Validação dos requisitos**: é nesta fase que todas as partes interessadas concordam com o documento de especificação de requisitos. A consistência e completude do documento de requisitos são checadas de forma a assegurar que todos os requisitos estejam definidos sem ambiguidades, inconsistências ou omissões, e que todos os erros tenham sido detectados e corrigidos.

Em paralelo com estas atividades está o processo de gerenciamento dos requisitos, voltado ao gerenciamento e controle das modificações dos requisitos seja em relação a atualização, adição, exclusão ou correção dos requisitos [SOM98].

Existem várias razões pelas quais as mudanças são inevitáveis. Uma vez que um sistema tenha sido implantado e usado de maneira regular, novos requisitos surgirão, pois à medida que clientes e usuários finais vão adquirindo experiência com o sistema vão descobrindo novas necessidades e prioridades [NUS00]. O ambiente técnico e de negócios também estão sujeitos a mudanças, como por exemplo, a necessidade da inserção de novas legislações e regulamentos que o sistema obrigatoriamente tenha que respeitar, as prioridades do negócio podem mudar (com alterações necessárias para apoiar o sistema), necessidade de fazer interface com outros sistemas, etc [SOM97].

O gerenciamento de requisitos é considerado um processo complexo, pois uma determinada mudança de requisito pode resultar em um grande impacto sobre o desenvolvimento do sistema [GOT95]. Para cada mudança deve ser levado em consideração o seu impacto. Antes da aprovação de uma mudança deve ter sido avaliado se os benefícios da implementação justificam os custos e retrabalho para tal [ESP04]. O processo de gerenciamento de mudanças definem atividades que avaliam o impacto e o custo das mudanças [PAN10].

O planejamento é crucial no processo de gerenciamento de requisitos, pois além de acompanhar as alterações dos requisitos acordados acompanham também as dependências do documento de requisitos com outros documentos produzidos durante o processo de desenvolvimento do software [GOT94]. Durante o estágio de planejamento é necessário decidir sobre: Identificação de requisitos de maneira única para ser possível a comparação com outros requisitos e serem rastreáveis [SOM97]. Uma ferramenta de apoio deve ser escolhida para automatizar o gerenciamento de requisitos através do armazenamento de requisitos, gerenciamento das mudanças e gerenciamento de rastreabilidade.

O gerenciamento de requisitos é uma atividade contínua, pois os requisitos podem continuar a mudar após o desenvolvimento do software durante a manutenção [PAN10]. Os requisitos também não podem ser efetivamente gerenciados sem rastreamento. Políticas de rastreabilidade definem os relacionamentos entre os requisitos que devem ser registrados e como devem ser mantidos [SOM97].

2.2.2 Engenharia de Requisitos Ágil

Pesquisas apontam que organizações de desenvolvimento de software não regulamentado estão se beneficiando com processos ágeis [RAM09]. A maioria das organizações de desenvolvimento de software são submetidas a lidarem com questões como: rápidas mudanças no escopo do projeto, ameaças competitivas, novas tecnologias de software e pressões (time-to-market) que desafiam seriamente o desenvolvimento de sistemas baseado em requisitos pré-especificados [BOE00]. Há indícios de que utilizar práticas ágeis traz benefícios significativos para as organizações de desenvolvimento de software, tais como redução de custos, o tempo de comercialização reduzido e aumento da qualidade [MCH12].

Muitas das metodologias ágeis que abordaram esse contexto ganharam bastante atenção na prática por parte das organizações de desenvolvimento de software por incluírem práticas como, pequenas iterações, lançamentos constantes, participação do cliente no local e feedback constante do cliente. Os métodos de desenvolvimento de software ágil também aparentam ser mais adequados para acomodar as mudanças de requisitos que surgem ao longo do processo de desenvolvimento. As atividades da ER são realizadas durante todo o ciclo de vida do desenvolvimento em pequenas etapas e de maneira informal, diferentemente do

processo de engenharia de requisitos tradicional que requer que os requisitos sejam recolhidos no início do projeto [BEC99]. Métodos Ágeis permitem um feedback mais rápido aos mercados competitivos por apoiarem ciclos curtos de desenvolvimentos[RAM10].

O termo " engenharia de requisitos ágil " é considerado pelo autor de [INA15] como uma maneira de definir uma forma ágil de planejamento, execução e raciocínio sobre atividades de engenharia de requisitos. Isso porque se baseiam nas metodologias ágeis que seguem os valores e princípios do manifesto ágil [BEC01]. São eles:

- Priorizar indivíduos e interação ao invés de processos e ferramentas;
- Colaboração do cliente ao invés de negociação de contrato;
- Software executável ao invés de documentação abrangente;
- Respostas rápidas a mudanças ao invés de seguir um plano.

Estes princípios ágeis introduzem flexibilidade em relação a mudanças no escopo do projeto e surgimento de novos requisitos [INA15].

Apesar de vários benefícios incorporados pelas metodologias ágeis, ainda é baixa a taxa de adoção de práticas ágeis entre organizações de desenvolvimento de software regulamentado, como por exemplo, na área de sáude, os dispositivos médicos [MCH13]. Um desses motivos é que as práticas ágeis parecem ser contraditórias com os requisitos regulamentares [VOG06].

Por exemplo, no contexto das organizações de software de dispositivos médicos para provar que o seu dispositivo é seguro para uso são obrigados apresentarem uma documentação extensa, a fim de conseguir aprovação regulamentar, enquanto que um dos principais valores do manifesto ágil é priorizar software executável ao invés de documentação abrangente [MAR03].

Apesar das práticas ágeis de desenvolvimento de software serem vistas como uma contradição aos requisitos regulatórios, estudos de casos em organizações de desenvolvimento de software para dispositivo médico revelam que estão adotando com êxito práticas ágeis em seus projetos e obtendo autorização regulatória para comercialização [MCH13].

Análises realizadas com organizações de desenvolvimento de software mostram que as práticas ágeis aplicadas na ER podem compensar algumas das deficiências dos métodos tradicionais de ER [INA15, CAO08]. Algumas delas são:

1. Foco na interação com o cliente facilita a compreensão das necessidades por causa do acesso imediato aos clientes e seu envolvimento no projeto quando necessário.

2. Com o desevolvimento iterativo dos requisitos, eles podem ser revistos e mudanças podem ser acomodadas em uma próxima iteração. A mudança é intrínseca a métodos ágeis.
3. A Extrema priorização de requisitos é valorizada na engenharia de requisitos ágil por oferecer inúmeras oportunidades para redefinição de prioridades promovendo melhor compreensão das necessidades do cliente agregando valor de negócio, em contraste com o desenvolvimento tradicional, onde alcançar redefinição de prioridades é algo difícil por geralmente ocorrer uma única vez.
4. O gerenciamento de mudanças dos requisitos através de um planejamento constante é possível ser realizado devido a comunicação freqüente entre o desenvolvedor e o cliente durante o desenvolvimento, acomodando mudanças, o que evita em grande parte a necessidade de grandes mudanças após o desenvolviemnto. Dessa forma, o custo de implementar uma mudança diminui drasticamente se comparado com o desenvolvimento de software tradicional.
5. A validação contínua realizada através dos testes são fundamentais em metodologias ágeis como uma forma de alcançar a qualidade validando se o que está sendo entregue a cada iteração está de acordo com as especificações de testes.

A adoção de métodos ágeis pode afetar a forma como as atividades de ER tradicional são conduzidas, infundindo soluções aos seus desafios. No entanto, pode representar novas limitações a sua aplicabilidade, na qual a gestão de requisitos é um deles. Os autores de [INA15], relatam que se faz necessário mais investigações por parte da academia e indústria em busca de melhorar as atuais práticas da ER no desenvolvimento ágil.

2.3 Rastreabilidade de Requisitos

A rastreabilidade de requisitos é um componente de grande utilidade para o gerenciamento de requisitos, pois consiste em manter informações relacionadas ao gerenciamento de suas alterações garantindo a conformidade do software com os seus requisitos [LEI01].

O centro de excelência para rastreabilidade de software e sistemas (CoEST) define rastreabilidade como sendo a “capacidade de inter-relacionar qualquer artefato de software unicamente identificável com outro, mantendo a relação existente ao longo do tempo. Além

de, usar a rede de conexões resultantes para responder questões relacionadas tanto ao produto de software quanto a processo de desenvolvimento [WIN10]. A rastreabilidade entre artefatos ajuda a lidar com a manutenção, evolução e sustentabilidade do sistema de software e seus artefatos [GAY16].

A garantia da conformidade do software com os seus requisitos é uma exigência básica da indústria de desenvolvimento de software por ser um elemento fundamental de qualquer processo de desenvolvimento de software rigoroso [RAM97]. Ela fornece suporte para várias tarefas de engenharia de software, tais como validação de requisitos, análise de impacto, análise de cobertura, verificação de conformidade e análise de derivação [DAV93]. A importância da rastreabilidade de requisitos é refletida pelo fato de que é exigida por padrões de requisitos, normas de segurança e modelos de maturidade [REM15]. Além disso, é uma das práticas recomendadas pelo padrão IEEE 830-1998 que propõem orientações para especificação de requisitos de software, bem como a norma ISO 29148-2011 que fornece orientações na aplicação de processos de engenharia e gerenciamento de requisitos recomendando que os requisitos devam ser rastreáveis.

Para o desenvolvimento de sistemas críticos, como por exemplo, os voltados para a área de saúde (dispositivos médicos), o atributo segurança é totalmente comprometedor, a rastreabilidade é fortemente exigida como um dos argumentos que garante que o sistema é seguro para o uso [ZEL14].

Por exemplo, a norma IEC 62304, conforme anteriormente discutida para ciclo de vida do software do equipamento médico, exige a rastreabilidade entre os requisitos do sistema, requisitos de software, testes e medidas de controle de risco implementados em software [MAD13].

Além disso, o FDA salienta que a análise da rastreabilidade deve ser usada para verificar se um design de software implementa corretamente requisitos de software. A norma ISO 26262 de segurança para a indústria automotiva, também exige que os requisitos sejam rastreáveis para cada implementação do projeto do sistema.

Por várias razões empresas de desenvolvimento de software são constantemente orientadas a introduzir a rastreabilidade em seu processo de desenvolvimento, por garantir que o produto atende as necessidades do cliente fazendo o que é necessário; por apoiar o gerenciamento de mudanças; além de facilitar a manutenção do produto a longo prazo [PAN10].

De acordo com os autores de [GOT94] a rastreabilidade de requisitos se refere à capacidade de descrever e seguir a vida de um requisito, tanto na direção para frente como na direção para trás (ou seja, desde as suas origens, passando pelo seu desenvolvimento, especificação até a sua implantação e posterior utilização, passando por períodos de refinamento e iteração em qualquer destas fases). A rastreabilidade de requisitos é utilizada para capturar as relações entre requisitos, projeto e implementação de um sistema [RAM95].

Apesar das pesquisas neste campo perdurarem já há algum tempo ainda não há um consenso sobre como o regime de rastreabilidade define as informações essenciais para sua captura e utilidade, no entanto, o padrao IEEE 830-1984 afirma que a especificação de requisitos de software é rastreável se (i) a origem de cada um dos seus requisitos é clara e se (ii) facilita a referência de cada requisito no desenvolvimento futuro ou na documentação de apoio.

Para o autor de [PIN04] a informação sobre a rastreabiliadade de requisito está classificada em relação a direção, que pode ser para frente ou para trás (foward/backward). A direção para frente representa a capacidade de rastrear um artefato em direção a um componente de design ou de implementacao. Isto acontece quando queremos investigar o impacto na mudança de um requisito, como por exemplo, qual componente será alterado, qual teste precisa ser adicionado ou alterado. Enquanto que a rastreabilidade na direção para trás representa a capacidade de rastrear um artefato a sua origem, isso ocorre quando há uma mudança e queremos entender o motivo através da fonte que gerou o requisito seja uma pessoa, instituição, norma, etc [WIN10].

Outra maneira de identificar a informação de rastreabilidade é através da Rastreabilidade Pré e Pós requisitos. Para os autores de [AHM07] a rastreabilidade pré-requisito refere-se aos aspectos da vida do requisito antes da sua inclusão na especificação do requisito, ou seja, qual a razão para a existência do requisito, quais partes interessadas, ou normas que participaram da criação do requisito, como essas fontes afetam na criação do requisito. Já a rastreabilidade pós-requisito refere-se a capacidade de rastrear um requisito aos artefatos gerados no processo de desenvolviemnto, tais como, componentes e processos de verificação que foram afetados pelos requisitos.

A relevância que a rastreabilidade tem para as partes interessadas envolvidas difere devido as diferenças de seus objetivos e prioridades. Para o cliente, a rastreabilidade pode ser capaz de certificar que os requisitos do sistema estão sendo atendidos, já para o engenheiro de

software é fundamental o cuidado com a manutenção, como uma mudança em um requisito poderá afetar o sistema, quais módulos serão diretamente afetados [RAM95].

Apesar da rastreabilidade servir como apoio as atividades de análise de impacto, manutenção e evolução, verificação e validação, entre outras, a obtenção da rastreabilidade confiável ainda é um desafio no que diz respeito a sua prática, pois a qualidade dos dados precisa ser amadurecida [CLE14].

2.4 Consideraões Finais

Neste capítulo, foi apresentada a fundamentação teórica necessária para o entendimento deste trabalho. Foram descritos os principais conceitos sobre sistemas de informação em saúde e sua evolução, também foram descritos conceitos sobre engenharia de requisitos tradicional e ágil. Além disso, os conceitos sobre rastreabilidade de requisitos foram descritos bem como, a sua relevância no gerenciamento de requisitos. No próximo capítulo será apresentada a definição do processo proposto e suas diretrizes.

3 O PROCESSO RE4CH

Conforme mencionado na seção anterior, no contexto da engenharia de requisitos, um processo é um conjunto estruturado de atividades que são seguidas para derivar, validar e manter um documento de requisitos do sistema. Uma descrição completa do processo de ER deve contemplar as atividades que são realizadas, bem como a estruturação dessas atividades, quem é o responsável por cada atividade e as ferramentas utilizadas para apoiar a engenharia de requisitos [SOM98].

Este capítulo tem como objetivo apresentar um processo de ER no contexto de SIS descrevendo fundamentações, atividades e elementos.

3.1 Introdução

Propomos neste trabalho um processo de ER chamado de RE4CH (Requirements Engineering for Connected Health), que serve para lidar com os requisitos do sistema durante todo o ciclo de vida.

As atividades do processo RE4CH devem ser realizadas de forma iterativa e incremental, diferindo dos modelos sequenciais adotados por processos mais tradicionais, no qual as atividades são realizadas uma única vez em um plano sequencial. Neste sentido, é possível refinar requisitos em diferentes fases do ciclo de vida do desenvolvimento, permitindo uma maior flexibilidade para acolher novos requisitos e novas solicitações de mudança, além de possibilitar o gerenciamento mais efetivo dos requisitos.

Este processo foi definido com atividades, entradas, saídas, papéis e diretrizes. Sua modelagem foi baseada em BPMN (Business Process Modeling Notation) que é uma notação padrão desenvolvida pela Business Process Management Initiative e mantida pela Object Management Group (OMG) para modelar processos de negócio. BPMN além de oferecer uma notação padrão, tem o objetivo de ter um alto grau de abrangência por todos os usuários do negócio, que vão desde analistas de negócios, analistas de requisitos, desenvolvedores até os usuários finais do produto.

O processo RE4CH sugere o uso de ferramentas para modelagem UML. É ideal usar uma ferramenta que permita a modelagem de processos, gerenciamento de requisitos, casos

de uso, rastreabilidade e demais artefatos requeridos pelo o processo. A ideia é avançar para a padronização dos documentos gerados. A Enterprise Architect (EA) é uma ferramenta de modelagem UML proprietária da Sparx System e foi utilizada para dar suporte ao processo proposto e promover a padronização dos documentos.

O processo RE4CH consiste nas seguintes atividades: Elicitar Requisitos, Definir Objetivos, Definir Contexto, Definir Requisitos, Definir Casos de Uso, Validar e Verificar ERS e Gerenciar Rastreabilidade. As atividades do processo podem ser realizadas de forma seqüencial, enquanto outras atividades podem ser realizadas de forma paralela, quando apropriada. O processo inicia com a atividade de Elicitar Requisitos e segue com a atividade de Definir Objetivos. As atividades de Definir Contexto e Definir Requisitos podem ser realizadas de maneira paralela. Já a atividade de definir Casos de Uso deve ser realizada após as atividades de Definir Requisitos e Definir Contexto.

Cada atividade deverá ser executada por um papel e tem entradas e saídas. O papel principal para executar o processo RE4CH é o Analista de Requisitos, mas outros papéis importantes fazem parte do processo, como o analista do domínio, desenvolvedor e usuário final. O processo tem como saída principal da Especificação de Requisitos de Sistema (ERS) que é composta por artefatos oriundos das atividades. Uma visão geral do processo RE4CH é apresentada na Figura 1.

