UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA

ROGNER RANGEL AHNERT

GERÊNCIA DE CONHECIMENTO APLICADA A DESENVOLVIMENTO ÁGIL DE SOFTWARE COM SCRUM.

VITÓRIA
2014

ROGNER RANGEL AHNERT

GERÊNCIA DE CONHECIMENTO APLICADA A DESENVOLVIMENTO ÁGIL DE SOFTWARE COM SCRUM.

Monografia apresentada à Universidade Federal do Espírito Santo como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Vítor E. Silva Souza.

VITÓRIA

2014

ROGNER RANGEL AHNERT

GERÊNCIA DE CONHECIMENTO APLICADA A DESENVOLVIMENTO ÁGIL DE SOFTWARE COM SCRUM.

COMISSÃO EXAMINADORA

_____

Prof. Dr. Vítor E. Silva Souza.
Departamento de Informática – UFES
Orientador.

_____

Profª Dr. Monalessa Perini Barcellos
Departamento de Informática – UFES

_____

Profª Dr. Renata Silva Souza Guizzardi
Departamento de Informática – UFES

Vitória, 14 de novembro de 2014.

A Deus e aos meus pais, Darci e Valéria

RESUMO

A utilização das metodologias ágeis para a gerência e construção de projetos de software está sendo indicada como uma solução para os problemas relacionados ao seu desenvolvimento, promovendo melhorias tais como tempo de entrega, qualidade do produto final, aderência ao cronograma e elaboração de documentos que exijam cada vez menos complexidade para serem construídos. A teoria ágil tem como princípio a reflexão das atividades realizadas, ocasionando um projeto com melhores resultados e a formação de uma equipe independente e mais eficaz, modelando conforme as necessidades são apresentadas. Como parte da proposta ágil, as organizações acabam registrando apenas as informações mais relevantes dos projetos, ao contrário das abordagens tradicionais que acabam registrando e documentando quase tudo. Diante deste cenário a gerência de conhecimento proporciona ao desenvolvimento ágil uma forma simplificada e eficiente de registrar essas informações e de armazenar todo o conhecimento gerado durante a elaboração de projetos. O objetivo deste trabalho é identificar quais tipos de informações podem ser capturadas na elaboração de projetos que utilizam o modelo *Scrum* dentro do contexto de organizações. Essa análise permitirá a construção de um portal de gerência de conhecimento que auxilie a tomada de decisão no ambiente organizacional.

Palavras-chave: Gerência de Projetos, Metodologias Ágeis, Scrum, Gerência de Conhecimento.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Os softwares estão cada vez mais desenvolvidos e sendo incorporados quase que em todos os setores do cotidiano. Apesar dos avanços da produção e desenvolvimento de software, grande parte dos sistemas continuam apresentando inúmeros problemas durante sua construção. O desenvolvimento de software confiável correto e entregue dentro do prazo e custos estipulados ainda é um acordo difícil de ser formalizado entre clientes e empresas desenvolvedoras (NETO, 2004).

As iniciativas para melhorar o processo de desenvolvimento de software vem crescendo significativamente na Engenharia de Software. Entende-se por melhorar o processo ter uma maior compreensão do mesmo a fim de melhorar a qualidade do produto e reduzir custo, risco e prazo de desenvolvimento (SOMMERVILLE, 2003).

De acordo com a abordagem escolhida pela empresa, uma decisão pode tornar o projeto um sucesso ou um fracasso total. Alguns fatores como desligamento de membros com conhecimento aprofundado do sistema, código de baixa qualidade e legibilidade e reformulação de requisitos (HIGHSMITH, 2004) são fatores que podem levar o projeto ao fracasso. Somando esses problemas com a pressa e pressão dos clientes em ter o produto final o quanto antes possível, algumas organizações acabam não usando nenhum processo, levando a efeitos desastrosos na qualidade do produto final e gerando dificuldade na entrega do produto nos prazos e custos definidos durante o acordo.

Para que essas empresas entreguem os produtos de acordo com a expectativa dos clientes e no tempo adequado, as metodologias ágeis aparecem como uma solução, trazendo credibilidade e eficiência para a equipe, pois o fluxo de informações e conceitos estão sempre organizados, produzindo projetos sem desperdício de tempo e mão de obra (AMARAL et al., 2011).

Os projetos que utilizam as metodologias ágeis acabam sendo desenvolvido de maneira mais direta e com menores complicações (CAMARA, 2005), tornando o processo mais simples, com melhores resultados e com menor esforço durante o gerenciamento, proporcionando um melhor resultado para o cliente (AMARAL et al., 2011). Sendo assim, metodologias ágeis acabam gerando um grande interesse em áreas de desenvolvimento de software.

Ao revisar as informações diariamente ou semanalmente, durante a criação e

planejamento, os membros da equipe de projetos aprendem com as alterações que são inevitáveis durante a construção do projeto (COHN, 2005), mas para que essas informações possam ser consultadas deve existir um processo que identifique, avalie, capture e propague todo o conhecimento identificado durante a construção de projetos na organização.

A elaboração de projetos que empregam o processo de desenvolvimento de software *Scrum* permite criar projetos mais bem adaptados às novas realidades organizacionais de forma ágil. Ele provavelmente não vai solucionar todos os problemas relacionados a projetos, mas ajudará a identificá-los (SOUZA, 2008). Este processo é uma fonte capaz de gerar muitas informações, isto porque a própria estrutura do *Scrum* está fundamentada em diversas reuniões, onde muita informação é produzida. Com a quantidade de informação gerada, se faz necessário utilizar ferramentas de apoio para gerência do conhecimento. Um portal de conhecimento é importante para adquirir conhecimento sobre novas tecnologias e novos domínios, compartilhar conhecimento, capturar lições aprendidas e analisar os resultados durante a execução do projeto, procurando registrar o que deu certo ou errado (RUS et al, 2002). A identificação de qual conhecimento pode ser útil e deve ser registrado é um processo com alto grau de dificuldade no contexto das organizações desenvolvedoras de softwares (COELHO, 2010).

É nesse contexto que a gerência de conhecimento se liga ao modelo *Scrum*. Uma organização que orienta suas ações com base em lições aprendidas, nas discussões envolvidas e no conhecimento já produzido, certamente não vai repetir os mesmos erros já cometidos. Embora cada projeto de software seja quase único, muitas experiências e informações comuns existem entre eles e essas experiências podem facilitar os desenvolvedores na execução de suas atividades, principalmente desenvolvedores menos experientes e mais novos na organização. A reutilização do conhecimento evita que falhas se repitam e facilita na resolução de novos problemas.

1.1 Objetivos

O objetivo geral deste trabalho é desenvolver um sistema de gerência de

conhecimento, como uma extensão do Portal de Gerência do Conhecimento (COELHO, 2010) aplicado ao desenvolvimento ágil com o processo *Scrum* em ambientes organizacionais. Esse objetivo geral pode ser detalhado nos seguintes objetivos específicos:

- Empregar técnicas e práticas da metodologia ágil do processo *Scrum* em projetos de uma organização (uma empresa de desenvolvimento de software).
- Identificar quais conhecimentos podem ser relevantes no contexto do desenvolvimento ágil do processo *Scrum*.
- Modelar e desenvolver um sistema de Gerência de Conhecimento como uma extensão do portal de Gerência do Conhecimento desenvolvido em (COELHO, 2010) de modo a permitir o registro de lições referentes à análise das práticas ágeis do processo *Scrum*.
- Integrar o componente desenvolvido ao ambiente ODE (*Ontolgogy-based software Development Environment*) (FALBO et al., 2013).

1.2 Histórico do Trabalho

Este trabalho foi desenvolvido a partir da realização de um estágio supervisionado realizado em uma empresa de desenvolvimento de software. A empresa está localizada em Vitória – ES desde 1992 e nos últimos anos tem atuado fortemente no mercado de soluções de software para o setor público. Dentre os clientes mais importantes estão bancos e prefeituras da região.

Apesar de existir há mais de 20 anos, a empresa tem pouca preocupação com a gerência de projetos, não sendo utilizada nenhuma metodologia de desenvolvimento nos projetos existentes. Apenas em alguns projetos são utilizadas abordagens tradicionais de desenvolvimento, onde são registrados documentos de requisitos, de especificação de requisitos e de projeto.

A proposta inicial foi empregar técnicas e práticas das metodologias ágeis em alguns projetos da organização. Foram utilizados os modelos *Scrum* e *Extreme Programming* (BECK et al., 2000). O *Scrum* tem seu processo baseado fortemente na

relação gerencial da organização e o *Extreme Programming* está principalmente ligado a parte de codificação. O emprego das técnicas permitiram gerar o conhecimento para desenvolver a ferramenta.

A parte do controle gerencial de projetos era a maior preocupação dos administradores da organização, uma vez que não existia nenhuma metodologia que auxiliasse os gerentes e desenvolvedores. Nesse contexto foram utilizadas algumas técnicas e praticas do *Scrum*. À medida que algumas práticas do modelo *Scrum* eram realizadas, uma grande quantidade de informações eram apresentadas e deveriam ser registradas para auxiliar o desenvolvimento de novos projetos que também utilizassem o *Scrum*. Sendo assim, percebeu-se a necessidade da utilização de um sistema que capturasse, avaliasse e propagasse o conhecimento adquirido a todos os membros da organização.

1.3 Organização do Texto

Esta monografia é estruturada em quatro capítulos e contém, além da presente introdução, os seguintes capítulos:

- *Capítulo 2 – Gerência de Conhecimento e Desenvolvimento Ágil: Scrum*: apresenta uma revisão da literatura acerca de temas relevantes ao contexto deste trabalho, a saber: Gerência de Conhecimento, Metodologias de Desenvolvimento, Desenvolvimento Ágil e Processo *Scrum*.
- *Capítulo 3 – Portal de Gerência de Conhecimento aplicado a Desenvolvimento Ágil: Scrum*: apresenta as fases de análise e projeto fundamentais para a construção do portal. A fase de análise inclui especificação dos requisitos, casos de usos e diagramas dos subsistemas apresentados e a fase de projeto apresenta a estrutura da arquitetura de software utilizada.
- *Capítulo 4 – Considerações Finais*: apresenta as conclusões do trabalho, dificuldades envolvidas, limitações e propostas de trabalhos futuros.

2 GERÊNCIA DE CONHECIMENTO E DESENVOLVIMENTO ÁGIL: SCRUM

Este capítulo aborda, brevemente, temas relevantes ao contexto deste trabalho, a saber: Gerência de Conhecimento, Metodologias de Desenvolvimento, Desenvolvimento Ágil e Processo *Scrum*.

2.1 Gerência de Conhecimento no Desenvolvimento de Software

O conhecimento adquirido na elaboração de um software é um recurso fundamental para as organizações, pois este conhecimento gera um processo de aprendizado evolutivo (SOUZA, 2006), evitando que um mesmo erro seja cometido novamente. Uma das propostas da Engenharia de Software é criar um produto com qualidade, sem desperdício de recursos e dentro do prazo estipulado. Para alcançar esse objetivo não se devem realizar melhorias que tenham foco apenas no produto, mas deve-se pensar também no processo de desenvolvimento do produto.

Mesmo que o processo de desenvolvimento de software seja algo único, o conhecimento e as experiências das pessoas envolvidas podem ajudar membros de outros projetos, na execução de suas atividades (ABECKER et al., 2001). Nesse contexto as organizações têm inserido a Gerência de Conhecimento para facilitar a troca de conhecimento entre os membros, além de facilitar o reúso e o acesso dos conhecimentos já registrados. Um dos objetivos da Gerência de Conhecimento é gerar novos conhecimentos a partir dessas experiências e lições aprendidas. A reutilização desse conhecimento ajuda na prevenção de falhas e na solução de problemas já enfrentados pela organização (SOUZA, 2006).

As exigências do mercado para a formação de um produto com qualidade estão fazendo com que as empresas desenvolvedoras de software capacitem cada vez mais seus funcionários, isso acaba gerando uma alta dependência das empresas por esses funcionários, que conhecem cada vez mais os processos e padrões da empresa. Para que essa vantagem seja eficaz o conhecimento adquirido não pode estar no nível de indivíduo (LIMA, 2004), isto porque algumas empresas costumam ter um alto grau de

rotatividade de funcionários e esse conhecimento pode ser perdido quando este funcionário deixar a empresa. O conhecimento registrado em papel também é um problema, uma vez que reutilizar e alterar esse conhecimento é um problema devido a sua dificuldade de acesso. Esses e outros problemas podem ser resolvidos se for criado um sistema ou a adoção de uma solução computacional já existente que gerencie o conhecimento dentro da organização.

As organizações que desejam inserir a gerência de conhecimento em seus processos não devem apenas registrar o conhecimento adquirido. Para ser eficaz a gerência de conhecimento deve ser aplicada da maneira correta. A sua utilização consiste em capturar e registrar o conhecimento adquirido (COELHO, 2010). Além disso, devem ser criados mecanismos que compartilhem, recuperem e propaguem o conhecimento relevante para a realização das atividades dos processos organizacionais (VILLELA, 2004). Na aplicação da Gerência de Conhecimento algumas atividades devem ser inseridas: identificação, aquisição, desenvolvimento, disseminação, uso e preservação de conhecimento (FALBO, 2004).

A adoção da gerência de conhecimento por uma organização pode ser resolvida de maneira eficaz se for utilizado um sistema de gerência de conhecimento (COELHO, 2010). A construção desse sistema implica em criar uma infraestrutura que envolve pessoas e tecnologias no ambiente organizacional para apoiar as atividades de gerência de conhecimento (LIMA, 2004).

Um sistema de gerência de conhecimento tem o objetivo de garantir que o conhecimento desejado possa estar disponível naquele exato momento em que for solicitado. O principal objetivo desse sistema é garantir que o conhecimento seja acessível e reutilizável para a organização (COELHO, 2010). Ao adotar um processo de Gerência de Conhecimento um dos problemas enfrentados pelas organizações é definir quais tipos de conhecimentos devem ser registrados na memória organizacional, isto é, quais itens de conhecimentos são úteis no contexto da organização. Essa atividade exige um profundo conhecimento das necessidades e regras da organização, portando a adoção de um sistema de gerência de conhecimento requer um alto grau de decisão.

As organizações podem registrar diversas atividades no processo de construção de software dentro do contexto de gerência de conhecimento. Uma delas pode ser uma atividade relacionada a uma metodologia específica como, por exemplo, a aplicação de desenvolvimento ágil na gestão de projetos. Esse tipo de atividade destaca-se por gerar

muitas informações relevantes na elaboração de projetos e esta informação, se capturada e transformada em conhecimento, pode ser fundamental para organização, principalmente quando utilizada em projetos futuros (COELHO, 2010).

2.2 Metodologias de Desenvolvimento

Uma metodologia de desenvolvimento de software é um conjunto de instruções recomendadas para o desenvolvimento de software que combina padrões e procedimentos a serem utilizados pelos membros da organização.

Esse conjunto de práticas recomendadas são ações realizadas em fases distintas da elaboração de projetos, com objetivo de organizá-lo e ordená-lo, de forma a simplificar a sua gerência (SOMMERVILLE, 2003). Segundo Pressman (2001), o desenvolvimento de software com alta qualidade é resultado de um conjunto de ações requeridas de modo a apoiar seu desenvolvimento. Apesar de existir inúmeros processos de desenvolvimento, existem atividades comum a todos eles (SOMMERVILLE, 2003), a saber: especificação, desenvolvimento, validação e evolução do software são exemplos de atividades genéricas que, quando utilizadas, contribuem na qualidade final do produto.

Um dos principais problemas enfrentados pelas organizações desenvolvedoras de software é entregar o produto dentro do prazo estipulado (FAGUNDES, 2005), com orçamentos previstos e com qualidade, atentando para as necessidades do cliente. A Engenharia de Software fornece metodologias que apoiam os desenvolvedores e projetistas durante o processo de desenvolvimento entre elas existem abordagens tradicionais, ágeis e outras que não estão no contexto deste trabalho.