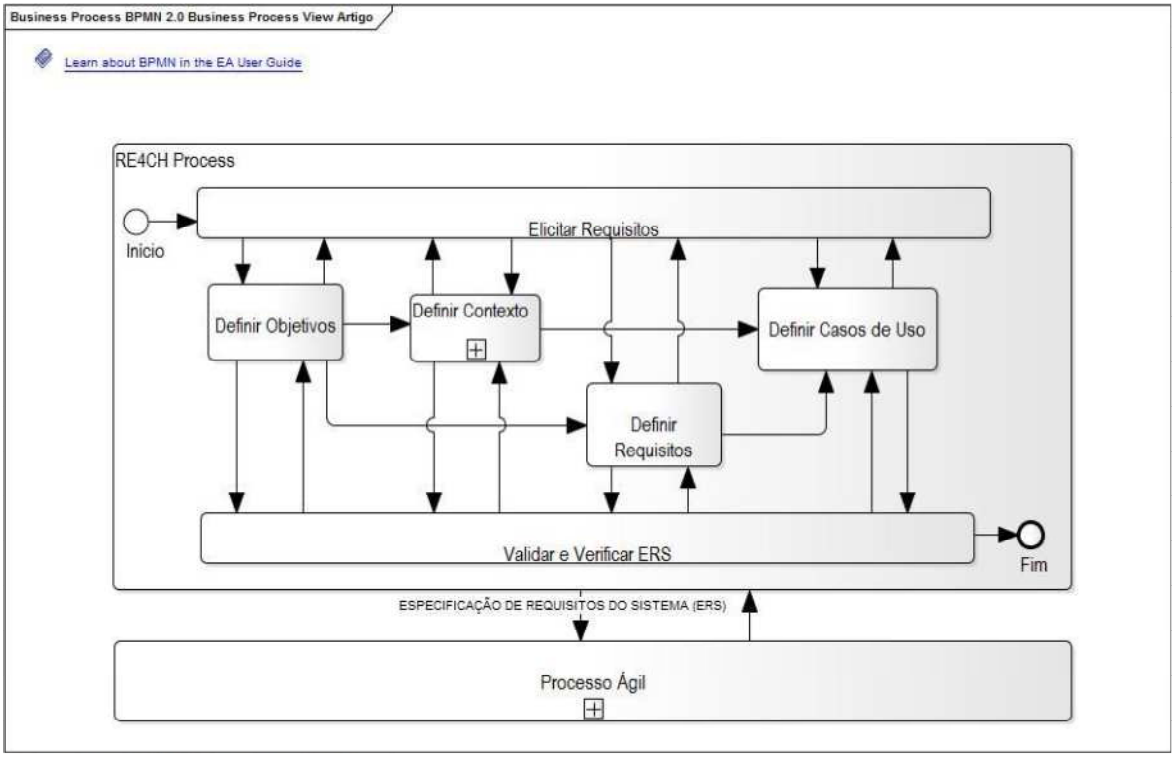

Figura 1 - Visão geral do processo RE4CH

Como pode ser visto na Figura 1, é possível observar a relação direta entre o processo RE4CH com as práticas de processo ágil, utilizado para gerenciar o projeto de desenvolvimento. As metodologias ágeis para o planejamento e gerenciamento de projetos tem uma grande aceitação e uso pela indústria de desenvolvimento de software. Isto se dá devido algumas vantagens apresentandas, tais como: permitir uma compreensão rápida das necessidades dos clientes, com foco na comunicação direta; aumentar a eficiência e flexibilidade no desenvolvimento de software, permitindo mudanças nos requisitos de uma forma rápida através de pequenas interações e constantes entregas [INA15].

Devido a estas razões, este tipo de metodologia foi considerada na concepção do RE4CH, que herdou algumas de suas práticas, como foco na interação com o cliente, priorização dos requisitos, desevolvimento iterativo dos requisitos, validação contínua, etc. No entanto, as metodologias ágeis estão mais focadas no processo de gestão e planejamento das atividades do que na definição de como um projeto será desenvolvido. Essa liberdade na execução do projeto pode levar equipes a minimizarem, tanto quanto possível a quantidade de documentação e o tempo necessário para a produção e manutenção. Nesse sentido, a documentação gerada pelo processo RE4CH tem como principal objetivo preencher esta lacuna que pode ser gerada por equipes ágeis durante o processo de desenvolvimento. O processo RE4CH lida com a produção de uma quantidade mínima de documentação necessária para obter a conformidade com os requisitos de qualidade no domínio da saúde.

As seções 3.2 a 3.8 descrevem as principais atividades que compõem o processo RE4CH, e discute a sua importância para o contexto que este trabalho está inserido. A seção 3.2 apresenta a atividade de Elicitar Requisitos. Nas seções 3.3, 3.4, 3.5 e 3.6 são definidas as atividades relacionadas à especificação dos requisitos do sistema, respectivamente: Definir Objetivos, Definir Contexto, Definir Requisitos e Definir Casos de Uso. A seção 3.7 descreve a atividade Validar e Verificar ERS. Finalmente, a seção 3.8 descreve a atividade Gerenciar Rastreabilidade que apoia a gerenciamento das mudanças nos requisitos.

3.2 Elicitar Requisitos

A elicitação de requisitos é atividade que envolve descobrir e coletar os requisitos do sistema, de forma a proporcionar uma compreensão mais precisa e completa do que é exigido do sistema. A atividade de elicitação de requisitos começa com entendimento do domínio da

aplicação a ser desenvolvida, a fim de compreender onde se insere o sistema, quais são as reais necessidades das partes interessadas, e buscar soluções para o problema [SOM98]. Os stakeholders representam as partes interessadas, ou seja, alguem que tem influência direta ou indireta sobre os requisitos do sistema, por exemplo, clientes, analistas de requisitos, desenvolvedores, usuários finais etc.

Uma correta elicitação de requisitos é essencial para o sucesso do projeto. Em se tratando de SIS, uma elicitação de requisitos realizada de maneira errada, quando implementada em um sistema, pode trazer sérios riscos para saúde humana. A aceitabilidade do sistema depende de quão bem as necessidades das partes interessadas forem atendidas.

Existem várias técnicas que podem ser utilizadas na elicitação de requisitos afim de auxiliar a sua definição, como por exemplo, entrevistas, pesquisas, brainstorming, prototipação, etnografia, etc. Em SIS a informação pode ser oriunda de diversas fontes, por exemplo, artefatos existentes (registros de saúde, sistemas existentes, manuais, literatura, normas regulamentares), como também, dos profissionais de saúde e gestores. Por isso, muitas técnicas podem ser combinadas e usadas de acordo com o cenário do contexto. No processo RE4CH, a entrevista e brainstorming são as técnicas mais indicadas para elicitação de requisitos.

A atividade de Elicitar é um processo iterativo, onde mudanças são constantes e novos requisitos podem surgir ao longo do projeto, quando é necessário obter novas informações. No RE4CH, a atividade Elicitar não é realizada apenas uma única vez no processo, mas em cada interação, tornando a especificação dos requisitos do sistema atualizada de acordo com as necessidades emergentes.

Após a elicitação dos requisitos uma visão geral do sistema é descrita de maneira sucinta, proporcionando uma compreensão inicial sobre as principais motivações do sistema. Em paralelo a esta atividade, o processo RE4CH recomenda que uma proposta de projeto base pode ser criada pelo processo de desesenvolvimento ágil utilizado, servindo como entrada para a definição das principais funcionalidades. A Tabela 1 apresenta um resumo sobre a atividade elicitar requisitos:

Tabela 1 - Atividade Elicitar Requisitos

Atividade:	Elicitar Requisitos
Descrição:	- Descrever uma visão geral do sistema em alto nível. Nesta descrição deverá conter uma compreensão inicial sobre as principais motivações e objetivos do sistema.
Entrada:	- Artefatos e sistemas existentes; - Informações coletadas de brainstorming e entrevistas; - Normas.
Saída:	Descrição do sistema
Papel:	Analista de Requisitos / Analista de negócio

3.3 Definir Objetivos

A atividade de Definir Objetivos constitui uma das atividades iniciais do RE4CH. Como mostrado na Figura 2, esta atividade tem como entrada a descrição do sistema e fornece na saída um conjunto de objetivos do sistema em alto nível.

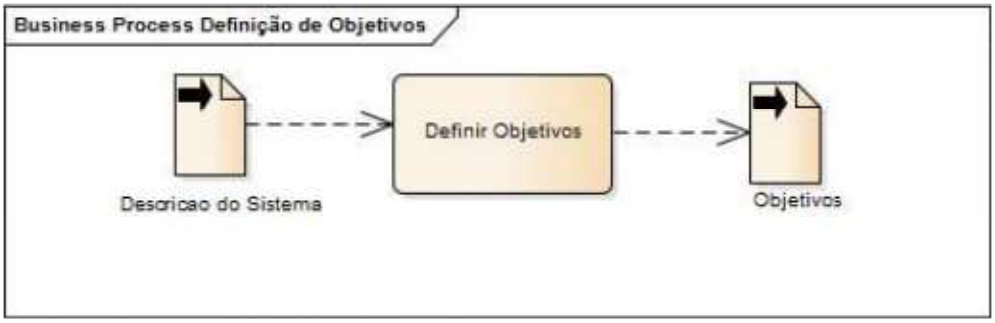

Figura 2 Visão Geral da Atividade Definir Objetivos

No RE4CH, os objetivos devem ser definidos de forma a assegurar um nível máximo de independência entre eles, permitindo a implementação de maneira paralela. Os objetivos também não devem limitar o escopo de soluções de design, mantendo um nível maior de abstração.

De acordo com [LEM09], os objetivos do sistema são refinados em requisitos verificáveis, ou seja, os requisitos podem ser validados por meio de atividades de teste. No

RE4CH, a rastreabilidade entre objetivos e requisitos deve ser mantida, uma vez que, os objetivos ajudam a explicar o porquê de um determinado requisito ser necessário.

Em processos ágeis, a definição de objetivos do sistema são cada vez mais usados no contexto de SIS, por possibilitar o paralelismo na implementação dos objetivos, proporcionando um ganho de eficácia. O RE4CH recomenda que eles sejam utilizados nos artefatos de processo de desenvolvimento ágil que definem as funcionalidades a serem implentadas, por exemplo no Product Backlog. A Tabela 2 apresenta um resumo sobre a atividade definir objetivos:

Tabela 2 Atividade Definir Objetivos

Atividade:	Definir Objetivos
Descrição:	- Definir um conjunto de objetivos constituindo uma especificação de alto nível do sistema. Os objetivos definidos devem estar diretamente relacionados a requisitos do sistema.
Entrada:	Descrição do sistema
Saída:	Objetivos do sistema
Papel:	Analista de requisitos / Analista de Negócio

No RE4CH, os objetivos devem ser registrados e organizados em uma ferramenta de modelagem UML e para cada objetivo são definidos os seguintes atributos apresentados na Tabela 3.

Tabela 3 Atributos Objetivos

Id*	Identifica o objetivo de forma única. Exemplo (GOAL001). (*) Deve ser obrigatório.
Nome*	O nome que melhor representar o objetivo. (*) Deve ser obrigatório.
Descrição	Descrição complementar do objetivo em alto nível.
Prioridade*	A prioridade do objetivo está relacionada com a importância para o negócio. Pode ser Alta, Média ou Baixa. (*) Deve ser obrigatório.
Autor	Identifica o indivíduo responsável por descrever um determinado objetivo.

3.4 Definir Contexto

A atividade definir contexto do sistema descreve em alto nível as entidades com que o sistema interage, bem como a natureza destas interações [LEM09]. Como apresentado na Figura 3, a atividade Definir Contexto no RE4CH terá como entrada os objetivos do sistema, fornecendo a necessária compreensão sobre o sistema, e como saída o diagrama de contexto contendo as entidades externas e subsistemas.

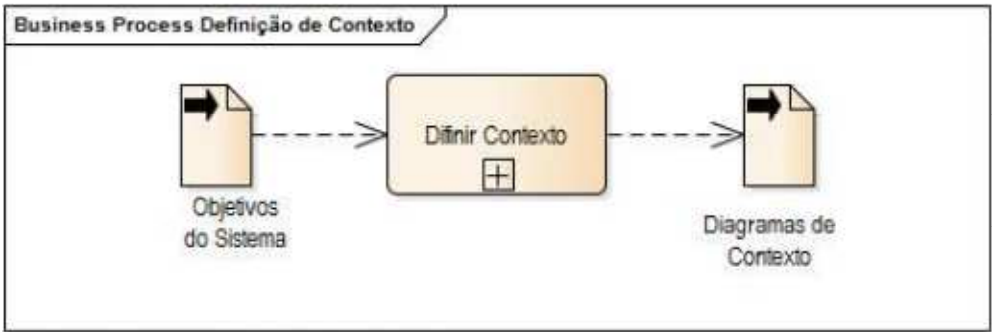

Figura 3Visão Geral da Atividade Definir Contexto

A atividade Definir Contexto no RE4CH é decomposta nas seguintes atividades: Definir Atores, Definir Elementos Contextuais, Definir Subsistemas e Definir Interações. Tais atividades são realizadas de forma sequencial, conforme pode ser visto na Figura 4.

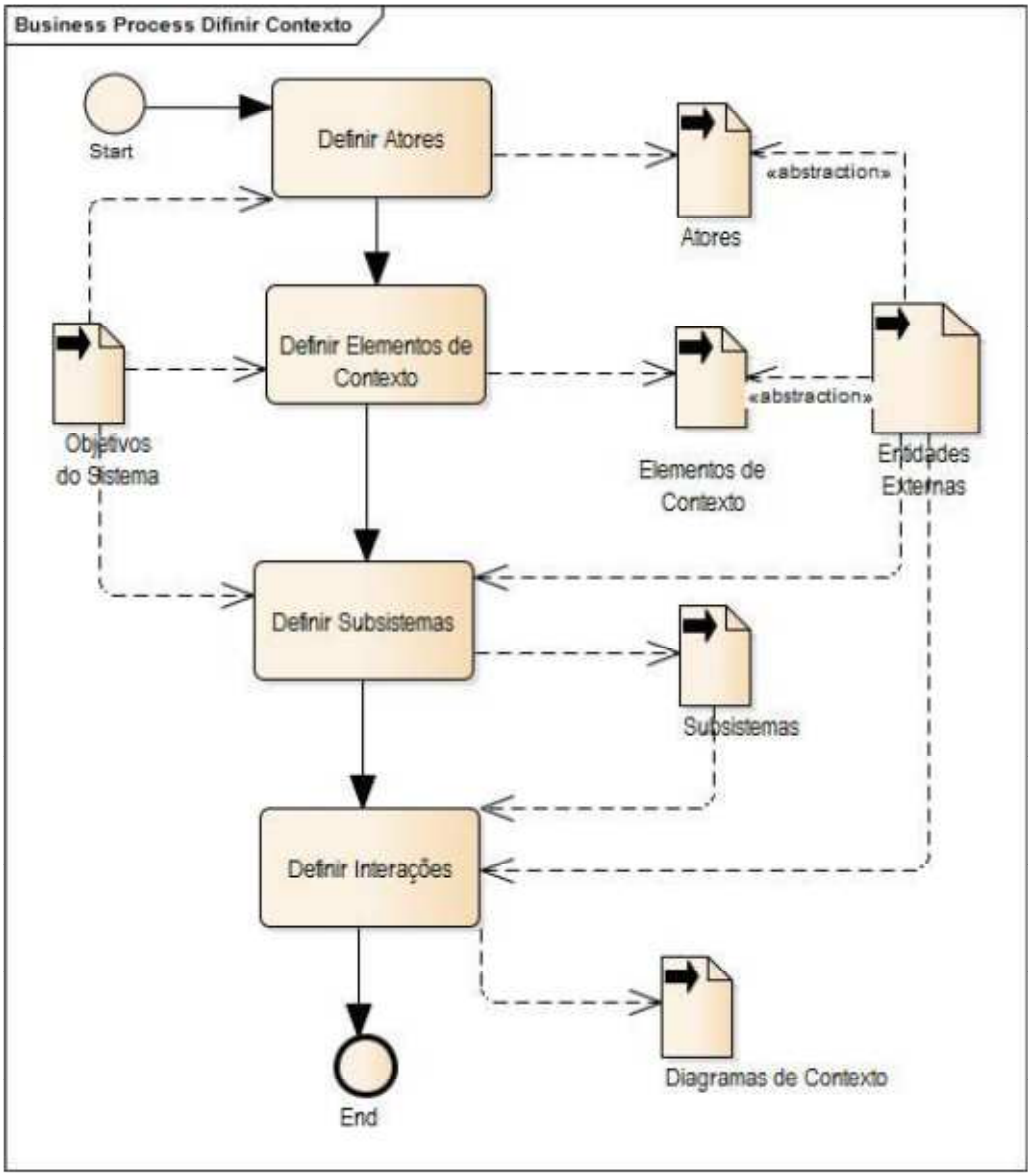

Figura 4 Atividade de Definir Contexto

Na atividade Definir Atores, os atores são definidos como entidades externas utilizadas para representar o papel humano que interage de maneira direta ou indiretamente com o sistema. Como apresentado na Figura 4, esta atividade tem como entrada os objetivos do sistema e como saída os atores que interagem com o sistema.

Os elementos contextuais que são definidos na atividade Definir Elementos Contextuais, também são usados para representar as entidades externas que interagem com o sistema. A diferença é que as entidades externas representadas nessa atividade, não desempenham o papel humano, sendo na maioria dos casos representadas por outros sistemas. Como pode ser visto na Figura 4, esta atividade tem também como entrada os objetivos do sistema, e gera os elementos contextuais que interagem com o sistema.

A atividade Definir Subsistemas consiste em modularizar o sistema em subsistemas. Esta divisão é possível, desde que, sejam conhecidos os objetivos do sistema. No RE4CH, essa atividade tem como entrada os objetivos do sistema e as entidades externas que interagem com o sistema. É possível realizar a decomposição do sistema em módulos denominados subsistemas.