O gerenciamento tradicional de projetos é baseado no planejamento detalhado do projeto, na formalidade dos documento e processos que deve ser construídos e monitorados (FAGUNDES, 2005). O modelo tradicional utiliza técnicas para o levantamento de requisitos e compreensão do problema antes mesmo do início de desenvolvimento do projeto. Quando os requisitos são levantados na fase de análise, é realizado um planejamento para que as mudanças possam ser monitoradas na elaboração do projeto.

O gerenciamento ágil prioriza a execução, sugerindo que os requisitos devem ser levantados aos poucos e que deve haver uma forte estrutura de planejamento para que

as mudanças possam ocorrer (ZANATTA, 2004). Portanto, a principal diferença das duas abordagens está na forma de como as mudanças são tratadas durante a execução do projeto. A abordagem tradicional planeja o projeto buscando controlar e realizar o mínimo possível de mudanças, em contrapartida a abordagem ágil incentiva as mudanças nos requisitos e planejamento contínuo no projeto.

Além disso, a abordagem tradicional tem fundamentos voltados para a documentação, tornando essa metodologia inflexível e problemática em relação a mudanças. Caso ocorra alguma mudança sem planejamento, isso implicará em um projeto com um custo maior e, assim, a qualidade do produto pode ser comprometida. Portanto, essa metodologia deve ser empregada em situações na qual o projeto está bem definido e não sofrerá mudanças que impactam seu escopo. As metodologias ágeis devem ser utilizadas quando esse escopo não está bem definido, permitindo que o projeto evolua durante seu desenvolvimento.

De acordo com Pressman, (2001) por melhor que sejam planejados os requisitos, os projetos sofrem mudanças. Sendo assim, as metodologias ágeis são bem adequadas para organizações pequenas (ZANATTA, 2004), que possuem projetos pequenos, que podem se adaptar às mudanças e não possuem uma documentação bem detalhada, como as produzidas nos grandes projetos que empregam as metodologias tradicionais.

2.3 Desenvolvimento Ágil

As metodologias ágeis foram criadas como uma resposta às chamadas metodologias tradicionais. Estas últimas, embora sejam abordagens clássicas de desenvolvimento de projetos, são alvos de críticas (HIGHSMITH, 2004), uma vez que não apoiam mudanças e focam no planejamento detalhado e excessivo. As metodologias ágeis surgiram a partir da necessidade de buscar uma solução mais simples aos projetos de software sob condições constantes de mudanças dentro do ambiente organizacional (AMARAL et al., 2011).

A adoção das metodologias ágeis implica em uma alteração da forma que as organizações trabalham. A equipe de desenvolvimento terá que se acostumar com a baixa documentação produzida e com os constantes envolvimentos dos clientes na elaboração do projeto que estará presente nas diversas etapas (COHN, 2005), gerando assim uma

maior informalidade nas relações humanas da organização. Seu emprego é recomendado em projetos que tenham de 4 a 9 membros alocados, por causa da necessidade de relação e comunicação entre os membros da equipe e pela pouca documentação produzida (COHN, 2005).

As metodologias ágeis de software tratam dos problemas das constantes mudanças, apesar de possuir diversas divergências com a abordagem tradicional (COHN, 2005), ela não é contra os modelos adotados nesta última. As metodologias ágeis não se desvinculam das ferramentas, documentações e planejamentos adotados na abordagem tradicional. Essas particularidades são deixadas em segundo plano (FILHO et al., 2005) se comparada às principais preocupações das metodologias ágeis envolvidas tais como: constantes mudanças, maiores interações entre os indivíduos e produção do software criado dentro do prazo estipulado.

É fundamental entender que as metodologias ágeis produzem principalmente uma mudança comportamental e não prescrevem um processo descritivo (FILHO, 2005). Elas agrupam não somente questões relacionadas à gerência de projetos como no *Scrum,* mas também boas práticas de programação como é o caso do *Extreme Programming (XP).* Cada uma das metodologias ágeis tem suas particularidades e suas práticas distintas. Consequentemente, é natural que hoje em dia se adotem modelos híbridos em projetos. Esses modelos são uma união dessas metodologias ágeis, onde o melhor de cada metodologia é empregado, resultando em um projeto a fim de produzir software com qualidade (HIGHSMITH, 2004).

Além de optarem por uma documentação simplificada, evitando excessos para que auxilie o processo de desenvolvimento de software, elas partem da premissa que somente a documentação necessária deve ser gerada e que esses documentos tenham realmente algum valor (COHN, 2005). Outro aspecto importante é que as metodologias ágeis priorizam os aspectos humanos em vez de modelos de engenharia. Assim os envolvidos são os principais condutores do sucesso ou fracasso do projeto.

2.4 Processo Scrum

O Scrum é um processo ágil com foco voltado para a gerência de projetos (SCHWABER, 2004). Sua proposta é entregar um produto de maior qualidade, no menor

tempo possível. O conceito de ágil no contexto de projetos de software é construir apenas o que o cliente deseja e valoriza, sem uso exagerado de processos que exijam formalidades tais como documentação extensa e planeamento detalhado das atividades realizadas, que podem reduzir o cronograma do projeto.

A utilização desta metodologia permite a criação de projetos que podem se adaptar as mudanças ocorridas na sua elaboração, de modo que os membros da equipe possam trabalhar e produzir um software em uma ambiente onde essas mudanças são inevitáveis. Portanto ela é indicada para pequenas organizações e em projetos onde não se pode prever tudo que vai ocorrer (SCHWABER, 2004).

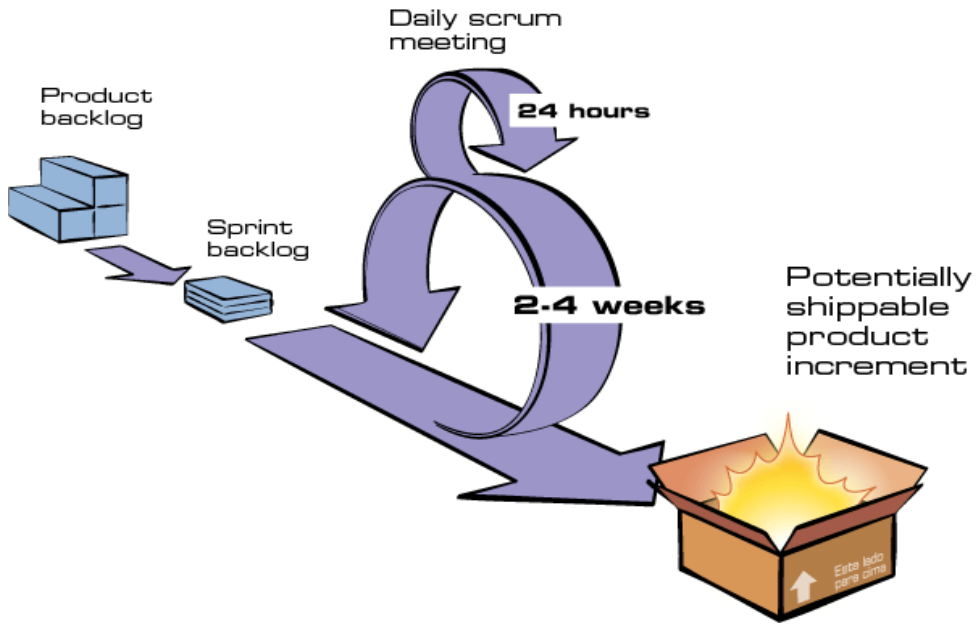

Figura 1: Gerência de Projetos com a utilização do Scrum (Scrum, 2014).

A Figura 1 representa o fluxo da gerência do *Scrum*. O ciclo do *Scrum* é iniciado na fase de planejamento onde é mantida uma lista de funcionalidades a serem implementadas pela equipe. Essa lista é denominada *Product Backlog,* e suas funcionalidades são definidas pelo cliente e priorizadas segundo algum valor.