A atividade Definir Interações define as interações entre entidades externas e subsistemas. Como resultado dessa atividade, tal como mostrado na Figura 4, temos o diagrama de contexto que pode proporcionar a comunicação entre clientes e desenvolvedores de forma eficiente, por permitir uma compreensão adequada do sistema.

Esta atividade constitui uma importante particularidade do processo RE4CH, representando um ponto chave na relação estabelecida entre o RE4CH e o processo de desenvolvimento ágil usado. Nos processos tradicionais de ER, o design interno desses subsistemas seria especificado antes de seu desenvolvimento ter iniciado. No entanto, no processo RE4CH os desenvolvedores têm autonomia para especificar o design interno dos subsistemas, reduzindo o tempo gasto com documentação e mantendo as características ágeis necessárias. Além de orientar o processo de desenvolvimento, a atividade de Definir Subsistemas apresenta um papel importante para a rastreabilidade do processo RE4CH, como será descrito na seção 3.8. Os requisitos do sistema deverão ser alocados aos diferentes subsistemas quando necessário, e esta rastreabilidade deve ser mantida ao longo do seu ciclo de vida. A Tabela 4 apresenta um resumo sobre a atividade de Definir Contexto:

Tabela 4 Atividade Definir Contexto

Atividade:	Definir Contexto
Descrição:	- Esta atividade deve ser realizada por meio da decomposição em outras atividades: Definir Atores; Definir Elementos Contextuais; Definir Subsistemas e Definir Iterações. As atividades devem ser realizadas de maneira sequencial.
Entrada:	Objetivos do sistema
Saída:	Diagrama de Contexto
Papel:	Analista de requisito

A atividade Definir Contexto bem como as atividades resultantes da sua decomposição, devem ser registradas e organizadas através da sua descrição numa ferramenta

de modelagem UML. O RE4CH estabelece os seguintes atributos para os objetos do contexto (Ator, Elemento de Contexto e Subsistema) que são apresentados na Tabela 5.

Tabela 5 Atributos do Contexto

Nome*	O nome que represente melhor o objeto (seja o ator, o elemento contextual ou subsistema). (*) Deve ser obrigatório.
Descrição	Descrição complementar do objeto em alto nível.
Autor	Identifica o indivíduo responsável por descrever um determinado objeto (ator, elemento contextual ou subsistema).

3.5 Definir Requisitos

A atividade Definir Requisitos tem o objetivo de especificar os requisitos do sistema. No RE4CH, as principais entradas são os objetivos do sistema, como está representado na Figura *5*. A identificação de requisitos deve ser realizada de acordo com os objetivos definidos. Dessa forma, além de se tornar mais fácil de entender a razão para existência de um requisito, a implementação destes poderão ocorrer de maneira paralela. Portanto, é importante observar a relação entre os objetivos e requisitos, que é uma relação de um-para-muitos, assim, um objetivo pode abranger muitos requisitos.

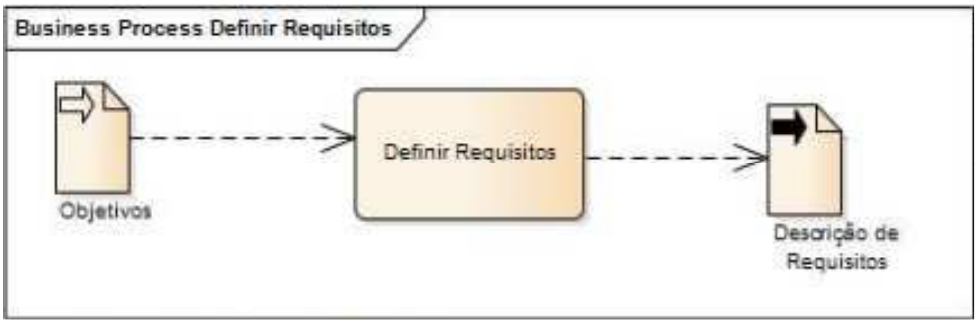

Figura 5 Visão Geral da Atividade Definir Requisitos

No RE4CH, a especificação dos requisitos é iniciada logo após sua identificação. Primeiramente, é realizada uma distribuição dos requisitos em categorias funcionais e não-funcionais bem como a classificação de cada um, de acordo com o grau de prioridade

definidos pelas partes interessadas. Os requisitos funcionais representam as funcionalidades do sistema, descrevendo ações que o sistema deverá realizar. Eles devem ser descritos em uma linguagem natural de maneira direta e objetiva para que seja de fácil compreensão. Os requisitos não funcionais são comportamentos e restrições do sistema relacionados à qualidade, como, por exemplo, segurança, confiabilidade, disponibilidade, etc. Recomenda-se que o requisito não funcional sempre que possível seja descrito de maneira quantificável, através de métricas, possibilitando maior precisão dos resultados.

A classificação de cada requisito é realizada de acordo com o grau de prioridade, ou seja, o grau de importância para o negócio, que deve ser analisado pelas partes interessadas. No RE4CH, o grau de prioridade dos requisitos deve ser classificado como Alto, Médio e Baixo.

- Alto: quando o requisito é extremamamente necessário. Sua implementação é obrigatória.
- Médio: quando o requisito não é obrigatório, porém é recomendado.
- Baixo: quando o requisito propõe algum valor adicional para o negócio, porém pode ser omitido sem afetar o uso do sistema.

Em seguida, os requisitos devem ser registrados e organizados através da sua descrição na ferramenta de modelagem utilizada. O RE4CH segue um modelo bem definido, para cada requisito estabelecendo atributos, como pode ser visto na Tabela 6.

Tabela 6 Atributos dos Requisitos

Id*	Identifica o requisito de forma única. Está padronizado de acordo com tipo de requisito e o número único de identificação respectivamente. Exemplo (RF01). (*) Deve ser obrigatório.
Tipo*	Pode ser funcional ou não funcional. (*) Deve ser obrigatório
Nome*	O nome que melhor representar o requisito. (*) Deve ser obrigatório
Prioridade	A prioridade do requisito pode ser alta, média ou baixa. (*) Deve ser obrigatório
Descrição	Descrição sobre os requisitos, enfatizando a razão de sua existência.
Autor	Identifica o indivíduo responsável por descrever um determinado requisito

Os requisitos devem ser geridos de modo a manter a rastreabilidade entre diferentes artefatos para que seja possível o controle da evolução do sistema. A rastreabilidade deve ser realizada de acordo com as diretrizes definidas na seção 3.8.

Os requisitos e os casos de uso são artefatos importantes que devem ser utilizados na definição de artefatos de processo de desenvolvimento ágil, que descrevem as funcionalidades a serem implementadas, bem como seus critérios de aceitação.

Em se tratando de desenvolvimento de sistemas para saúde, vale salientar a importância desta atividade no processo de desenvolvimento, pois um requisito levantado e implementado de maneira errônea pode causar sérios danos à vida de pessoas. A Tabela 7 apresenta um resumo sobre a atividade de definir requisitos:

Tabela 7 Atividade Definir Requisito

Atividade:	Definir Requisitos
Descrição:	- Identificar e descrever os requisitos do sistema de acordo com as diretrizes apresentadas na Tabela 3.6. Cada requisito deve ser associado com seu objetivo correspondente, como também deve ser associado com o subsistema que está relacionado, quando houver.
Entrada:	Objetivos do sistema
Saída:	Descrição dos Requisitos
Papel:	Analista de requisitos

3.6 Definir Caso de Uso

Os casos de uso descrevem o comportamento do sistema através de uma sequência de ações resultantes da interação entre sistema e ator. A atividade Definir Caso de Uso no RE4CH tem a finalidade de representar os requisitos funcionais em diagrama de casos de uso, como também descrevê-los.

O modelo de caso de uso é amplamente adotado na indústria e também comumente utilizado para especificar sistemas no domínio da saúde, por contribuir para uma melhor compreensão dos requisitos e funcionamento do sistema pelas partes interessadas, possibilitando comunicação eficiente com a equipe de desenvolvimento.

Tendo como entrada a descrição de requisitos e o diagrama de contexto, é possível identificar os casos de uso do sistema, como pode ser visto na Figura 6. No RE4CH, os casos de uso estão relacionados com os requisitos. Para esta identificação, é importante observar a relação entre os requisitos e casos de uso, como também a relação entre casos de usos e contexto. A relação entre os requisitos e casos de uso é uma associação de muitos-para-muitos, de uma forma que um único requisito pode englobar muitos casos de uso, ao passo que um caso de uso podem ser englobados por diversos requisitos. A execução de um caso de uso específico envolve a interação do sistema com os seus elementos contextuais.

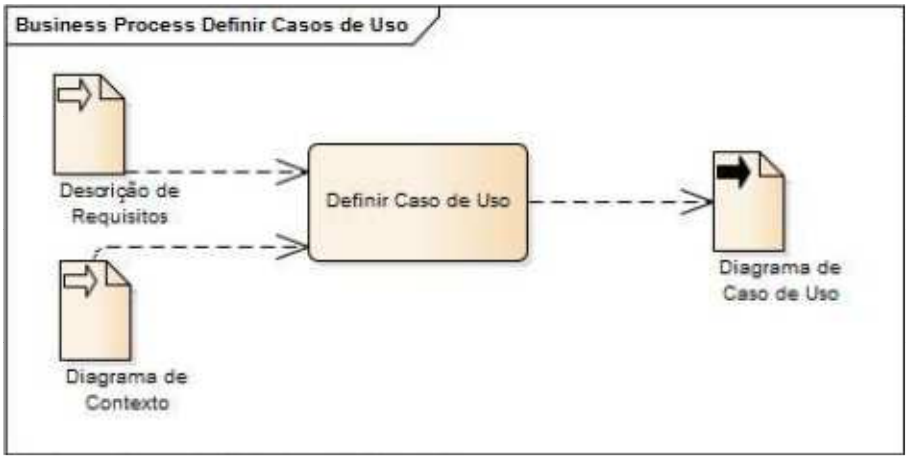

Figura 6 Visão Geral da Atividade Definir Caso de Uso

Após a identificação de casos de uso, a descrição é iniciada em uma ferramenta UML. O RE4CH define atributos para cada caso de uso. Os seguintes atributos são apresentados na Tabela 8. O nível de detalhe na descrição dos casos de uso dependerá da sua importância no contexto do sistema.

Tabela 8 Atributos Caso de Uso

Id*	Identifica o caso de uso de forma única. Exemplo (UC01). Onde o número identifica de maneira única o caso de uso. (*)Deve ser obrigatório.
Nome*	O nome dado ao caso de uso. (*)Deve ser obrigatório.
Descrição	Descreve de maneira sucinta o que o caso de uso representa. A descrição é opcional.
Ator(es)*	Define o(s) ator(es) que participam do caso de uso. (*)Deve ser obrigatório.
Elementos do Contexto*	Define os elementos do contexto que interage com o sistema. (*)Deve ser obrigatório.
Dependência	A dependência pode ser Include ou Extend, ou seja, se o caso de uso

	inclui ou se extende outro caso de uso. A dependência é opcional.
Pré-Condições*	A condição que deve ser verdadeira antes do caso de uso ser executado. (*)Deve ser obrigatório.
Fluxo Básico*	Representa o fluxo principal do sistema sem interrupções, ou seja, a sequência contínua de interações entre os atores e o sistema, sem intervenções. (*)Deve ser obrigatório.
Fluxo Alternativo	É quando representa desvios do fluxo básico, no sentido de descrever um caminho diferente do fluxo principal, sem erros. Eles devem ser identificados por um identificador, como, por exemplo, FA1.
Fluxo de Exceção	É quando causa erro ou interrupção no fluxo principal ou alternativo. Eles devem ser identificados por um identificador, como, por exemplo, FE1.
Pós-Condições*	Descrição do estado do sistema após execução do caso de uso. (*)Deve ser obrigatório.

Os casos de uso devem ser geridos de modo a manter a rastreabilidade entre diferentes artefatos de requisitos e de contexto para que seja possível o controle da evolução do sistema. A rastreabilidade é mantida de acordo com as diretrizes definidas na Seção 3.8. A Tabela 9 apresenta um resumo da atividade definir casos de uso:

Tabela 9 Atividade Definir Caso de Uso

Atividade:	Definir Casos de Uso
Descrição:	- Identificar e descrever os casos de uso a partir dos requisitos funcionais. A descrição dos casos de uso deve seguir as diretrizes apresentadas na Tabela 3.8. O nível de detalhe na descrição dos casos de uso depende da sua importância no contexto do sistema. Cada caso de uso deve ser associado ao seu requisito correspondente, como também ao subsistema que está relacionado.
Entrada:	Descrição dos Requisitos e Diagrama de Contexto
Saída:	Diagrama de Caso de Uso
Papel:	Analista de requisitos

3.7 Validar e Verificar ERS

A atividade Validar e Verificar ERS tem como objetivo certificar que as partes interessadas concordam e aprovam a ERS, verificando sua qualidade. Na validação, os requisitos são avaliados pelo cliente para assegurar a conformidade destes com suas necessidades. A verificação tem como objetivo checar se a ERS (objetivos, requisitos, contexto, casos de uso) apresenta problemas de inconsistência, incompletude, ambiguidade, falta de padronização e rastreabilidade.

No RE4CH, a qualidade dos artefatos que compõem a ERS, deve ser aferida através de uma análise crítica realizada por meio de inspeções. O processo de inspeção de software é uma maneira de revisar os artefatos de software, ajudando na detecção de defeitos durante as diferentes fases do desenvolvimento [KAL05]. Inspeções realizadas em artefatos gerados na fase de requisitos podem identificar defeitos precocemente, diminuindo o retrabalho e custos associados, garantindo uma maior qualidade dos artefatos.

No RE4CH, as atividades do processo de inspeção da ERS foram baseadas no processo de inspeção de software tradicional [FAG99]. São elas:

- Planejamento: etapa inicial do processo, no qual o moderador seleciona a equipe de inspeção, os artefatos a serem inspecionados e os artefatos que serão utilizados para conduzir a inspeção;
- Inspeção: os inspetores analisam os artefatos individualmente por meio de um checklist para identificação de defeitos e produzem um relatório com os defeitos encontrados;
- Reunião: é realizada uma reunião entre o moderador, inspetor e autor, caso haja necessidade de avaliar divergências encontradas em relação aos defeitos detectados;
- Correção: o autor corrige os defeitos encontrados;
- Acompanhamento: o moderador acompanha os artefatos corrigidos e decide se deve passar novamente por uma inspeção.

Na Figura 7 é apresentadao uma visão geral do processo de verificação da ERS.

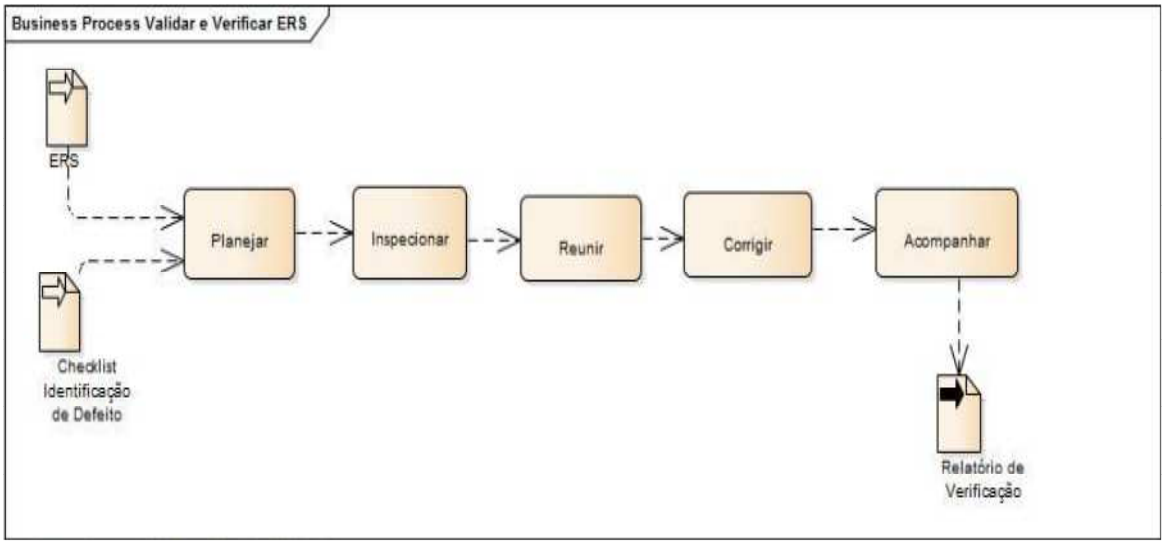

Figura 7 Visão Geral do Processo de Inspeção

Os papéis Moderador, Inspetor e Autor mencionados são responsáveis por realizarem as atividades de inspeção. O Moderador é responsável por gerenciar a equipe de inspeção, bem como orientar os inspetores. O Inspetor é quem realiza a inspeção dos artefatos, que pode ser realizada tanto por um analista de requisitos, como por um desenvolvedor. Desde que não seja realizada pela mesma pessoa que produziu o artefato, para que a inspeção não fique tendenciosa. O Autor é responsável por produzir os artefatos, no RE4CH, o analista de requisitos é o principal responsável.

A atividade Validar e Verificar Requisitos tem como saída o Relatório de Verificação que é gerado pelos inspetores. O relatório de verificação contem informações que identifica a inspeção, como por exemplo, o Inspetor responsável e data da inspeção. Nesse relatório as informações mais importantes estão relacionadas com os defeitos encontrados como, item, descrição, tipo de defeito, gravidade e causa. O relatório de verificação pode ser visto de forma mais detalhada no Apêndice B.

Os tipos de defeitos encontrados durante a inspeção podem ser classificados como: incompletude, inconsistência, ambiguidade, padronização e rastreabilidade.

Um defeito deve ser classificado do tipo incompletude quando, qualquer um dos artefatos que compõem a ERS (objetivos, requisitos, casos de uso ou elementos contextuais) estiver faltando. É também considerado como incompletude a falta de informações específicas de qualquer um dos artefatos que estão descritas em suas diretrizes, tornando a informação insuficiente.

Para um defeito ser classificado como inconsistente deve haver contradições nas descrições dos artefatos de qualquer um dos artefatos da ERS, como também haver conflitos entre eles. É considerado um defeito do tipo ambiguidade quando quaisquer um dos artefatos da ERS não estiverem descritos de maneira clara e objetiva, podendo ter várias interpretações possíveis para ele. Um defeito deve ser classificado do tipo falta de padronização quando qualquer um dos artefatos da ERS não estiver descrito conforme o padrão definido para ele. O tipo de defeito está relacionado com a rastreabilidade quando não for possível identificar os links de rastreabilidade entre os artefatos da ERS. Os links de rastreabilidade estabelecidos no processo RE4CH estão definidos na seção 3.8.

Um checklist foi criado como artefato para o processo de inspeção, com itens de verificação para auxiliar o inspetor, ajudando-o a identificar o tipo de defeito encontrado, como pode ser visto no Apêndice C.

A gravidade dos defeitos encontrados também é levada em consideração durante o processo de inspeção. Pode ser grave, média ou trivial. No RE4CH, um defeito deve ser considerado grave quando uma falha na ERS pode interromper o funcionamento do sistema. Os defeitos devem ser considerados com gravidade média quando falhas na ERS fizerem com que o sistema passe a operar de modo degradado. Defeitos considerados como triviais são caracterizados quando falhas na ERS não afetam o funcionamento do sistema. A gravidade dos defeitos foi baseada na norma IEEE 982.2-1988.

A causa do defeito encontrado pode estar relacionada com a falta de experiência, falta na documentação de entrada, falta de padrão de documentação, omissões na análise e falha na documentação de entrada. O relatório de defeitos será de grande importância na avaliação do processo RE4CH, porque por meio das informações nele encontrada, será possível analisar as causas dos defeitos encontrados, bem como a gravidade, ajudando a equipe a obter experiência com a ER.

A atividade Validar e Verificar não é realizada apenas uma única vez ao final do processo, como acontece na engenharia de requisitos tradicional, mas de maneira contínua, a medida que vai sendo entregue iterações. Dessa forma, é possível obter a especificação dos requisitos do sistema atualizada, com as mudanças que venham a surgir. A Tabela 10 apresenta um resumo sobre a atividade validar e verificar ERS:

Tabela 10 Atividade Validar e Verificar ERS

Atividade:	Validar e Verificar ERS
Descrição:	- Validar e Verificar se a ERS foi definida de forma (completa, consistente, sem ambiguidades, rastreável, padronizada) e que esteja em conformidade com as necessidades das partes interessadas.
Entrada:	- ERS (Objetivos, Descrição dos Requisitos, Diagrama de Contexto, Diagrama de Casos de Uso) - Artefatos da Inspeção (Chekclist para Identificação de Defeitos)
Saída:	- Relatório de Verificação
Papel:	- Representante das partes interessadas - Analista de Requisitos - Desenvolvedores

3.8 Gerenciar Rastreabilidade

Cada atividade proposta pelo processo RE4CH tem como saída artefatos arquiteturais gerados com a finalidade de alcançar uma documentação mais completa dos requisitos, do que a documentação geralmente exigida por processos de desenvolvimento ágeis. Outra finalidade do RE4CH é de alcançar a rastreabilidade entre os artefatos, tal como exigida por normas e certificações existentes no domínio da saúde. Neste sentido, a atividade de Gerenciar Rastreabilidade tem como objetivo manter a rastreabilidade entre os artefatos arquiteturais gerados das atividades de ER, como também manter a rastreabilidade com os artefatos gerados do processo de desenvolvimento ágil utilizado.

Na Figura 8 pode-se os observar três diferentes níveis de rastreabilidade entre artefatos definidos pelo processo RE4CH. De acordo com as características dos artefatos envolvidos, tem-se: (i) a rastreabilidade entre artefatos arquiteturais gerados das atividades de ER que compõem o RE4CH; (ii) a rastreabilidade entre artefatos do processo de desenvolvimento ágil utilizado, e (iii) a rastreabilidade entre artefatos pertencentes a ambos processos (RE4CH e processo de desenvolvimento ágil).

A rastreabilidade entre os artefatos pode ser observada na Figura 8. Nela são apresentados os artefatos arquiteturais que são resultantes das atividades do processo RE4CH (objetivos, requisitos, subsistemas e casos de uso) que estão envolvidos na atividade de

rastreabilidade. Conforme apresentado na seção 3.3, os objetivos devem ser decompostos em um conjunto de requisitos, e essa informação deve ser mantida através da rastreabilidade proposta. Quando houver relação entre requisito e subsistema essa informação deve ser rastreada, mantendo informações importantes quanto às decisões arquiteturais. Da mesma forma, quando houver relação entre casos de uso e subsistemas deve ser mantida a rastreabiliadade entre eles. Os casos de uso estão diretamente relacionados com os requisitos e a rastreabilidade entre eles deve ser mantida.

O RE4CH recomenda que seja realizada a rastreabilidade entre artefatos do processo de desenvolvimento ágil utilizado. Deve existir rastreabiliadade, pelo menos, entre os artefatos que representem as funcionalidades a serem implementadas, nos artefatos que descreva as características esperadas destas funcionalidades e nos artefatos que representem critérios de aceitação para estas funcionalidades. Na Figura 8 foram ilustrados artefatos do SCRUM, como é o caso do Epic, User Story e Test Case. Como pode ser visto, deve ser mantida a rastreabilidade entre Epic e User Story pois, cada Epic presente no Product Backlog é decomposto em um conjunto de User Stories, sendo mantida a relação entre estes artefatos. Já a rastreabilidade entre User Stories e Test Case deve ser realizada por manter informação relacionadas as atividades de teste realizadas no processo de validação das Sprints.

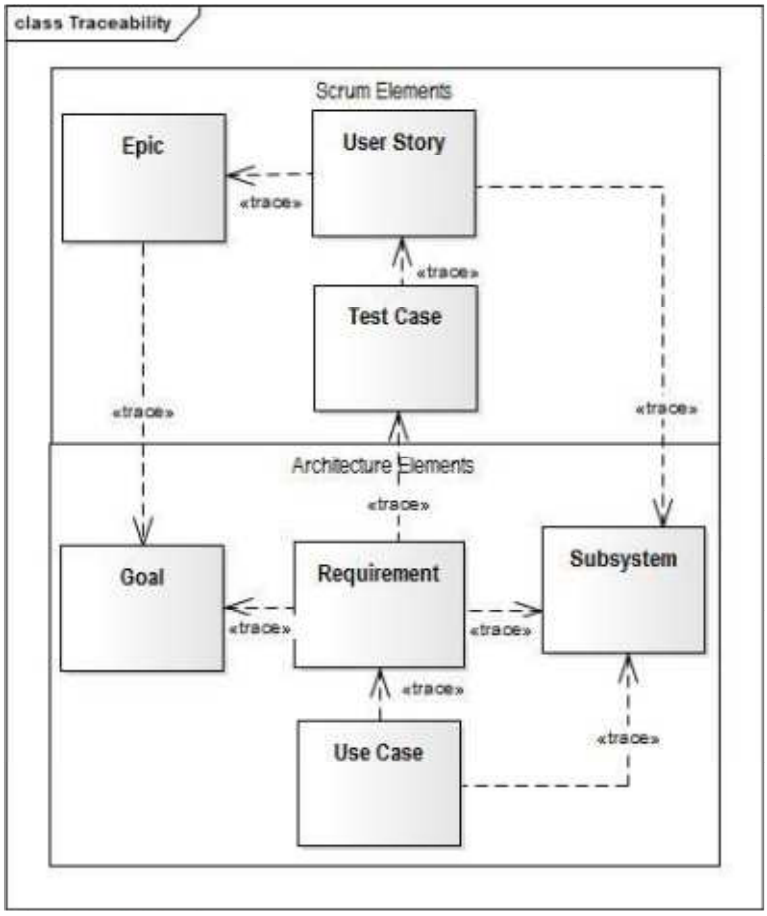

Figura 8 Modelo de Informação de Rastreabilidade

A rastreabilidade entre os artefatos provenientes tanto de atividades de ER que compõe o RE4CH quanto do processo de desenvolvimento ágil, constitui um elo entre a especificação do sistema e a sua implementação.

O RE4CH recomenda que deve ser mantida a rastreabilidade entre os objetivos do sistema com o artefato que contemple as funcionalidades a serem implementadas no processo ágil utilizado. Também deve ser mantida rastreabilidade entre os requisitos e o artefato que representa os critérios de aceitação das funcionalidades do processo ágil utilizado. Na Figura 8, foram ilustrados a rastreabilidade entre o processo RE4CH com o processo SCRUM. Os Epics são rastreados para os objetivos do sistema, para inspirar a criação de várias User Stories. Já as relações dos requisitos e seus respectivos casos de uso estão realcionados a User Stories por meio de seus Test Case, garantindo que o requisito será cumprido e validado. Finalmente, a relação entre a User Stories e Subsistema deve ser mantida para orientar a implementação [GAY16]. A Tabela 11 apresenta um resumo sobre a atividade gerenciar rastreabilidade:

Tabela 11 Atividade Gerenciar Rastreabilidade

Atividade:	Gerenciar Rastreabilidade
Descrição:	- Criar e manter links de rastreabilidade entre artefatos do processo de engenharia de requisitos (RE4CH); - Criar e manter links de rastreabilidade entre artefatos do processo ágil utilizado; - Criar e manter links de rastreabilidade entre artefatos do processo RE4CH e do processo ágil.
Entrada:	- Artefatos da ERS (Objetivos, Requisitos, Contexto, Caso de Uso); - Artefatos do processo ágil
Saída:	Rastreabilidade entre os artefatos do processo RE4CH, do processo ágil utilizado e de ambos os processos.
Papel:	Analista de requisitos

3.9 Considerações Finais

Este capítulo apresentou um processo proposto de engenharia de requisitos para o domínio da saúde, que é um domínio crítico. O processo RE4CH define um conjunto de atividades, entradas, saídas, papéis e diretrizes para tratar aspectos peculiares deste contexto. Os artefatos gerados por meio de suas atividades (Elicitar Requisitos, Definir Objetivos, Definir Contexto, Definir Requisitos, Definir Casos de Uso, Validar e Verificar ERS e Gerenciar Rastreabilidade) cobrem todo o ciclo de vida da ER fazendo uso de práticas ágeis. Próximo capítulo ilustra um exemplo de utilização da RE4CH no contexto SIS.

4 ESTUDO DE CASO

Neste capítulo, é apresentado um estudo de caso utilizando o processo RE4CH. Esse estudo de caso foi realizado no projeto HAM (Health Aggregator Manager), gerido pelo LIBE (Laboratório de Instrumentação Biomédica e Ensaios Eletrônicos) do NUTES (Núcleo de Tecnologia Estrategicas em Saúde), em parceria com a Signove Tecnologia S/A. O projeto HAM trata do desenvolvimento de uma aplicação no contexto de SIS, no qual lidará com a especificação e desenvolvimento de uma plataforma para gerenciar e integrar dispositivos médicos, seguindo a gestão dos requisitos do sistema durante todo o ciclo de vida. As atividades referentes à ER foram realizadas de acordo com o processo RE4CH. O detalhamento de cada uma das atividades será observado nas seções subsequentes.

4.1 O Projeto HAM

O projeto HAM constitui-se do desenvolvimento de um componente crítico de um sistema de saúde conectada. Este componente atua como um núcleo para processamento de mensagens entre dispositivos pessoais de saúde, dispositivos médicos, dispositivo de controle de infusão e registros de saúde eletrônico, que estejam em conformidade com os padrões de interoperabilidade adotados.

No projeto HAM, a norma ISO / IEEE 11073 foi utilizada como quadro de comunicação base entre o componente HAM e os dispositivos pessoais de saúde [[IEEE11073-20601]. Para implementação da ISO / IEEE 11073, foi utilizada a solução Antídote, uma biblioteca de código aberto bem consolidada fornecida pela Signove Tecnologia S / A, com uma vasta comunidade mundial. A principal razão para usar ISO / IEEE 11073 é a sua ampla adoção por fóruns industriais, tais como a Continua Health Alliance para a interoperabilidade de dispositivos [CAR07]. Como um facilitador ligado aos sistemas de saúde, a ISO / IEEE 11073 também é um candidato importante para a adoção dos serviços de saúde na Internet das coisas [SAN15]. Além disso, o HAM está sendo integrado a outros projetos de código aberto através de uma interface desenvolvida para se comunicar com o middleware Serviço de Distribuição de Dados (DDS). Este tipo de serviço permite o acesso a ambientes clínicos integrados da iniciativa Medical Device and Plug Play por meio do projeto OpenICE [PLO14].

O projeto HAM foi desenvolvido usando o processo RE4CH para atividades de ER e práticas ágeis do SCRUM para o planejamento e gerenciamento do projeto.

Como um processo de desenvolvimento ágil, SCRUM é baseado em interações contínuas, chamadas Sprints, no qual a equipe de desenvolvimento desenvolve itens definidos em um Product Backlog. O Product Backlog define todas as funcionalidades e os requisitos que o produto deve ter, e pode ser atualizado continuamente após a realização de cada Sprint. As funcionalidades e os requisitos definidos no Product Backlog são descritos claramente em User Stories, que podem ser agrupadas dentro de temas Épicos. Os Épicos representam os objetivos do sistema. Para cada Sprint, a equipe de desenvolvimento escolhe um conjunto de User Stories a ser desenvolvida durante essa iteração em uma reunião de planejamento. Para cada User Story deve ser definido o critério de aceitação, que tem como objetivo validar se a User Story foi implementada de acordo com o que o Product Owner havia requerido. Após o final da Sprint, todos os itens desenvolvidos são analisados pelo Product Owner, que é responsável pela definição e manutenção do Product Backlog, como também pela validação do produto. Após o processo de revisão, um novo planejamento é executado para a próxima iteração. Dessa maneira, o SCRUM foi aplicado ao gerenciamento do projeto HAM.

Nas seções seguintes será apresentado como se deu a aplicação do RE4CH dentro de um escopo definido que compreende quatro objetivos principais que compõem o desenvolvimento do HAM.

4.2 Atividade: Elicitar Requisitos

Para o levantamento de requisitos, as principais técnicas utilizadas para a compreensão de domínio foram entrevistas e brainstorming com especialistas da área de saúde conectada, como sugere o RE4CH. Como resultado, temos a descrição do sistema, apresentada na *Tabela 12*, constituída de uma visão geral contemplando as principais finalidades. Tais finalidades foram acordas pelas partes interessadas. Esta atividade foi executada pelo analista de requisitos do projeto HAM.

Tabela 12 Descrição do Sistema

Visão Geral do Sistema	O projeto HAM consiste do desenvolvimento de um Sistema de Saúde Conectada, que visa realizar a integração de dados entre dispositivos médicos, onde os dados devem ser coletados de dispositivos pessoais de saúde e compartilhados com Serviços de Saúde na nuvem. Para o alcançar este propósito é fundamental a compreensão de quatro subsistemas principais: Dispositivos Pessoais de Saúde, Dispositivos Clínicos, Serviços de Saúde e Hub de Dados de Saúde.
Dispositivo Pessoal de Saúde	Pequenos, dispositivos portáteis para a coleta de dados de saúde dos usuários. Eles fornecem funcionalidades de comunicação sem fio para permitir disponibilizar os dados coletados. Exemplos destes dispositivos incluem monitor de pressão arterial, oxímetro de pulso, termômetro e medidor de glicose, entre outros.
Dispositivo Clínico	Um dispositivo médico utilizado em ambientes clínicos, tais como monitores de sinais vitais.
Serviços de Saúde	Nuvem de servidores disponíveis 24/7 na Internet, responsáveis por armazenar e gerenciar os dados de saúde do usuário em um Registo de Saúde Pessoal.
Hub de Dados de Saúde	Pequenos, dispositivos portáteis sem fio para coletar dados a partir de dispositivos pessoais de saúde e sincronizar os dados com o Servidor nas nuvens Internet. O Hub de Dados de Saúde pode ser um smartphone, um tablet ou uma plataforma de propósito específico.

4.3 Atividade: Definir Objetivos

A partir da descrição do sistema, os objetivos foram definidos, de acordo com a Figura 9.