O *Product Backlog* representa uma lista de funcionalidades do que se deseja desenvolver, a equipe realiza uma reunião inicial de planejamento, denominada *Sprint Planning Meeting*, na qual os itens desta lista são priorizados pelo cliente, chamado no *Scrum* de *Product Owner.* Durante a reunião de planejamento, a equipe define quais

funcionalidades poderão ser atendidas dentro da *Sprint*. A lista destas funcionalidades escolhidas representa o *Sprint Backlog* na Figura 1.

A *Sprint* representa a iteração do processo de desenvolvimento do *Scrum*: basicamente é o período de tempo determinado para cumprir determinada tarefa. Cada *Sprint* pode durar de 2 a 4 semanas e ao final é gerado um incremento, isto é, o produto que deve estar “pronto”. A definição de “pronto” é relativo, uma vez que o incremento pode ser melhorado em próximas *Sprints.*

Durante a execução da *Sprint,* são realizadas reuniões diárias denominadas *Daily Scrum Meeting* para acompanhar o andamento, ajustar as informações do projeto e disseminar conhecimento, identificando os possíveis acontecimentos e medidas adotadas.

Após o término da *Sprint,* acontece a reunião chama de *Sprint Review*, onde a equipe apresenta as funcionalidades que foram concluídas durante a *Sprint*. Além disso o *Scrum Master* realiza uma reunião com a equipe, denominada de *Sprint Retrospective Meeting*, para discutir os detalhes da *Sprint*, tais como produtividade, problemas, erros e acertos, revisando e melhorando seus métodos e valores.

A seguir são apresentados os principais papéis e responsabilidades implementados pelo *Scrum:*

- ***Product Owner:*** É o dono do produto. É ele que define e prioriza os requisitos, quem toma as decisões e define o que precisa ser feito.
- ***Scrum Master:*** Um conhecedor do processo e técnicas do *Scrum*, responsável por fazer a metodologia funcionar dentro da organização. Deve ensinar e proporcionar treinamento a todos os envolvidos no processo. Sua função é garantir que todos tenham as ferramentas para cumprir suas tarefas. Além disso pode trabalhar com o *Product Owner* na estruturação dos requisitos.
- ***Scrum Team:*** Desenvolve os itens que foram priorizados em cada iteração com a intenção de produzir o incremento ao final de cada *Sprint*. São responsáveis pelo desenvolvimento do projeto.

3 PORTAL DE GERÊNCIA DE CONHECIMENTO APLICADO AO DESENVOLVIMENTO ÁGIL: SCRUM

O Portal de Gerência de Conhecimento (COELHO, 2010) tem como principal objetivo gerenciar itens de conhecimento. Dentro do portal, é possível criar itens de conhecimento, de lições aprendidas e de conhecimentos relativos a uma discussão. A principal característica que o diferencia de outros portais é um conjunto de funcionalidades que foram criadas especificamente para gerência de conhecimento. É possível avaliar, valorar e buscar itens de conhecimento, além de apresentar uma lista com itens de conhecimentos mais acessados, isso tudo com o objetivo de guardar e transmitir os itens de conhecimento mais importantes dentro da organização.

Esse portal foi criado para gerenciar itens de conhecimentos com características gerais de Engenharia de Software, sendo que novos projetos visando ao conhecimento mais específico podem ser construídos baseados nele. Recentemente, foi desenvolvido um sistema de Gerência de Conhecimento como uma extensão desse portal voltado para o apoio ao processo de teste de software (SPECIMILLE, 2014).

O objetivo deste trabalho é construir uma nova extensão do portal de conhecimento de modo a permitir o registro de lições referentes à análise das práticas ágeis de projetos que utilizam o *Scrum*.

3.1 Análise e Especificação dos Requisitos

Este capítulo aborda os resultados da Engenharia de Requisitos do Portal de Gerência de Conhecimento aplicado a Desenvolvimento Ágil. São apresentados a descrição do mini mundo, casos de uso e diagrama de classes dos subsistemas. A tabela completa com os requisitos pode ser visualizada no Apêndice A.

3.1.1 Descrição do Mini Mundo

O Portal de Gestão do Conhecimento é uma ferramenta para auxiliar o gerente de projetos e os desenvolvedores na construção de projetos que utilizam o *Scrum*. Durante o ciclo de desenvolvimento de projetos de software, vários conhecimentos podem ser

alcançados. Membros da organização podem adquirir o conhecimento através de experiências sobre resolução de problemas. Diante disso apenas alguns membros ou grupos de membros podem adquirir esse conhecimento. Isto pode ser um problema muito grave para qualquer organização, ainda mais se a organização possuir um alto grau de rotatividade de funcionários. Essa deficiência, que atinge grande parte das organizações, pode ser minimizada se todo esse conhecimento for registrado com auxílio de uma ferramenta.

A importância principal do portal é guardar conhecimento de projetos que utilizam a metodologia ágil do processo *Scrum*, para que possam ser consultados na construção de novos projetos dentro da organização.

Itens de conhecimento são importantes para tornar o conhecimento acessível e reutilizável para a organização. Um item de conhecimento pode ser: lições aprendidas, conhecimento relativo a uma discussão, descritivo e *Sprint*. Do item de conhecimento referente a uma lição aprendida é importante saber: tipo, descrição do problema, solução adotada e resultado esperado. Do item de conhecimento referente a uma discussão é importante saber: o conhecimento adquirido, pontos fortes, pontos fracos e o link da discussão. Do item de conhecimento referente a um descritivo é importante saber: nome, descrição, url da página e o artigo servido como referência. E por fim sobre uma *Sprint* deseja-se saber: quantidade realizadas, duração em dias, data de início, dificuldade, tipo de *Sprint,* objetivos e se a *Sprint* foi revisada. Para cada item de conhecimento, são criados eventos *Scrum* para aquele item. Esses eventos podem ser reuniões diárias, reuniões de planejamento, revisão, ou seja, todos os possíveis eventos do *Scrum.* De um evento *Scrum* deseja-se saber: nome e descrição.

Durante a construção de uma *Sprint* é importante guardar quais funcionalidades foram implementadas dentro daquela *Sprint.* Funcionalidades são implementadas por um recurso humano e possuem: nome, descrição, prioridade e o estado da funcionalidade. Em algum momento, pode-se querer registrar algum acontecimento referente aquela *Sprint.* Um acontecimento guarda as seguintes informações: sumário, descrição, tipo e dificuldade. Se for registrado algum acontecimento, pode-se querer guardar a ação adotada para aquele acontecimento. Da medida, deseja-se saber: medida adotada, complexidade e estado da medida. Além disso, dentro da *Sprint,* pode-se querer registrar trechos de códigos mais importantes, como, por exemplo, abrir uma conexão com banco de dados é um conhecimento adquirido referente a código. Sendo assim é registrado o título e o trecho do código.

Para que um item de conhecimento possa ficar disponível para todos os membros no portal, ele precisa ser avaliado por um gerente de conhecimento. A avaliação guarda os seguintes itens: nota da correção, nota de completude, nota de consistência, nota de utilidade, nota de aplicabilidade, parecer, resultado final e a data da avaliação. Além disso para consultar os itens de conhecimento que foram mais importantes e que foram mais acessados é importante que esse conhecimento seja valorado. Sobre a valoração é importante saber: grau de utilidade, comentário e a data da valoração.