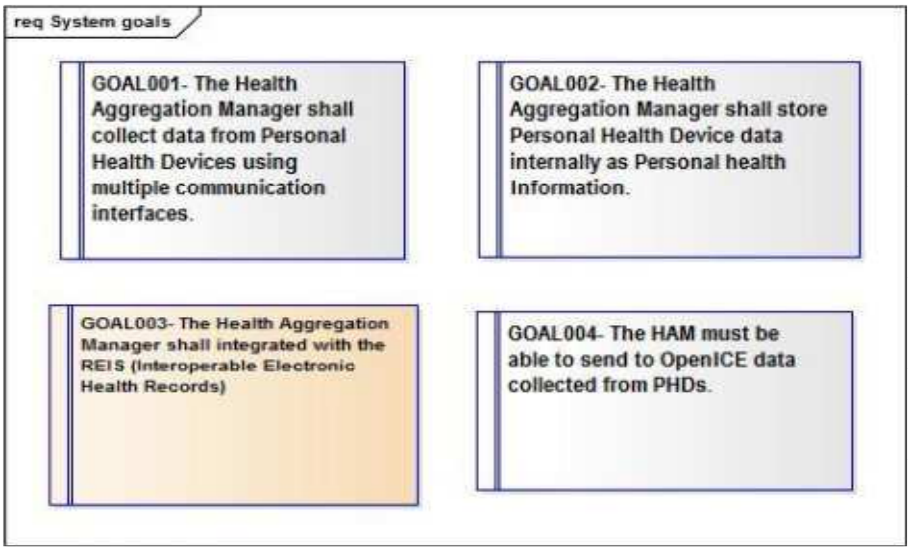

Figura 9 Objetivos do sistema

Os objetivos definidos abaixo representam o escopo definido para o estudo de caso.

- GOAL001- o Health Aggregator Manager (HAM) deverá coletar dados a partir de dispositivos pessoais de saúde com múltiplas interfaces de comunicação;
- GOAL002- o Health Aggregator Manager (HAM) deverá armazenar dados do dispositivo pessoal de saúde internamente como de informações médicas pessoais;
- GOAL003- o Health Aggregator Manager (HAM) deverá se integrar com o sistema de registro eletrônico de saúde REIS (Interoperable Electronic Health Records);
- GOAL004- o Health Aggregator Manager (HAM) deverá se integrar com o ambiente Clínica Integrada Open-Source (OpenICE).

Tal como recomendado pelo processo RE4CH, estes objetivos foram definidos, a fim de permitir o paralelismo nas suas implementações e manter um nível mais alto de abstração, não limitando o escopo das soluções de design.

A saída desta atividade foi mapeada para o processo SCRUM, no Produto Backlog, representando os Épicos.

4.4 Atividade: Definir Contexto

Com os objetivos do sistema especificados, foi definido o contexto do sistema. Na Figura 10 é apresentado um diagrama de contexto definido para o HAM, no qual temos como saída desta atividade, atores, elementos contextuais, subsistemas e interações. Esses

elementos representam informações mínimas necessárias para descrever em alto nível as entidades com quais o sistema interage, bem como natureza destas interações.

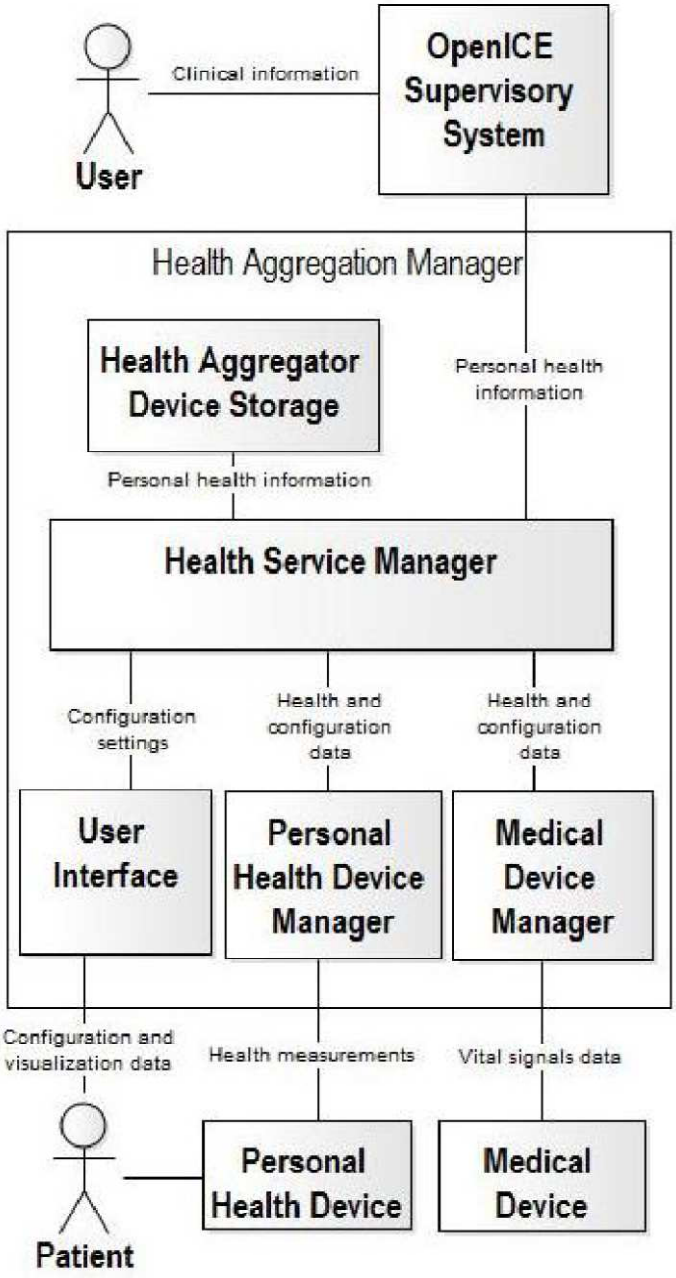

Figura 10 Diagrama de Contexto

O diagrama de contexto apresentado na figura acima representa em alto nível a interação entre as entidades e o sistema de saúde conectada. A interação é iniciada quando um usuário, como um paciente ou um clínico, ambos usuários do sistema (que representa os atores), desejam enviar/receber dados. Eles devem possuir um Personal Health Device (dispositivo de saúde pessoal) ou um Medical Device (dispositivo médico conectado),

representando os elementos contextuais, que são capazes de configurar e visualizar dados a partir de uma interface do usuário. As definições de configuração obtidos pela interface do usuário são processados pelo Health Service Manager (Gerenciador de Serviço de Saúde), que representa o subsistema, que por meio de gerenciadores Personal Health Device Manager e Medical Device Manager obtem medições de saúde do paciente e os sinais vitais de dados.

O Health Service Manager envia informações médicas pessoais a um Health Aggregator Device Storage, que deve estar na nuvem. Por fim, o Health Aggregator Device Storage integra as informações de saúde pessoal do paciente com o OpenICE (um sistema de supervisão de terceiros). Através desta integração deve ser possível enviar informação clínica a um usuário (médico, enfermeiro, etc.) que poderão ler registros eletrônico de saúde, caracterizando como um sistema de informação em saúde.

4.5 Atividade: Definir Requisito

A partir dos objetivos especificados também é possível iniciar a definição dos requisitos do sistema. A Figura 11 mostra alguns requisitos derivados do objetivo identificado como, GOAL001. Nesses requisitos, podemos observar o mínimo de informações obrigatórias que contemplam a execução desta atividade, como os atributos id, nome, prioridade e o tipo. Todos os requisitos selecionados são do o tipo funcional e apresentam alta prioridade, ou seja, de grande importância para as partes interessadas.

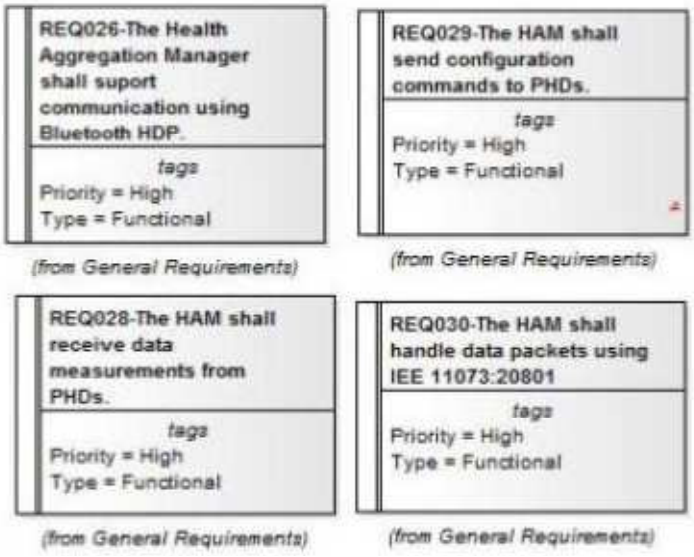

Figura 11Requisitos derivados da GOAL001

4.6 Atividade: Definir Caso de Uso

Com os requisitos e o contexto descritos, foram definidos os casos de uso. A Figura *12* apresenta um caso de uso associado a requisitos funcionais referentes ao objetivo GOAL001. Essa representação descreve a interação das entidades externas com as funcionalidades do sistema. Nesse diagrama de caso de uso podemos ver a interação entre usuário e as funcionalidades do sistema, como também a interação entre o elemento do contexto Personal Health Device e as funcionalidades.

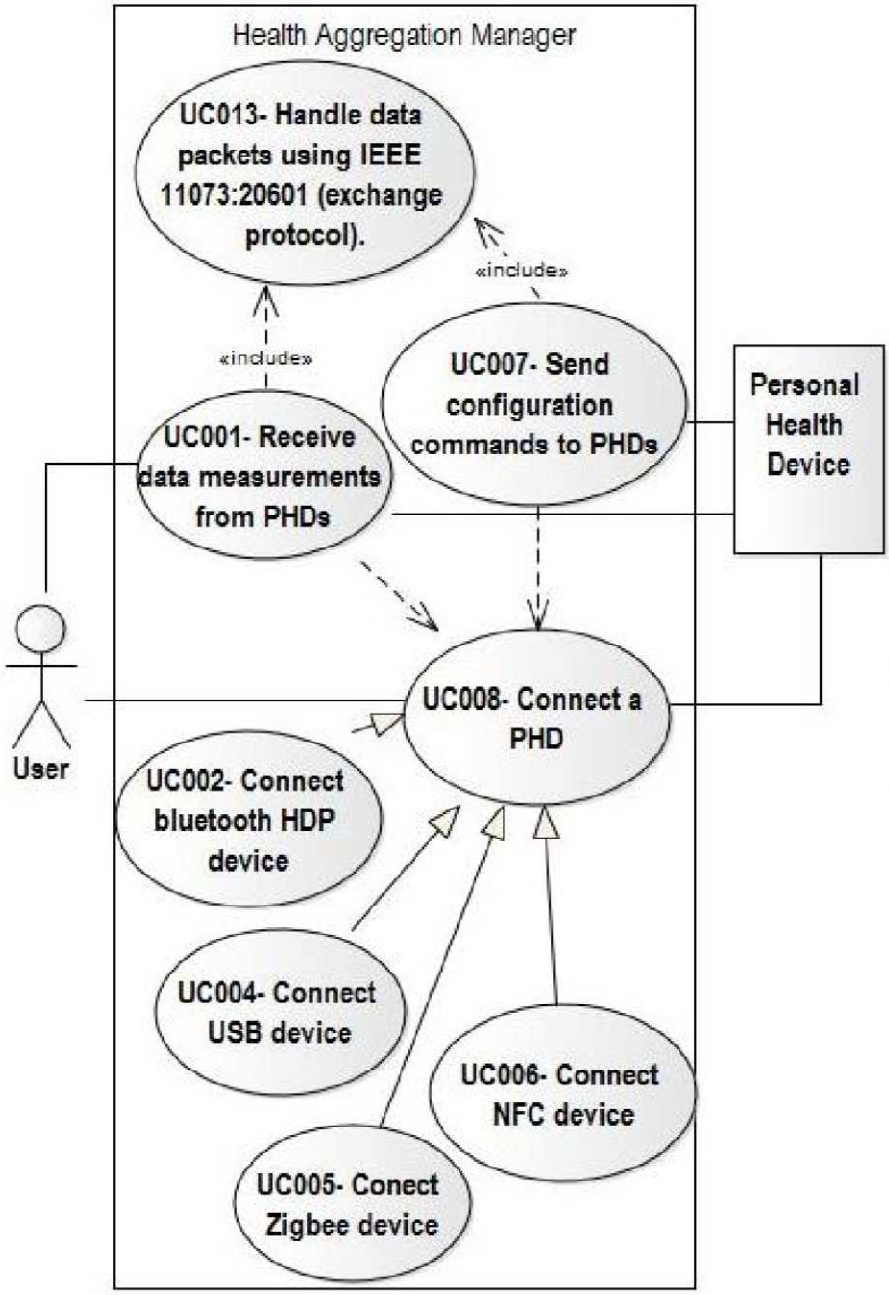

Figura 12 Caso de Uso associado a GOAL001

O diagrama de contexto, os requisitos e os respectivos casos de uso foram utilizados na definição de artefatos do SCRUM como é o caso da User Stories, que descrevem as fucnionalidades a serem implementadas, bem como dos seus critérios de aceitação, através dos Tests Case.

4.7 Atividade: Gerenciar Rastreabilidade

Com os artefatos arquiteturais resultantes das atividades de ER, como também dos artefatos gerados do processo de desenvolvimento SCRUM, foram criados os links de rastreabilidades entre eles. De acordo com o RE4CH, os artefatos arquiteturais envolvidos na atividade de rastreabilidade foram: objetivos, requisitos, subsistemas e casos de uso. Conforme o RE4CH, os artefatos gerados do processo de desenvolvimento ágil utilizado, deve manter rastreabilidade, neste caso, a rastreabilidade deve ser mantida entre os artefatos epic, user story e test case, gerados do SCRUM.

A Figura 13 abrange vários níveis de rastreabilidade definidos, envolvendo os artefatos arquiteturais gerados do processo RE4CH, como a rastreabilidade definida entre objetivo, requisito, subsistema e caso de uso. É possível que os objetivos (GOAL001) de cada requisito (RE027, REQ028 e REQ029) sejam rastreados, bem como cada requisito relacionado aos objetivos e com possíveis subsistemas. Além disso, é possível identificar os casos de usos relacionados a cada susbsistema, bem como os subsistemas de cada caso de uso.

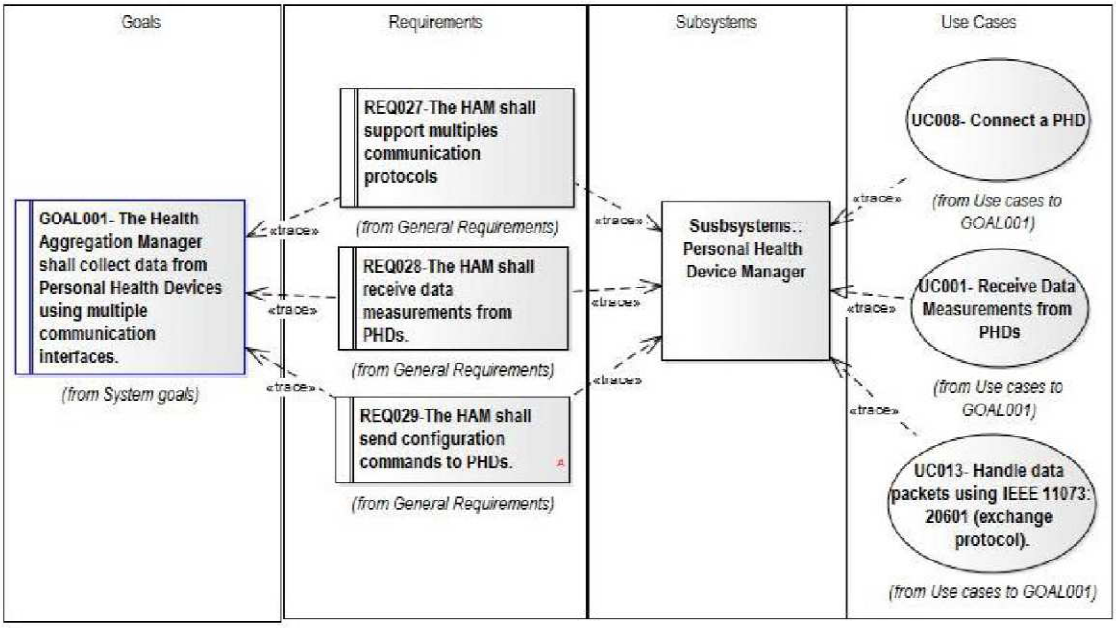

Figura 13 Trecho da rastreabilidade entre artefatos do processo RE4CH

Já na Figura 14 é a apresentada a rastreabilidade entre os artefatos arquiteturais e os artefatos do SCRUM.