3.1.2 *Subsistemas do Portal Gerência de Conhecimento – Metodologias Ágeis: Scrum*

A Figura 2 mostra os subsistemas identificados no contexto deste projeto e são descritos a seguir:

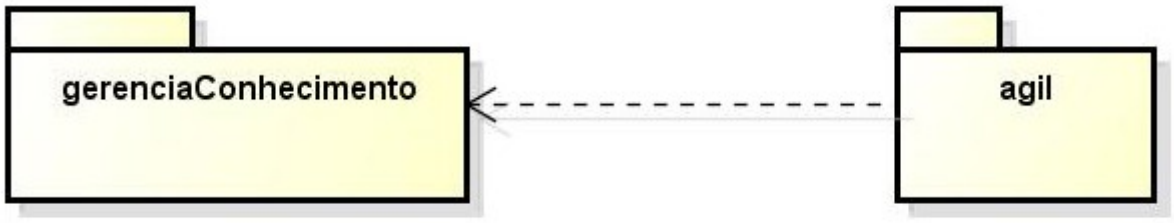

Figura 2: Diagrama de Pacote e os Subsistemas

- **gerenciaConhecimento:** Envolve todas as funcionalidades relacionadas com o controle do conhecimento de projetos dentro da organização, abrangendo controle de itens de conhecimento, lições aprendidas e itens relativo a uma discussão. Este subsistema foi desenvolvido em (COELHO, 2010), porém foram feitas modificações em sua modelagem no contexto desse trabalho. Os trechos que sofreram alterações encontram-se no Apêndice B..
- **agil:** Envolve toda a funcionalidade relacionada com o controle de projetos em andamento que adotaram as metodologias ágeis na sua composição. A criação de itens de conhecimentos para as metodologias ágeis, como *Sprint* e itens descritivos.

3.1.3 Casos de Uso

O modelo de casos de uso visa capturar e descrever as funcionalidades que um sistema deve prover para que os atores interajam com o mesmo. Utilizam esse sistema os desenvolvedores da organização (projetistas, analistas, programadores, etc.), que são os responsáveis pela manutenção dos itens de conhecimento, e o gerente de conhecimento, funcionário da organização que tem papel gerencial, realizando atividades específicas, como a avaliação de itens de conhecimento.

A Figura 3 apresenta o diagrama de casos de uso do subsistema *gerenciaConhecimento*. O subsistema *gerenciaConhecimento* foi modelado em uma Dissertação do Mestrado (COELHO, 2010), portanto as descrições dos casos de uso podem ser encontradas no trabalho original. Os casos de uso que sofreram alguma correção ou inclusão encontram-se descritos no Apêndice B.

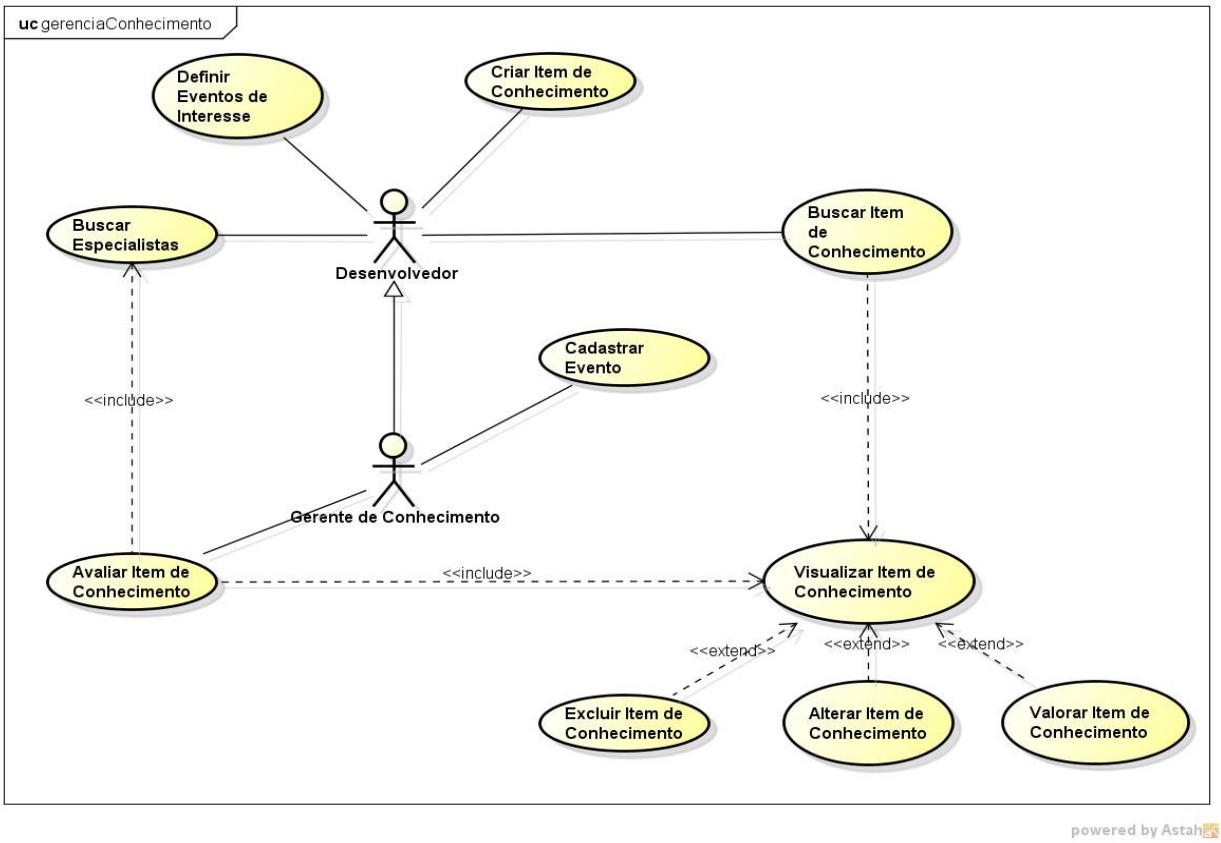

Figura 3: Diagrama de Casos de Uso do Subsistema gerenciaConhecimento

Para que o portal funcione de maneira simples e eficiente, o gerente de conhecimento deve primeiramente cadastrar eventos *Scrum*, o evento é ligado a um item de conhecimento. Essa união facilita o usuário a encontrar um item de conhecimento que está procurando, uma vez que será realizado um filtro a partir desse evento. Desenvolvedores podem criar e buscar itens de conhecimento sempre que desejarem. Ao visualizar um item encontrado, um desenvolvedor pode excluí-lo, alterá-lo ou valorá-lo. O desenvolvedor pode ainda encontrar especialistas para avaliar seu item de conhecimento. Apenas o gerente de conhecimento pode avaliar um item de conhecimento, essa característica permite que o portal registre apenas os itens de conhecimentos relevantes na memória organizacional.

3.1.3.1 Subsistema ágil

A Figura 4 apresenta o diagrama de casos de uso do subsistema *agil*.

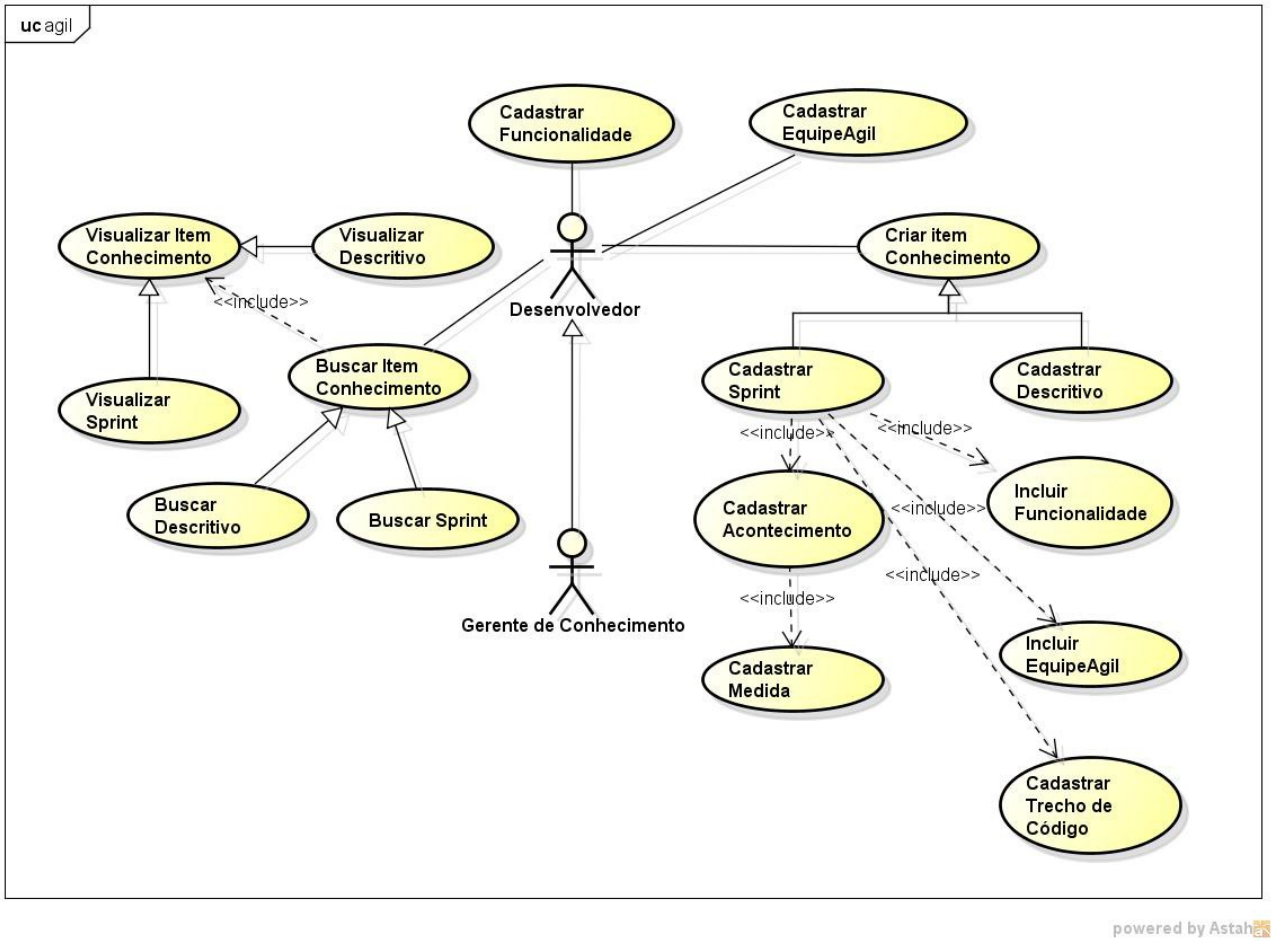

Figura 4: Diagrama de Casos de Uso do Subsistema agil

Desenvolvedores e gerentes de conhecimento podem criar outros tipos de itens de conhecimento. Podem cadastrar um item descritivo. Esse item é responsável por armazenar o conhecimento referente a uma metodologia ou a uma nova tecnologia empregada na organização. Podem ainda cadastrar uma *Sprint, e* realizar uma série de ações sobre ela. É fundamental que eles cadastrem funcionalidades e a equipe ágil no projeto, antes que se inicie uma *Sprint*. Quando a *Sprint* é iniciada, eles podem incluir uma equipe ágil, selecionar as funcionalidades que pertencem à *Sprint,* e realizar outras ações. Quando a *Sprint* for iniciada, desenvolvedores podem encerrar as funcionalidades que já tiverem sido implementadas e podem retomar o andamento da mesma, para realizar alguma alteração. Ainda sobre a *Sprint,* os usuários podem cadastrar acontecimentos que foram gerados na execução da *Sprint. C*aso esse acontecimento tenha sido esclarecido através de alguma solução, pode registrar uma medida para este acontecimento. Acontecimentos e medidas sofrem pequenas alterações em alguns dos seus campos, mas não podem ser excluídos da memória organizacional, uma vez que é um conhecimento já adquirido. Ainda é possível cadastrar um trecho de código dentro da *Sprint*, para apoiar membros na execução de atividades realizadas, relacionadas a codificação de projetos. Análogo ao subsistema *gerenciaConhecimento* os desenvolvedores podem criar e buscar esses itens de conhecimento sempre que desejarem. Ao visualizar um item encontrado, um desenvolvedor pode excluí-lo, alterá-lo ou valorá-lo.

3.1.4 Diagrama de Classes

O modelo conceitual estrutural visa capturar e descrever as informações (classes, associações e atributos) que o sistema deve representar para prover as funcionalidades.

A seguir, são apresentados os diagramas de classes e uma breve descrição para cada um dos subsistemas identificados no contexto deste trabalho.

A Figura 5 apresenta o diagrama de classes do subsistema *gerenciaConhecimento.*

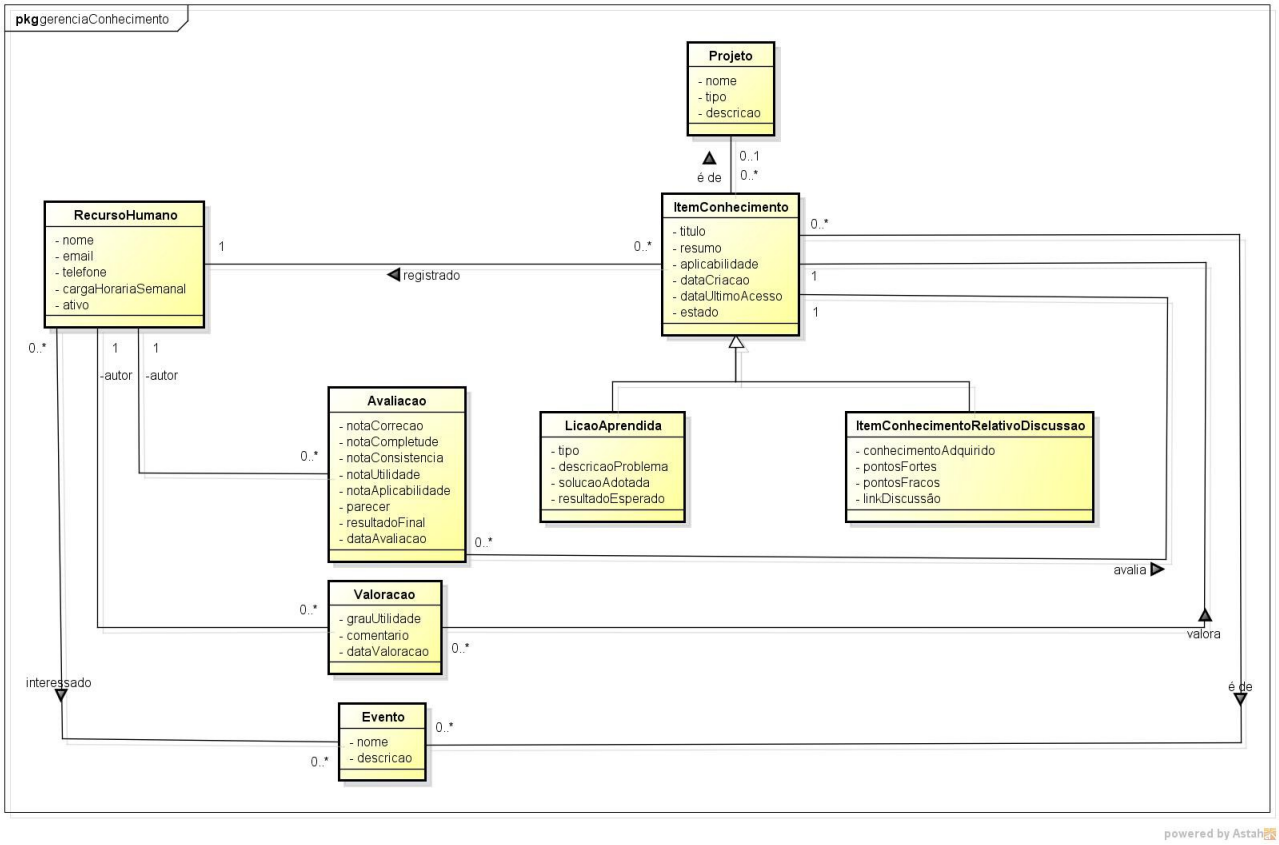

Figura 5: Diagrama de Classes do Subsistema gerenciaConhecimento

O subsistema *gerenciaConhecimento* possui **Projeto** que é capaz de armazenar todas as informações dos itens de conhecimentos criados.