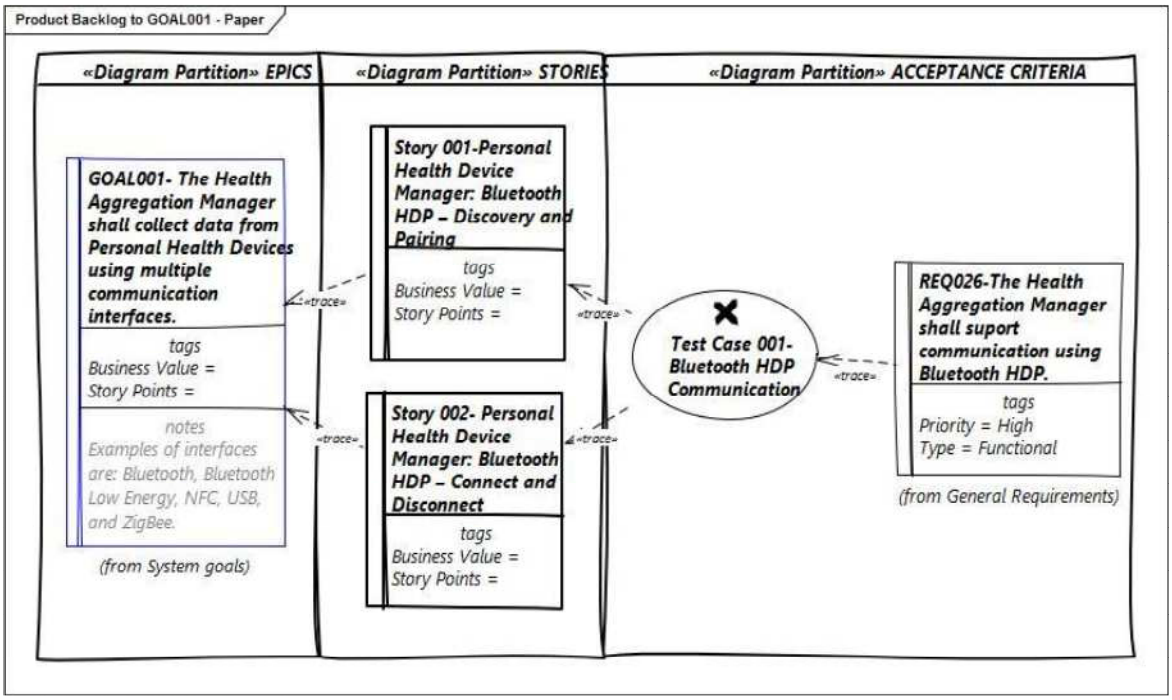

Figura 14 Trecho de rastreabilidade envolvendo artefatos arquiteturais e SCRUM

Podemos observar que, por meio da rastreabilidade, o objetivo (GOAL001) foi usado como épico (EPIC) pelo processo SCRUM, evitando informação redundante e também

originou as User Stories (Story 001 e Story 002). O caso de teste (Test Case 001) é utilizado como critério de aceitação para ambas as User Stories apresentadas, e o requisito (REQ026) é verificado por este caso de teste, estando relacionado com a User Story. A rastreabilidade entre as User Stories e Subsistemas também é encontrada de maneira indireta por meio da ligação que existe entre User Story e Requisito e pode ser vista na Tabela 13.

Tabela 13 Descrição de User Story

Id	Nome	Subsistema
Story 001	Bluetooth HDP - Discovery and Pairing	Personal Health Device
Story 002	Bluetooth HDP - Connect and Disconnect	Personal Health Device Manager

4.8 Atividade: Validar e Verificar ERS

Por fim, foi realizada a atividade de validação e verificação dos requisitos que tem como objetivo validar que a ERS gerada estivesse em conformidade com as necessidades das partes interessadas, bem como tenha sido definida de forma completa, consistente e sem ambiguidades. Esta atividade foi realizada tanto pelos analistas de requisitos e equipe de desenvolvimento, como também pelo representante das partes interessadas, nesse estudo de caso, pelo Product Owner, principal responsável por avaliar se as Sprints entregues satisfaziam os requisitos.

A revisão da ERS foi realizada pelos analistas de requisitos e desenvolvedores, por meio do processo de inspeção. Primeiramente, houve um planejamento realizado pelo moderador, no qual foi selecionado os participantes do processo de inspeção, bem como os artefatos a serem inspecionados e os que seriam utilizados como apoio durante a inspeção.

A inspeção foi realizada por inspetores selecionados, e estes fizeram a análise dos artefatos individualmente, produzindo um relatório com os defeitos encontrados. Em seguida, cada inspetor entregou o relatório de verificação para o analista de requisitos responsável pela especificação, para que este tomasse ciência dos problemas detectados, e apontasse o que levou a causa daqueles defeitos. Houveram algumas reuniões entre o moderador, inspetor e autor, para sanar qualquer tipo de divergência encontrada em relação aos defeitos detectados. Por fim, os autores corrigiram os defeitos encontrados e o moderador fazia um acompanhamento, decidindo se havia necessidade de passar por outra inspeção.

4.9 Considerações Finais

Neste capítulo, foi apresentado um estudo de caso utilizando o processo RE4CH e práticas ágeis do SCRUM, no qual foi ilustrado através do desenvolvimento de um sistema de saúde, o projeto HAM. Por meio das atividades do RE4CH, seguindo as suas diretrizes, foi possível obter a especificação dos requisitos do sistema dentro de um escopo definido bem como, realizar rastreabilidade entre os artefatos gerados, apoiando o gerenciamento das mudanças nos requisitos. Além disso, proporcionou uma relação direta com um processo ágil, mantendo rastreabilidade entre seus artefatos. No próximo capítulo será apresentada a avaliação deste estudo de caso com relação à eficácia da ERS gerada, por meio de métricas.

5 AVALIAÇÃO EXPERIMENTAL

Um estudo experimental será apresentado para avaliar a viabilidade do processo RE4CH por meio de um estudo de caso realizado num projeto de SIS, buscando avaliar a sua eficácia. O processo seguido para esta avaliação experimental está dividido nas seguintes atividades: definição, planejamento, operação, análise e interpretação [HOS00]. Estas atividades serão respectivamente apresentadas nas subseções seguintes.

5.1. Definição

A definição estabelece a base para que o experimento seja conduzido. Esta avaliação experimental foi elaborada com base na abordagem Goal Question Metric (GQM), que tem como objetivo dar suporte às organizações na institucionalização de processos de medições, especificamente na identificação de objetivos que serão traduzidos em medições quantitativas, por meio do uso de métricas [BAS94]. O GQM define orientações para objetivos, questões e métricas.

Os objetivos são estabelecidos para as diversas entidades de uma organização, sejam elas: projetos, processos, produtos, etc; por uma variedade de razões. As questões são utilizadas para caracterizar o alcance de um objetivo específico. As métricas definem um conjunto de medidas associadas a uma determinada questão afim de respondê-la de uma forma quantitativa. A medição existe em qualquer processo de construção de algo em que se deseja avaliar o que está sendo construído, pois, a partir dela, podemos ter a percepção da sua eficácia. As métricas usadas para processo e projeto de software através de medidas quantitativas permitem a equipe ter ideia da eficácia do processo de software que está sendo usado.

Foi definido o objetivo, as questões e as métricas utilizadas para avaliar a eficácia do processo proposto, objeto de estudo deste trabalho.

5.1.1 Objetivo

Analisar o processo de engenharia de requisitos RE4CH em relação à sua eficácia, do ponto de vista de estudantes e profissionais de TI, no contexto de desenvolvimento de SIS, realizada na universidade em parceria com a indústria.

5.1.2 Questão

Para o objetivo foram definidas questões quantitativas que serão respondidas por meio das métricas capazes de identificar se o objetivo foi atingido.

As questões foram descritas de acordo com o grau de eficácia do processo proposto ou seja, se o processo cumpre o que se espera dele. A eficácia está relacionada com a identificação dos tipos de defeitos recolhidos durante o período em que o experimento será executado, por meio da revisão da ERS, assim como, a capacidade do processo em gerar especificação de requisitos em conformidade com os seus padrões. São elas:

- Questão 1. Quais os tipos de defeitos encontrados durante a revisão da ERS?
- Questão 2. O processo gera especificação de requisitos do sistema de acordo com as normas definidas para representação do processo?

5.1.3 Métrica

Será abordada a utilização de métricas para os artefatos gerados pelo processo proposto. Métricas são utilizadas como indicadores da evolução do processo, como também, podem apoiar o gerenciamento do projeto. Em caso de identificação de desvios em relação ao que se espera do processo, devem ser tomadas medidas corretivas o quanto antes, a fim de resolver os problemas detectados; quanto mais cedo se faz essa correção, menor será o custo a ela associado [PRE11].

A eficácia é avaliada em relação à capacidade do processo em gerar requisitos em conformidade com os seus padrões. Dessa forma, serão adotadas as métricas quantitativas Tipo de Defeito, Distribuição do Erro e Índice Defeito [IEEE982.2-1988].

- **Métrica 1. Tipo de Defeito**: Essa métrica serve para identificar os tipos de defeitos encontrados por meio dos dados recolhidos na revisão.

  $D = (TD_1 : NTD_1; TD_2 : NTD_2 ;...; TD_i : NTD_i ;...; TD\square : NTD\square)$

  sendo D uma seqüência da quantidade de defeitos por tipo encontrado durante o período de revisão da ERS; TD_i é o tipo de defeito encontrado; NTD_i é o número de

defeitos encontrados daquele tipo, e k é o número de tipos de defeitos existentes. Os tipos de defeitos estão definidos como, incompletude, inconsistência, ambiguidade, padronização e rastreabilidade.

- **Métrica 2. Indíce de Defeito**: Essa métrica serve para medir a gravidade média dos defeitos encontrados. Essa é uma medida que fornece um índice contínuo e relativo do quão correto está o software à medida que prossegue por meio do ciclo de desenvolvimento. Essa medida pode ser aplicada tanto no início do ciclo de vida, na fase de requisitos, como quando o usuário tem produto que já pode ser avaliado [IEEE982.2-1988]. Essa métrica é baseada na IEEE 982.2-1988 e aplica as seguintes primitivas:

  d = número total de defeitos detectados durante a fase de ER.

  g = número de defeitos graves encontrados.

  m = número de defeitos médios encontrados.

  t = número de defeitos triviais encontrados.

  w_1 = fator de ponderação para defeitos grave (o valor é 10).

  w_2 = fator de ponderação para defeitos médio (o valor é 3).

  w_3 = fator de ponderação para defeitos triviais (o valor é 1).

  $ID = w_1 \frac{g}{d} + w_2 \frac{m}{d} + w_3 \frac{t}{d}$

De acordo com a norma a sua interpretação baseia-se na magnitude do Índice de Defeito (ID), que diz que quanto menor for o valor, melhor. Sendo assim, os limites de ID são $0 \leq ID \leq w_1$. O número de linha de base do Índice de Defeitos geralmente é calculado de projetos anteriores, do mesmo tipo de software produzido, ou seja, os dados recolhidos em projetos anteriores podem ser usados como um valor de referência para comparação. No entanto, neste trabalho a eficácia do processo foi analisada em relação ao fator de ponderação. Se o valor encontrado é mais próximo do fator de ponderação para defeitos graves (> 6,5) do que dos demais, então ele é considerado como um indicativo de que há necessidades do processo ser melhorado.

O uso do índice, incentiva a aplicação de processos de remoção precoce de defeitos, ou seja, quanto mais cedo os defeitos forem encontrados, melhor será o resultado. De acordo com a norma IEEE 982.2-1988, essa medida pode ser melhor aplicada quando combinada com outras medidas, pois fornece uma indicação do que está acontecendo, porém não o porquê daquele problema estar acontecendo. Utilizar medidas que analisam a causa de erro em conjunto com cálculos de índice de defeitos pode ser bastante esclarecedor.

- **Métrica 3. Distribuição de Erro:** Essa métrica é recomendada como um método importante para analisar as causas dos erros e áreas propensas a erro, pois ela permite a classificação dos modos de falha predominante [IEEE982.2-1988]. A busca das causas das falhas envolve a análise de erro dos dados de defeito recolhidos durante cada fase do desenvolvimento de software, que vai desde início do ciclo de vida do desenvolvimento. De acordo com a norma, os erros podem ser classificados e contados por fase, por causa, etc. Neste trabalho definiu-se como:

  DE = número de causas de erro por categoria, que pode ser por: falta de experiência, falta na documentação de entrada, falta de padrão de documentação, omissões na análise e falha na documentação de entrada. A análise de erro por distribuição é um fator chave para a compreensão de qual causa de erro está diretamente relacionada com o processo, como também identificar e analisar as dificuldades dos usuários em aplicar o processo. Por meio de intervenções ou ações corretivas é possível evitar que erros e falhas se propaguem novamente.

5.2. Planejamento

O planejamento é iniciado após a definição e é nele que será estabelecido a forma como o experimento será conduzido.

O ambiente no qual o experimento foi executado foi no Laboratório de Instrumentação Biomédica e Ensaios Eletrônicos (LIBE), do NUTES, no projeto HAM (Health Aggregator Manager) em paceria com a Signove Tecnologia S/A. O projeto HAM está sendo conduzido com o apoio do processo RE4CH, no qual as atividades de ER são realizadas de acordo com o processo - objeto de estudo deste trabalho. O contexto do experimento é caracterizado como estudo com base em problemas do mundo real. A variável dependente considerada neste estudo foi a eficácia do processo.

Os participantes do experimento foram seis pesquisadores, dos quais cinco eram estudantes de 8º e 9º período do curso de ciências da computação da Universidade Estadual da Paraíba e um estudante do curso de engenharia elétrica da Universidade Federal de Campina Grande do 9° período. Todos os pesquisadores faziam parte da equipe do projeto HAM do laboratório LIBE. Alguns eram bolsistas e outros voluntários, mas todos com o mesmo comprometimento no estudo. Os pesquisadores foram informados de que era necessário investigar o resultado da aplicação do processo. A maioria dos participantes tinha pouca

experiência em projetos industriais, com pouquíssima experiência com rastreabilidade. No entanto, em relação a ER, a maioria tinha tanto conhecimento teórico, como também algum know-how com o domínio de sistemas de saúde que já vinham participando nos projetos do LIBE. Os participantes têm papéis definidos no processo, de acordo com a sua experiência e interesse. Um participante pode executar mais do que um papel, bem como, o mesmo papel pode ser executado por mais de um participante.

A instrumentação elaborada se deu da seguinte forma. Aos participantes foram fornecidos material de apoio com a descrição completa do processo, como também, foi realizado treinamento com a finalidade de orientá-los no experimento. O treinamento começou com os conceitos relacionados com diagramas UML que seriam utilizados, bem como conceitos de rastreabilidade e práticas ágeis. Em seguida, foi discutido todo o processo RE4CH, o qual foram apresentados todos os detalhes das atividades, entradas, saídas e papéis do processo. Para coleta de dados, foram utilizados questionário, relatório e checklists como instrumentos de medição. O questionário (ver Apêndice A), foi entregue a cada participante, com o intuito de avaliar sua experiência, em projetos de desenvolvimento de software, atividades de ER, rastreabilidade e práticas ágeis. Os participantes receberam um relatório de verificação (ver Apêndice B) - para registrar os problemas encontrados - que iria auxiliar na análise dos dados relacionados com a eficácia. Além disso, um checklist (ver Apêndice C) também foi passado com a finalidade de ajudá-los a identificar o tipo de problema encontrado.

5.3. Operação

A fase de Operação consiste na aplicação do processo RE4CH. Na primeira etapa, a de execução, os participantes realizaram suas tarefas de acordo com o processo RE4CH e os dados foram coletados. Na segunda etapa, foi realizada a validação dos dados coletados. Antes do experimento ser executado, toda a instrumentação definida na subseção 5.2 foi fornecida. Além disso, os participantes fizeram uso de ferramentas de suporte a atividades de ER.

5.3.1 Execução

A avaliação experimental foi realizada durante o segundo semestre de 2016. O procedimento do experimento foi concedido inicialmente com um projeto piloto que foi

realizado com a mesma estrutura definida no planejamento. No entanto, o projeto piloto foi realizado com um único participante, que recebeu treinamento sobre como usar o processo proposto. O intuito deste projeto piloto era de fazer um estudo baseado na observação, com o objetivo de detectar problemas e o pesquisador responsável poder melhorar o material planejado antes da sua utilização. Este projeto piloto foi realizado durante os meses de agosto e setembro. O experimento com todos os participantes aconteceu durante os meses de novembro e dezembro.

Os papéis foram distribuídos entre os participantes de acordo com a sua experiência, bem como o interesse de cada um. Os papéis utilizados de acordo com o processo, foram: Gerente de Projeto, Analista de Requisitos, Desenvolvedor e Revisor. Em relação ao papel de ER, havia dois analistas de requisitos, mas toda a equipe trabalhou com as atividades de requisitos.

As atividades de ER foram realizadas conforme o escopo definido na seção 4. Foram identificados e especificados 4 objetivos, 88 requisitos, 2 diagramas de contexto. Em relação aos casos de uso, 26 foram identificados, e apenas 10 foram especificados de acordo com sua relevância. Foram contabilizados 96 links de rastreabilidade entre artefatos. Nem todos os requisitos foram implementados devido às limitações de tempo. No entanto, todos os artefatos gerados de acordo com o processo foram revisados. A medida que iam revisando a ERS, preenchiam o relatório de verificação e utilizavam o checklist de identificação de defeitos como auxílio na detecçao do tipo de problema que encontravam.

5.3.2 Validação dos Dados

Após a coleta dos dados, todos os questionários e relatórios foram revisados. Nenhum deles foi rejeitado devido a imperfeições ou erros, e todos participaram da avaliação.

5.4. Análise e Interpretação

A seguir são apresentadas a análise e interpretação dos dados decorrentes da realização do estudo de caso para avaliar a eficácia do processo proposto.

De modo a avaliar a eficácia do processo RE4CH, foram formuladas as seguintes hipóteses. A hipótese nula determina que a utilização do processo RE4CH em projeto de SIS não fornece benefícios suficentes para o seu uso, devido a uma pobre eficácia.

Ha1. μID > 6.5

Ha2. μErros devido à falta de experiência e omissões < 50%

A hipótese alternativa afirma que o uso do processo RE4CH fornece benefícios que justificam o seu uso, apresentando uma boa eficácia:

Hb1. μID ≤ 6.5

Hb2. μErros em relação à falta de experiência e omissões ≥50%

5.4.1 Tipo de Defeito

Com base na análise dos relatórios de verificação preenchidos pelos participantes, durante a revisão da ERS, a Tabela 14 apresenta os tipos de defeitos encontrados, bem como a quantidade de cada tipo.