Grande parte das organizações que produzem software desejam adotar uma política de gerência de conhecimento. Porém, devido à grande carga de tarefas acumuladas, acabam deixando esse desejo em segundo plano. Para permitir a criação de conhecimentos, devem ser criados mecanismos que permitam gerenciar o conhecimento. O **Item de Conhecimento** representa esse papel e possui: projeto, recurso humano que o registrou, título, resumo, aplicabilidade, data de criação, data do último acesso e estado (aguardando avaliação, disponível, excluído ou rejeitado). Para que a manipulação de um item de conhecimento funcione de maneira adequada eles devem ser gerenciados por **Recursos Humanos.** Um recurso humano é responsável por criar, avaliar e valorar um item de conhecimento, ele possui: nome, e-mail, telefone, carga horária semanal e uma verificação da sua situação. Os **Eventos** são responsáveis por caracterizar um determinado tipo de item de conhecimento, esse conhecimento pode ser adquirido em uma reunião de revisão por exemplo, ou de alguma outra situação ainda não definida. Eventos possuem nome e descrição.

Quando uma organização aceita adotar a gerência do conhecimento na organização o próximo desafio é definir quais tipos de itens de conhecimento serão tratados e que tipo de informações podem ser capturadas sobre cada item de conhecimento. Foram definidos os seguintes itens de conhecimento para essas organizações: lições aprendidas e conhecimento relativo a uma discussão. A **Lição Aprendida** fornece mecanismos para auxiliar a como proceder em determinada situação resultado de um conhecimento adquirido de ideias, fatos, opiniões, etc. Ela possui os seguintes dados: tipo, descrição do problema, solução adotada e resultado esperado. Um **Conhecimento Relativo a uma Discussão** permite que membros das organizações compartilhem experiências, algumas que não puderam ser respondidas em reuniões com o intuito de fomentar novos conhecimentos a serem capturados. Esse item possui os seguintes dados: conhecimento adquirido, pontos fortes, pontos fracos e link da página Web relacionada à discussão.

Em um portal de gerência de conhecimento é importante para o usuário recuperar os conhecimentos mais importantes. De acordo com ABECKER et al., (2001) usuários não pretendem gastar tempo procurando informações e precisam de mecanismos que propaguem os conhecimentos mais relevantes. Para isso, são criados mecanismos para avaliar e valorar cada item de conhecimento. Todo conhecimento criado deve ser avaliado por algum membro antes que esteja disponível para todos os usuários. Essa característica é fundamental para a disseminação apenas de conhecimentos com relevância e corretos na estrutura organizacional. A **Avaliação** possui os seguintes critérios: nota de correção, nota de completude, nota de consistência, nota de utilidade, nota de aplicabilidade, parecer, resultado final e data de avaliação. A valoração é um mecanismo disponibilizado no portal para capturar a opinião do usuário sobre determinado item de conhecimento, esse *feedback* é fundamental para a manutenção dos itens de conhecimento criados na organização, é uma forma de definir quais itens efetivamente estão sendo usados e quais não estão, que podem ser atualizados ou excluídos. A **Valoração** possui os seguintes critérios: grau de utilidade, comentário e data de valoração.

3.1.4.1 Subsistema agil

A Figura 6 apresenta o diagrama de classes do subsistema *agil.*

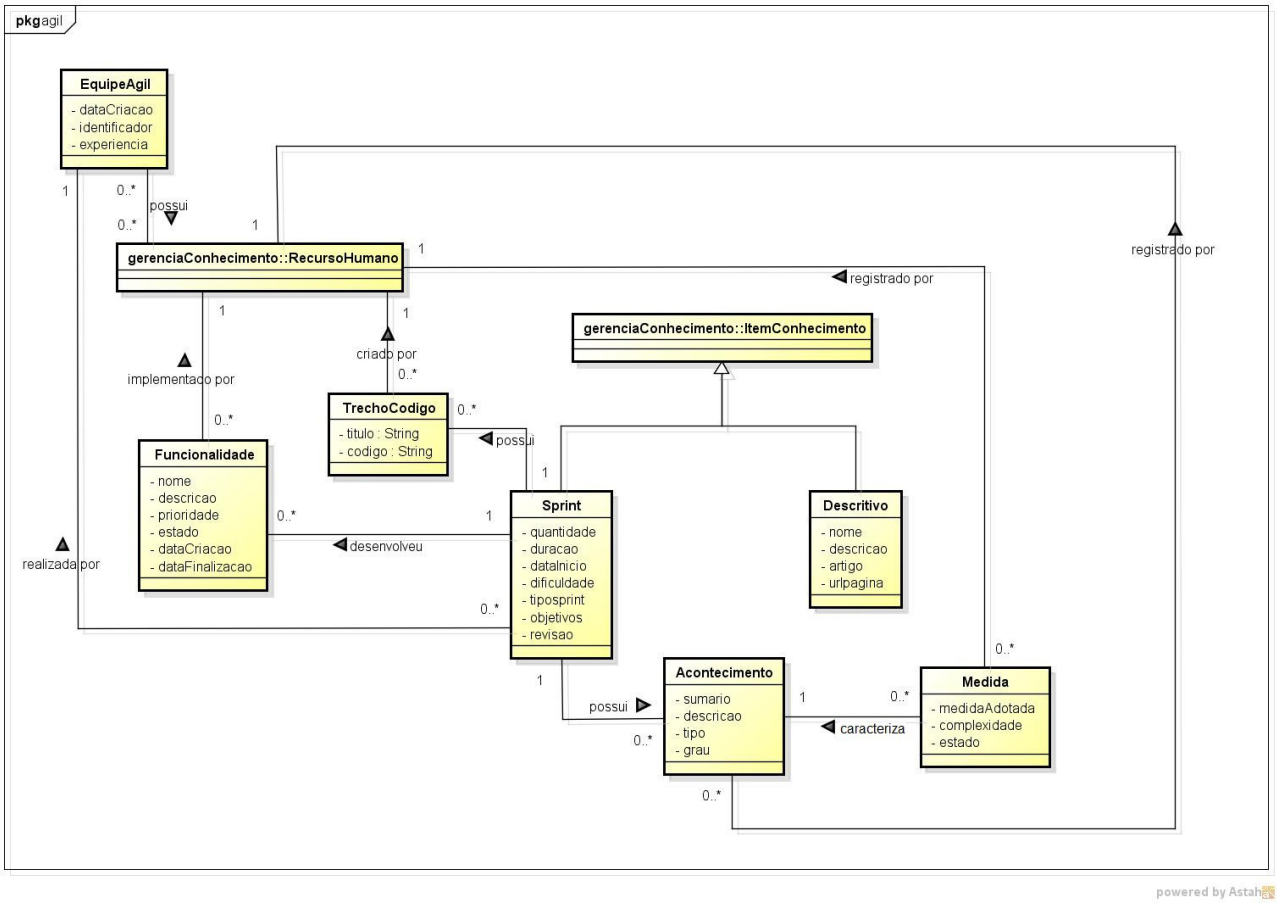

Figura 6: Diagrama de Classes do Subsistema agil

Além de Conhecimento Relativo a uma Discussão e Lição Aprendida, no subsistema *agil* um item de conhecimento pode ser do tipo **Descritivo** e **Sprint,** essas classes herdam da classe **ItemConhecimento** do subsistema *gerenciaConhecimento.*

Um item de conhecimento descritivo é um tipo de conhecimento para armazenar as informações sobre a descrição da metodologia usada no portal. Em vez de procurar e buscar esses conceitos na literatura, o portal armazenará essas informações (OLIVEIRA, 2009). Esse item de conhecimento está diretamente ligado aos membros novatos ou a membros que não conhecem o domínio da metodologia que está sendo empregada no projeto. Um **Descritivo** armazena as seguintes informações: o item de conhecimento, nome, descrição, artigo e url da página.

A equipe ágil é responsável por todo o planejamento e execução para completar o conjunto de trabalho com o qual se comprometeram a realizar dentro de uma *Sprint*. A **Equipe Ágil** possui, data de criação, um identificador, experiência (nenhuma, média, alta e muito alta) e os recursos humanos membros da equipe.