Tabela 14 Tipos de Defeitos Encontrados na ERS

Questão 1. Quais os tipos de defeitos encontrados durante a revisão da ERS?	
Tipo de Defeito	**Quantidade de Defeito**
Incompletude	4
Inconsistência	25
Ambiguidade	2
Padronização	9
Rastreabilidade	12

5.4.2 Índice de Defeito

A Figura 15 apresenta o percentual da gravidade média dos defeitos encontrados na ERS. A gravidade média dos defeitos encontrados foi analisada de acordo com os limites de 0 <= ID <= 10. Foi identificado um índice de defeitos (ID) de 4,45, com base em 52 defeitos encontrados, nos quais 16 eram graves, 17 eram médios e 19 eram triviais.

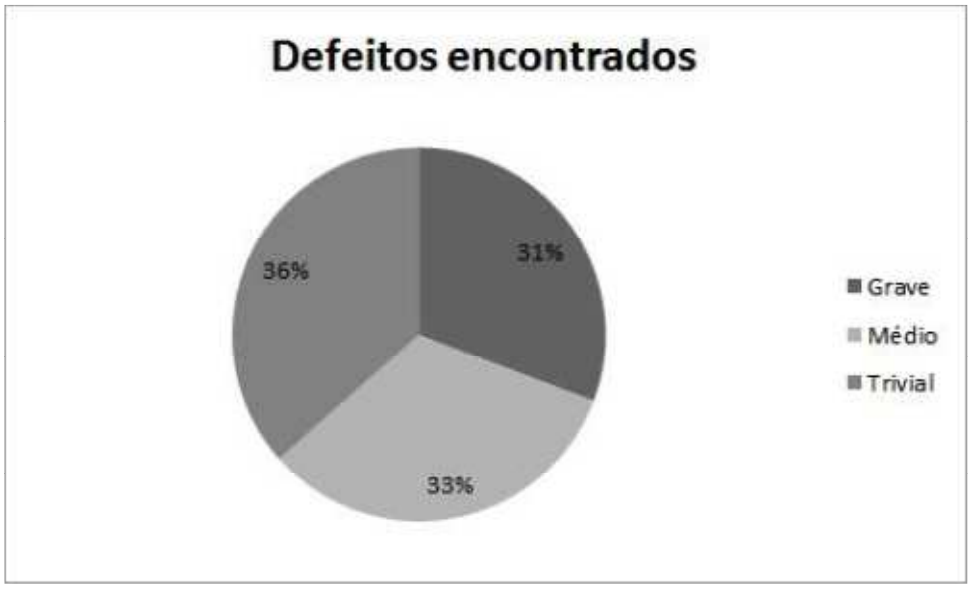

Figura 15 Gravidade média dos defeitos encontrados na ERS

O índice de defeito rejeita a hipótese nula: μID (~ 4,45) > 6.5. Ela indica que o ID encontrado representa uma gravidade moderada. Portanto, os resultados do índice de defeitos juntamente com a distribuição de erro (ver seção 5.4.3) validam a possibilidade do processo propiciar a geração de especificação de requisitos do sistema em conformidade com suas normas.

5.4.3 Distribuição de Erros

Com base na análise dos relatórios de verificação gerados pelos revisores, foram analisadas as causas de erros encontrados na ERS. Foi encontrado um total de 52 erros de acordo com a seguinte distribuição: 18 erros ocorreram devido à falta de experiência, 1 erro devido à falta na documentação de entrada, 4 erros devido à falta de padrão da documentação, 16 erros devido a omissões na análise, 13 erros devido a falhas na documentação de entrada. A densidade desta distribuição pode ser vista na Figura 16.

Figura 16 Distribuição de Erros

Estes resultados mostram que aproximadamente 65% dos erros estavam diretamente relacionados com a falta de experiência e omissão na análise. Assim, é rejeitada a hipótese nula: *μErros devido à falta de experiência e omissões (~65%) < 50%.* A concentração de erros relacionados à omissão na análise pode estar relacionada com o interesse em ER pelos participantes, apesar deles terem participado das atividades de ER, alguns deles poderiam não se encaixar no perfil para analista de requisitos. Um menor número de erros foi causado por problemas no processo. Portanto, os erros na ER podem ser reduzidos com o aumento da curva de aprendizagem ao longo do tempo.

Na interpretação dos resultados do experimento foram levadas em consideração algumas ameaças à validade. Primeiramente, é necessário observar que o estudo realizado é composto por seis estudantes, o que constituí uma amostra pequena. Além disso, por serem estudantes de graduação e terem um conhecimento semelhante por ambos estarem cursando entre 8° e 9° período, se torna impossível generalizar os resultados para um grupo de analistas de requisitos profissionais. No entanto, a maioria dos participantes possuía alguma experiência prática com desenvolvimento de software, de maneira que os resultados poderiam ser generalizados para desenvolvedores iniciantes.

5.5 Considerações Finais

Este capítulo apresentou a avaliação experimental do processo RE4CH com relação a sua eficácia no contexto de SIS. A análise quantitativa demonstrou que o processo tem uma eficácia significativa, a maioria dos erros encontrados estavam relacionados com a falta de experiência e omissão na análise, validando a possibilidade do processo gerar especificações de requisitos conforme suas normas. No entanto, o processo ainda apresentou erros com relação a falta de padrão na documentação e falha na documentação de entrada, necessitando de ajustes a fim de melhorar sua viabilidade. Mais experimentos devem ser realizados para que seja possível a generalizção dos resultados em relação aos benefícios que justificam a sua utilização.

6 TRABALHOS RELACIONADOS

Neste capítulo, são apresentados os trabalhos relacionados com esta pesquisa, que foram encontrados durante a revisão da literatura. Foram identificados vários estudos analisando a condução de atividades de ER em processos de desenvolvimento de software ágeis e como estes lidam com a rastreabilidade de requisitos, questão importante no contexto de desenvolvimento de sistemas críticos. Muitos desafios e limitações foram identificados, quando se tem a necessidade de desenvolver uma documentação e o gerenciamento de requisitos de maneira mais completa e rastreável, tal como exigido no desenvolvimento de produtos de software de domínios críticos.

6.1 Revisão Sistemática da Literatura sobre Engenaria de Requisitos em Projetos Ágeis:

6.1.1 Uma revisão sistemática da literatura sobre práticas e desafios de engenharia de requisitos ágeis [INA15]

Em [INA15], é realizada uma revisão sistemática da literatura sobre engenharia de requisitos ágil. Os autores expõem evidências encontradas sobre como práticas ágeis respondem aos desafios da engenharia de requisitos tradicional e como gerenciar equipes de desenvolvimento usando ágil.

Os resultados mostram que as práticas tradicionais e ágeis podem ser combinados a fim de obter melhores resultados. No entanto, os autores enfatizam a necessidade de mais estudos nesta área para responder a alguns desafios que ainda permanecem em aberto: gestão de mudança dos requisitos; escassez de modelos para guiar a aplicação da engenharia de requisitos na indústria; a falta de estudos sobre as práticas que estão sendo adotadas em domínios específicos, etc.

Além disso, a adoção de métodos ágeis pode afetar a forma como as atividades de engenharia de requisitos são conduzidas e resolver os desafios da engenharia de requisitos tradicional. Por exemplo, eles poderiam introduzir alterações de flexibilidade no escopo dos projetos em relação ao surgimento de novos requisitos. No entanto, pode apresentar novos desafios para sua aplicabilidade.

A avaliação cuidadosa sobre os riscos inerentes ao ambiente do projeto é recomendada, a fim de entender se os benefícios das práticas ágeis superam os custos inerentes aos desafios de engenharia de requisitos tradicional.

6.1.2 Engenharia de requisitos em projetos ágeis: um mapeamento sistemático baseado em evidências da indústria [AL15]

Em [ALV15], também foi realizada uma revisão sistemática com o objetivo de reunir conhecimento sobre como a ER estava sendo aplicada no desenvolvimento ágil de software. Nesse contexto, essa revisão sistemática identificou técnicas e processos de ER que estão sendo utilizados na indústria, quais os desafios e limitações encontradas e ainda sugeriu práticas que podem minimizar os efeitos de alguns desafios.

O estudo apontou que as técnicas mais utilizadas no levantamento de requisitos foram o encontro com cliente (meeting) e o brainstorming. Dentre as técnicas mais utilizadas para a especificação de requisitos estão user stories, protótipos, features, story card, use cases e etc. No entanto, as técnicas e processo encontrados pelo o autor diz respeito apenas as atividades de elicitar e especificar.

Em relação a análise dos desafios e limitações, o controle insuficiente de mudanças nos requisitos e a documentação insuficiente para implementação, manutenção ou treinamento se destacaram com várias ocorrências. O estudo apontou práticas de gerenciamento de mudanças e validações constantes. Essas práticas podem ser utilizadas para auxiliar na minimização dos efeitos causados por problemas gerados em relação ao controle ineficiente de mudança nos requisitos. Documentar requistos de forma mais detalhada, como critérios de aceitação, a complementação da documentação pertinente com comunicação formal, a utilização de equipes multifuncionais, entre outras, podem ajudar a reduzir os efeitos causados por problemas gerados com documentação insuficiente.

No entanto, os autores relatam que as empresas ainda apresentam diversos problemas relacionados principalmente com a gestão de requisitos quando se adota metodologias ágeis. E que várias questões ainda precisam ser investigadas pela academia na busca de melhorar as atuais práticas da ER, quando aplicadas a metodologias ágeis.

6.2 Técnicas e Abordagens para Atividades de Engenharia de Requisitos em Projetos Ágeis:

6.2.1 Uso de modelos i* para enriquecer requisitos em métodos ágeis [JAQ13]

Devido a falta de documentação no ambiente de desenvolvimento ágil ser apontada como um dos principais desafios da metodologia em [JAQ12], em [JAQ13] foi proposta uma técnica com o objetivo de ajudar a enriquecer a documentação de requisistos no desenvolvimento ágil, por meio de uma documentação visual através de modelos i*. A técnica consiste em mapear as histórias de usuário para os modelos i* fornecendo assim uma visualização abrangente das histórias de usuário, no que se refere a atores envolvidos, seus relacionamentos no contexto do sistema, bem como suas relações de dependências entre si e o sistema.

A técnica i* tem sido bastante explorada nas pesquisas na área de ER por dispor de capacidade semântica para representar relações de dependências dos atores no contexto organizacional, além de permitir mapear os requisitos funcionais e não funcionais de um software [SIL11]. Nesse contexto, esse trabalho apresentou uma forma de documentação de requisitos do sistema no ambiente ágil, propondo melhoria no entendimento do contexto do sistema, como também mais facilidade ao acesso a informação a partir de um modelo visual.

No entanto, a técnica proposta apesar de prover uma documentação mais completa para especificar requisitos do que a histórias de ususários, não abrange as demais atividades da ER, como a validação e gerenciamento dos requisitos.

6.2.2 Rastreabilidade leve para arquitetura ágil [GAY16]

A fim de atingir um equilíbrio entre a diminuição da sobrecarga de um processo e a manutenção de uma documentação mínima necessária, os autores de [GAY16] desenvolveram um método semi-automático que cria e mantém a rastreabilidade em uma arquitetura de software desenvolvida com processos ágeis. Este método foi aplicado em um estudo de caso industrial e os resultados mostram que é possível integrar informações de rastreabilidade em artefatos existentes com alguns esforços adicionais.

O método consiste em importar artefatos, tais como, user stories, casos de teste, requisitos, por meio da ferramenta TraceMan. Os artefatos podem ser criados em ferramentas como, Word, Excel, ferramentas de teste, etc., e estes são importados para a ferramenta de modelagem Enterprise Architect através do TraceMan. Por fim, os links de rastreabilidade são estabelecidos e mantidos em um modelo centralizado.

Dessa forma, é possível que os profissionais continuem criando os artefatos em ferramentas que já estão familiarizados, sem que seja necessário utilizar uma ferramenta específica para realizar rastreabilidade. Também é possível apoiar o gerenciamento dos requisistos através da análise de impacto durante a evolução do software. Os links de rastreabilidade podem ajudar a prever como as mudanças nos respectivos requisitos poderiam afetar a arquitetura. No entanto, as diretrizes da abordagem proposta estão relacionadas apenas com a rastreabilidade entre os artefatos, e não abrange outras atividades da ER, como especificação, validação de requisitos, etc.

6.3 DISCUSSÃO E CONSIDERAÇÕES FINAIS

Este capítulo apresentou algumas abordagens encontradas na revisão da literatura, para ER que se relacionam com o trabalho proposto. As abordagens identificadas de [JAQ13] e (GAY16) são as que mais se aproximam, focando em superar os desafios e limitações encontrados pela indústria em relação a condução das atividades de ER, quando lidam com processos de desenvolvimento de software ágil. Nesses trabalhos, técnica e abordagem foram propostas para conduzir especificação de requistos de maneira mais completa e apoiar a gestão de mudanças dos requisitos, através da rastreabilidade. No entanto, entre as abordagens incluídas nesta revisão, no geral, não foi encontrado um processo de ER que lide com todas as atividades de ER.

Assim, o RE4CH tem o objetivo de alcançar uma documentação e gerenciamento mais completo dos requisitos, lidando com a rastreabilidade dos artefatos gerados, e alinhando-se ao processo de desenvolvimento ágil. O RE4CH contempla as atividades de elicitação, análise, especificação, validação, e gerenciamento de requisitos, bem como diretrizes para cada atividade. No próximo capítulo serão apresentadas as conclusões deste trabalho.

7 CONCLUSÕES E TRABALHOS FUTUROS

A engenharia de requisitos tradicional tem sido desafiada nos últimos anos devido o ambiente de negócio da maioria das organizações necessitarem atender a mudanças constantes dos requisitos, rápida evolução das tecnologias, curtos prazos, etc. As metodologias ágeis foram criadas com intuito de minimizar esses problemas através de suas práticas e tem recebido grande atenção por parte da indútria e academia.

No entanto, as empresas que tradicionalmente desenvolvem sistemas de informação e fazem uso de metodologias ágeis, passam a enfrentar novos desafios quando lidam com o desenvolvimento de sistemas críticos. O desenvovlimento de software regulado na área de saúde é um exemplo, por requerer exigências que normalmente não são enfrentados pelos desenvolvedores de software tradicional. A norma IEC 62304, que lida com processos de software na área de saúde requisita que o fabricante documente e rastrei os requisitos do sistema. Embora as informações de rastreabilidade seja de grande importância para muitas atividades de engenharia de software, a maioria das abordagens ágeis de desenvolvimento dificilmente usam por causa do esforço extra necessário.

Tomando a norma IEC 62304 como inspiração, este trabalho apresentou um processo de engenharia de requisitos no contexto de SIS, lidando com a documentação e a rastreabilidade dos requsistos, ao mesmo tempo que tenta se alinhar com características ágeis. Para avaliar a viabilidade do processo proposto um estudo experimental foi realizado e apresentou benefícios na sua utilização.

Neste capítulo, é apresentada a conclusão deste trabalho, apresentando as principais contribuições, bem como trabalhos futuros que serão vistos nas próximas seções.

7.1 Principais Contribuições

Este trabalho contribuíu com a definição de um processo de engenharia de requisitos com atividades, entradas, saídas, papéis e diretrizes voltado ao contexto de SIS. Através das suas atividades, foi possível obter a especificação de requisitos mais completa e gerenciável do que a documentação normalmente exigida por metodologias ágeis.

A fim de garantir a rastreabilidade entre os artefatos, tal como exigida por normas e certificações existentes no domínio da saúde, o processo definiu diretrizes, apoiando o gerenciamento das mudanças nos requisitos. Além disso, proporcionou uma relação direta com um processo ágil, mantendo rastreabilidade entre os artefatos gerados.

O processo também incluíu na atividade de Validar e Verificar ERS o processo de inspeção, com o objetivo de checar se a ERS gerada apresentava problemas de inconsistência, incompletude, ambiguidade, falta de padronização e rastreabilidade.

Para avaliar o processo proposto foi realizado um estudo de caso em um projeto que trata do desenvolvimento de uma aplicação no contexto de SIS. Foram definidas métricas para avaliar a eficácia do processo proposto.

7.2 Trabalhos Futuros

Com a conclusão deste trabalho, fizemos uma análise do mesmo e vimos que há propostas para dar continuaidade com o objetivo de aperfeiçoar o que foi iniciado.

- Implementação da norma ISO/IEC 62304 em relação a atividade 5.2 que trata da análise de requisitos de software no processo de desenvolvimento para dispositivos médicos;
- Novos estudos experimentais são necessários em conjunto com a indústria, referente a outros sistemas de informação de saúde e sistemas físicos cibernéticos, para que seja possível observações empíricas que conduzam a resultados experimentais mais concretos;
- Acrescentar métricas para mensurar o processo proposto em relação a sua eficiência e usabilidade com satisfação do usuários com um número mais significativo de participantes.

8 REFERÊNCIAS BIBLIOGRÁFICAS

[COO08] COOPER, Alecia. Initiatives to Improve Patient Safety. Perioperative Nursing Clinics, v. 3, n. 4, p. 287-295, 2008.

[SAN10] SANTOS, Marcelo R.; BAX, Marcello Peixoto; PESSANHA, Christiano. Uma leitura ontológica da norma ISO 13606 para o Registro Eletrônico de Saúde. In: CONGRESSO BRASILEIRO DE INFORMÁTICA NA SAÚDE. 2010.

[ZEM15] ZEMA, M. et al. Developing medical device software in compliance with regulations. In: Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE. IEEE, 2015. p. 1331-1334.

[DEN07] DENGER, Christian et al. A snapshot of the state of practice in software development for medical devices. In: Empirical Software Engineering and Measurement, 2007. ESEM 2007. First International Symposium on. IEEE, 2007. p. 485-487.

[AXE11] AXELROD, C. Warren. Applying lessons from safety-critical systems to security-critical software. In: Systems, Applications and Technology Conference (LISAT), 2011 IEEE Long Island. IEEE, 2011. p. 1-6.

[HAQ14] HAQUE, Shah Ahsanul; AZIZ, Syed Mahfuzul; RAHMAN, Mustafizur. Review of cyber-physical system in healthcare. International Journal of Distributed Sensor Networks, v. 2014, 2014.

[MCC05] MCCAFFERY, Fergal et al. A software process improvement lifecycle framework for the medical device industry. 2005.

[MCH13] MCHUGH, Martin et al. An agile v-model for medical device software development to overcome the challenges with plan-driven software development lifecycles. In: Software Engineering in Health Care (SEHC), 2013 5th International Workshop on. IEEE, 2013. p. 12-19.

[SER15] SERRADOR, Pedro; PINTO, Jeffrey K. Does Agile work?—A quantitative analysis of agile project success. International Journal of Project Management, v. 33, n. 5, p. 1040-1051, 2015.

[INA15] INAYAT, Irum et al. A systematic literature review on agile requirements engineering practices and challenges. Computers in human behavior, v. 51, p. 915-929, 2015.

[IEC62304-2006] INTERNATIONAL ELECTROTECHNICAL COMMISSION et al. Medical device software: software life cycle processes. IEC, 2006.

[REM15] REMPEL, Patrick; MADER, Patrick. A quality model for the systematic assessment of requirements traceability. In: Requirements Engineering Conference (RE), 2015 IEEE 23rd International. IEEE, 2015. p. 176-185.

[BJA11] BJARNASON, Elizabeth; WNUK, Krzysztof; REGNELL, Björn. A case study on benefits and side-effects of agile practices in large-scale requirements engineering. In: Proceedings of the 1st Workshop on Agile Requirements Engineering. ACM, 2011. p. 3.

[CAS09] CASSIANI, Silvia Helena de Bortoli; GIMENES, Fernanda Raphael Escobar; MONZANI, Aline Aparecida Silva. O uso da tecnologia para a segurança do paciente. Rev. eletrônica enferm, v. 11, n. 2, 2009.

[FAT10] DE FÁTIMA MARIN, Heimar. Sistemas de informação em saúde: considerações gerais. Journal of Health Informatics, v. 2, n. 1, 2010.

[WAG09] WAGER, Karen A.; LEE, Frances Wickham; GLASER, John P. Health care information systems: a practical approach for health care management. John Wiley & Sons, 2009. 88 p.

[CAR08] CARTER, Jerome H. Electronic health records: a guide for clinicians and administrators. ACP Press, 2008.

[JOR11] JORDANOVA, M.; LIEVENS, F. Global Telemedicine and eHealth (A synopsis). In: E-Health and Bioengineering Conference (EHB), 2011. IEEE, 2011. p. 1-6.

[BOR14] BORIC-LUBECKE, Olga et al. E-healthcare: Remote monitoring, privacy, and security. In: 2014 IEEE MTT-S International Microwave Symposium (IMS2014). IEEE, 2014. p. 1-3.

[BUI11] BUI, Nicola; ZORZI, Michele. Health care applications: a solution based on the internet of things. In: Proceedings of the 4th International Symposium on Applied Sciences in Biomedical and Communication Technologies. ACM, 2011. p. 131.

[ROJ14] ROJAS-MENDIZABAL, Veronica et al. E-saúde e complexidade: uma proposta para o desenho de políticas públicas. Jornal Brasileiro de TeleSSaúde, v. 3, n. 2, p. 33-44, 2014.

[ROJ13] ROJAS-MENDIZABAL, Veronica A. et al. Toward a Model for Quality of Experience and Quality of Service in e-health Ecosystems. Procedia Technology, v. 9, p. 968-974, 2013.

[SOM06] SOMMERVILLE, Ian. Software Engineering. Addison-Wesley, 8th edition, 2006.

[PRE05] PRESSMAN, Roger S. Software engineering: a practitioner's approach. Palgrave Macmillan, 2005.

[SOM05] SOMMERVILLE, Ian. Integrated requirements engineering: A tutorial. Software, IEEE, v. 22, n. 1, p. 16-23, 2005.

[HAS15] HASSAN, Shoaib; QAMAR, Usman; IDRIS, Muhammad Arslan. Purification of requirement engineering model for rapid application development. In: Software Engineering and Service Science (ICSESS), 2015 6th IEEE International Conference on. IEEE, 2015. p. 357-362.

[SOM98] SOMMERVILLE, Ian; KOTONYA, Gerald. Requirements engineering: processes and techniques. John Wiley & Sons, Inc., 1998.

[NUS00] NUSEIBEH, Bashar; EASTERBROOK, Steve. Requirements engineering: a roadmap. In: Proceedings of the Conference on the Future of Software Engineering. ACM, 2000. p. 35-46.

[SOM97] SOMMERVILLE, Ian; SAWYER, Pete. Requirements engineering: a good practice guide. John Wiley & Sons, Inc., 1997.

[GOT95] GOTEL, Orlena Cara Zena. Contribution structures for requirements traceability. 1995. Tese de Doutorado. Imperial College.

[ESP04] DE ESPINDOLA, Rodrigo Santos; MAJDENBAUM, Azriel; AUDY, Jorge Luis Nicolas. Uma Análise Crítica dos Desafios para Engenharia de Requisitos em Manutenção de Software. In: WER. 2004. p. 226-238.

[PAN10] PANDEY, Dhirendra; SUMAN, Ugrasen; RAMANI, A. K. An effective requirement engineering process model for software development and requirements management. In: Advances in Recent Technologies in Communication and Computing (ARTCom), 2010 International Conference on. IEEE, 2010. p. 287-291.

[GOT94] GOTEL, Orlena CZ; FINKELSTEIN, Anthony CW. An analysis of the requirements traceability problem. In: Requirements Engineering, 1994., Proceedings of the First International Conference on. IEEE, 1994. p. 94-101.

[RAM09] RASMUSSEN, Rod et al. Adopting agile in an FDA regulated environment. In: Agile Conference, 2009. AGILE'09. IEEE, 2009. p. 151-155.

[BOE00] BOEHM, Barry. Requirements that handle IKIWISI, COTS, and rapid change. IEEE Computer, v. 33, n. 7, p. 99-102, 2000.

[MCH12] MC HUGH, Martin. Integrating agile practices with plan-driven medical device software development. 2012.

Agile Alliance. 2016 [acesso 2016 fevereiro 25]. Disponível em: https://www.agilealliance.org/

[BEC99] BECK, Kent. Embracing change with extreme programming. Computer, v. 32, n. 10, p. 70-77, 1999.

[RAM10] RAMESH, Balasubramaniam; CAO, Lan; BASKERVILLE, Richard. Agile requirements engineering practices and challenges: an empirical study. Information Systems Journal, v. 20, n. 5, p. 449-480, 2010.

[BEC01] BECK, Kent et al. Manifesto for agile software development. 2001.

[VOG06] VOGEL, D. Agile Methods: Most are not ready for prime time in medical device software design and development. DesignFax Online, v. 2006, 2006.

[MAR03] MARTIN, Robert Cecil. Agile software development: principles, patterns, and practices. Prentice Hall PTR, 2003.

[CAO08] CAO, Lan; RAMESH, Balasubramaniam. Agile requirements engineering practices: An empirical study. Software, IEEE, v. 25, n. 1, p. 60-67, 2008.

[LEI01] LEINO, Virve. Documenting Requirements Traceability Information: A Case Study. Helsinki University of Technology, 2001.

[WIN10] WINKLER, Stefan; PILGRIM, Jens. A survey of traceability in requirements engineering and model-driven development. **Software and Systems Modeling (SoSyM)**, v. 9, n. 4, p. 529-565, 2010.

[GAY16] GAYER, Sebastian et al. Lightweight Traceability for the Agile Architect. **Computer**, v. 49, n. 5, p. 64-71, 2016.

[RAM97] RAMESH, Balasubramaniam et al. Requirements traceability: Theory and practice. Annals of software engineering, v. 3, n. 1, p. 397-415, 1997.

[DAV93] DAVIS, Alan et al. Identifying and measuring quality in a software requirements specification. In: Software Metrics Symposium, 1993. Proceedings., First International. IEEE, 1993. p. 141-152.

[IEEE830-1998] Ieee recommended practice for software requirements specifications. Tech. Rep. 830-1998, 1998.

[IEEE29148-2011] ISO/IEC/IEEE 29148:2011, Systems and software engineering – Life Cycle Processes – Requirements Engineering, ISO/IEC/IEEE, Tech. Rep., 2011.

[ZEL14] ZELLER, Marc; HÖFIG, Kai; ROTHFELDER, Martin. Towards a cross-domain software safety assurance process for embedded systems. In: Computer Safety, Reliability, and Security. Springer International Publishing, 2014. p. 396-400.

[MAD13] MADER, Patrick et al. Strategic traceability for safety-critical projects. **IEEE software**, v. 30, n. 3, p. 58-66, 2013.

[RAM95] RAMESH, Balasubramaniam et al. Implementing requirements traceability: a case study. In: Requirements Engineering, 1995., Proceedings of the Second IEEE International Symposium on. IEEE, 1995. p. 89-95.

[PIN04] PINHEIRO, Francisco AC. Requirements traceability. In: Perspectives on Software Requirements, ed.Dordrecht, The Netherlands: Kluwer Academic Publishers, p. 91-113, 2004.

[WIN10] WINKLER, Stefan; PILGRIM, Jens. A survey of traceability in requirements engineering and model-driven development. Software and Systems Modeling (SoSyM), v. 9, n. 4, p. 529-565, 2010.

[AHM07] AHMAD, Arslan; GHAZALI, Muhammad Ahmad. Documenting requirements traceability information for small projects. In: Multitopic Conference, 2007. INMIC 2007. IEEE International. IEEE, 2007. p. 1-5.

[CLE14] CLELAND-HUANG, Jane et al. Software traceability: trends and future directions. In: Proceedings of the on Future of Software Engineering. ACM, 2014. p. 55-69.

[LEM09] Lempia, David L and Miller, Steven P. Requirements Engineering Management Handbook. U.S. Department of Transportation, Federal Aviation Administration, 2009. DOT/FAA/AR-08/32.

[KAL05] M. Kalinowski and G. H. Travassos. Software technologies: The use of experimentation to introduce ispis, a software inspection framework, into the industry. In Proceedings of ESELAW05, 2005.

[FAG99] FAGAN, Michael E. Design and code inspections to reduce errors in program development. IBM Systems Journal, v. 38, n. 2/3, p. 258, 1999.

[IEEE11073-20601] IEEE. ISO/IEEE 11073-20601: health informatics – point-of-care medical device communication – Part 20601: optimized exchange protocol standards, 1st ed.; 2010.

[CAR07] CARROLL, Randy et al. Continua: An interoperable personal healthcare ecosystem. **IEEE Pervasive Computing**, v. 6, n. 4, 2007.

[SAN15] SANTOS, Danilo FS; ALMEIDA, Hyggo O.; PERKUSICH, Angelo. A personal connected health system for the Internet of Things based on the Constrained Application Protocol. **Computers & Electrical Engineering**, v. 44, p. 122-136, 2015.

[PLO14] PLOURDE, Jeffrey; ARNEY, David; GOLDMAN, Julian M. Openice: An open, interoperable platform for medical cyber-physical systems. In: **Cyber-Physical Systems (ICCPS), 2014 ACM/IEEE International Conference on**. IEEE, 2014. p. 221-221.

[HOS00] HÖST, Martin; REGNELL, Björn; WOHLIN, Claes. Using students as subjects—a comparative study of students and professionals in lead-time impact assessment. Empirical Software Engineering, v. 5, n. 3, p. 201-214, 2000.

[BAS94] Basili, V. R., Caldiera, G., and Rombach, H. D. (1994). The Goal Question Metric Approach, volume II, pages 528–532. Encyclopedia of Software Engineering.

[PRE11] PRESSMAN, Roger S. Engenharia de software: uma abordagem profissional. 7ª Edição. Ed: McGraw Hill, p.374, 2011.

[IEEE982.2-1988] IEEE Guide for the Use of IEEE Standard Dictionary of Measures to Produce Reliable Software. 1988.

[ALV15] ALVES, Daniela de Castro Pereira. Engenharia de requisitos em projetos ágeis: um mapeamento sistemático baseado em evidências da indústria. 2015.

[JAQ12] JAQUEIRA, A. et al. Desafios de Requisitos em Métodos Ágeis: uma revisão sistemática. In: 3rd Brazilian Workshop on Agile Methods, São Paulo. 2012.

[JAQ13] JAQUEIRA, Aline de Oliveira Prata. Uso de modelos i* para enriquecer requisitos em métodos ágeis. 2013. Dissertação de Mestrado. Universidade Federal do Rio Grande do Norte.

[SIL11] SILVA, Josias Paes Junior. AGILE: uma abordagem para geração automática de linguagens i. 2011.

APÊNDICES

A Questionário Experiência dos Participantes

1) Quanto tempo de experiência você tem com projetos de software?

Até 1 ano	Até 2 anos	Mais de 2 anos

2) Qual sua experiência com engenharia de requisitos?

Nenhuma	Baixa	Média	Alta

3) Qual sua experência com metodologias ágeis?

Nenhuma	Baixa	Média	Alta

4) Qual sua experiência com rastreabilidade entre artefatos nos projetos em que participou?

Nenhuma	Baixa	Média	Alta

5) Quais técnicas / métodos você conhece em engenharia de requisitos ou metodologias ágeis? Por favor, descreva-os abaixo:

_____
_____
_____
_____

B Relatório de Verificação

IDENTIFICAÇÃO DA REVISÃO	
INSPETOR(A):	**DATA:**
ANALISTA DE REQ. (resp.):	
ARTEFATO(S) REVISADO(S)	
OBJETIVO: **REQUISITOS:** **CASOS DE USO:** **CONTEXTO:**	
Ator(es):	
Elemento(s) do Cont.: **Subsistema(s):** **OUTRO(S):**	
DEFEITO(S) DETECTADO(S)	
Item:	
Descrição:	
Tipo: () Incompletude () Inconsistência () Ambiguidade () Padronização () Rastreabilidade	
Gravidade: () Grave () Médio () Trivial	
Causa: () Falta de experiência () Falta na documentação de entrada () Falta de padrão na documentação () Omissões na análise () Falha na documentação de entrada	

C Checklist para Identificação de Defeitos

CHECKLIST PARA REVISÃO

Ambiguidade

- Os objetivos estão descritos de maneira clara e objetiva?
- Os requisitos estão descritos de maneira clara e objetiva?
- Os casos de uso estão descritos de maneira clara e objetiva?
- O contexto (atores, elementos do contexto e subsistemas) estão descritos de maneira clara e objetiva?
- Os objetivos podem ser interpretados de maneira diferente?
- Os requisitos podem ser interpretados de maneira diferente?
- Os casos de uso podem ser interpretados de maneira diferente?
- O contexto (atores, elementos do contexto e subsistemas) pode ser interpretado de maneira diferente?

Completude

- Está faltando objetivos serem descritos?
- Está faltando requisitos serem descritos?
- Está faltando casos de uso serem descritos?
- Está faltando elementos do contexto (atores, elementos do contexto e subsistemas) serem descritos?
- É suficiente o que está descrito em cada objetivo?
- É suficiente o que está descrito em cada requisito?
- É suficiente o que está descrito em cada caso de uso?
- É suficiente o que está descrito em cada elemento do contexto (atores, elementos do contexto e subsistemas)?

Padronização

- Os objetivos estão descritos conforme o padrão definido?
- Os requisitos estão descritos conforme o padrão definido?
- Os casos de uso estão descritos conforme o padrão definido?
- Os atores do contexto estão descritos conforme o padrão definido?

- Os elementos do contexto estão descritos conforme o padrão definido?
- Os subsistemas do contexto estão descritos conforme o padrão definido?
- Os atributos obrigatórios de cada objetivo descrito, estão de acordo com o padrão definido?
- Os atributos obrigatórios de cada requisito descrito, estão de acordo com o padrão definido?
- Os atributos obrigatórios de cada caso de uso descrito, estão de acordo com o padrão definido?
- Os atributos obrigatórios de cada ator do contexto descrito, estão de acordo com o padrão definido?
- Os atributos obrigatórios de cada elemento do contexto descrito, estão de acordo com o padrão definido?
- Os atributos obrigatórios de cada subsistema descrito, estão de acordo com o padrão definido?

Consistência

- Há contradição na descrição do requisito em relação ao objetivo o qual está associado?
- Há contradição na descrição do caso de uso em relação ao requisito o qual está associado?
- Há contradição na descrição do requisito em relação ao subsistema o qual está associado?
- Há contradição na descrição do caso de uso em relação ao subsistema o qual está associado?

Rastreabilidade

Rastreabilidade entre artefatos do processo de ER

- Cada requisito está diretamente relacionado com o objetivo?
- Cada caso de uso está diretamente relacionado com o requisito funcional?
- Cada requisito está relacionado com o susbsitema?
- Cada caso de uso está relacionado com o subsistema?

Rastreabilidade entre artefatos do processo de desenvolvimento ágil

- Cada user story está diretamente relacionado com o epic?

- Cada caso de teste está diretamente relacionado com a user story?

Rastreabilidade entre artefatos do processo de ER e do processo desenvolvimento ágil

- Cada epic está relacionado com o objetivo?
- Cada requisito funcional está relacionado com casos de testes?