Um item de conhecimento **Sprint** é um tipo de conhecimento onde serão

armazenadas todas as informações que ocorreram na execução da *Sprint*. Ao participar de uma *Sprint,* os recursos humanos, que são membros da organização estão expostos a uma grande quantidade de objetivos, características e novas informações sobre o projeto, com isso acabam quase sempre adquirindo novas experiências. Esta experiência se transformada em conhecimento pode ser usada em futuros projetos, minimizando erros e evitando retrabalho. A **Sprint** armazena as informações da equipe ágil, quantidade necessária, duração, data de início, dificuldade (baixa, média, alta), tipo de *Sprint* (funcional ou teste), objetivos e revisão.

Durante a execução da *Sprint*, é comum surgirem opiniões, problemas e divergências para se atingir o objetivo final, essas divergências podem ser questões que tenham sido solucionadas ou estão aguardando uma solução, mas é fundamental que sejam registradas na memória organizacional. Os acontecimentos e medidas são um forma de registrar essas informações. **Acontecimento** possui o recurso humano que o registrou, a *Sprint* pertencente àquele acontecimento, sumário, descrição, tipo (funcionalidade, código, método, tecnologia) e grau (baixo, médio, alto). Caso exista algum acontecimento dentro da *Sprint*, pode ser criada uma **Medida** para esse acontecimento. Dela deve-se saber o acontecimento, o recurso humano que a registrou, a medida adotada, complexidade (baixa, média, alta) e o estado (não resolvida, resolvida e pendente).

É muito comum no ciclo de vida do projeto o cumprimento de atividades relacionadas a código que são executadas sempre que são criados ou encerrados os projetos, é um procedimento muito comum nas organizações da mesma maneira pode-se querer guardar essas informações referentes aos trechos de códigos que são executados nestas ou em outras etapas, esse artefato pode conter informações relevantes para a organização, evitando que apenas alguns membros detenham aquele conhecimento ou até mesmo facilitando a atividade para os novos membros inseridos na organização. **Trechos de Códigos** armazenam os seguintes dados: recurso humano que o registrou, sprint que foi inserida aquele trecho de código, título e código.

Quando o projeto está em curso, a quantidade de tarefas e funcionalidades concorrentes costumam ser grandes e controlar seu andamento pode ser algo complexo. É importante para o gerente de projeto e para a equipe envolvida saber o que está acontecendo, quais itens foram resolvidos e quais itens apresentaram alguma dificuldade**.** A **Funcionalidade** é responsável por armazenar essas informações, ela possui: *Sprint* que a desenvolveu, o recurso humano que a implementou, nome, descrição, prioridade

(nenhuma, baixa, normal, alta, urgente e imediato), estado (não iniciada, iniciada e finalizada), data de criação e data de finalização.

3.2 Projeto do Sistema

Após identificados e modelados os requisitos do sistema é iniciada a fase de projeto do sistema. Essa fase é uma extensão do modelo de análise e o seu principal objetivo é gerar um documento que possa ser compreendido pelos desenvolvedores e pessoas envolvidas na construção do sistema. Nesta fase, é possível ter uma noção dos recursos necessários, principalmente os recursos ligados a aspectos tecnológicos. Esses recursos englobam: tempo, investimento, arquitetura do software, linguagem de programação, *frameworks*, características de interface com o usuário e a forma de como os dados serão persistidos.

O portal de gerência de conhecimento desenvolvimento ágil: *Scrum* é uma extensão do portal de gerência de conhecimento em engenharia de software (COELHO, 2010). Levando-se em consideração essas características e por ser um sistema que utilizou a infraestrutura do projeto ODE-Web, decidiu-se por manter a implementação do sistema usando a linguagem de programação Java, o banco de dados relacional PostgreSQL e a API JPA com o *framework* de mapeamento objeto relacional *Hibernate* (BAUER et al., 2005).

Neste capítulo, serão apresentados os componentes da arquitetura do subsistema *agil,* que representa o subsistema realizado neste trabalho.

3.2.1 Arquitetura do Sistema

A arquitetura de software do sistema portal de gerência de conhecimento do processo *Scrum* está organizada na combinação de camadas e partições. Os subsistemas *gerenciaConhecimento* e *agil* mostrados na Figura 7, estão organizados em três camadas: Camada de Interface com o Usuário (ciu) onde é feita a interação direta com o usuário, Camada de Lógica de Negócio (cln) onde fica as funções e as regras de todo o negócio e a Camada de Gerência de Dados (cgd) que é responsável por persistir as requisições da camada de lógica de negócio no banco de dados.

Para organizar a camada de lógica de negócio dos subsistemas apresentados, foi escolhido o padrão Camada de Serviço (FOWLER, 2003). A camada de lógica de negócio foi dividia em: Componente de Domínio do Problema (*cdp*) e Componente de Gerência de Tarefas (*cgt*). O *cgt* é responsável por receber as requisições da interface e o *cdp* é responsável pelo tratamento das classes na etapa da construção do modelo elaborado na fase de análise.

Além do subsistema gerenciaConhecimento (COELHO, 2010) e do subsistema ágil foi reutilizado a componente _infraestruturaBase e _infraestruturaCRUD do projeto ODE-Web. Na Figura 7 é apresentado o projeto da arquitetura do sistema.

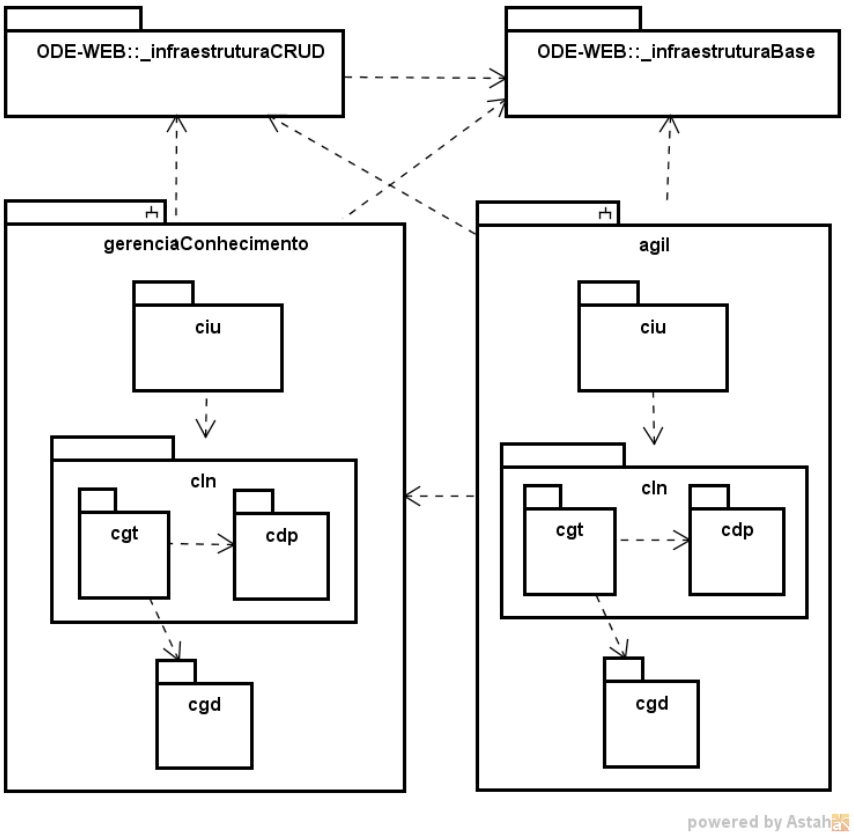

Figura 7: Projeto da Arquitetura de Software

3.2.1.1 Componente de Domínio do Problema (CDP)

A Figura 8 apresenta o diagrama de classes do *cdp* do subsistema *agil*. Este

diagrama tem como base o diagrama de classes do subsistema ágil em tempo de análise, mostrado na Figura 6. Os tipos enumerados identificados foram inseridos visando atender os requisitos da categoria facilidade de operação e eficiência (tática “O sistema deve ser fácil de usar, devendo-se evitar a digitação desnecessária de informações, de modo a dar agilidade ao processo”). Além disso por ser tratar de um diagrama em nível de projetos, os tipos identificados na fase de análise, foram apresentados como classes e a visibilidade dos atributos foram explicitadas.

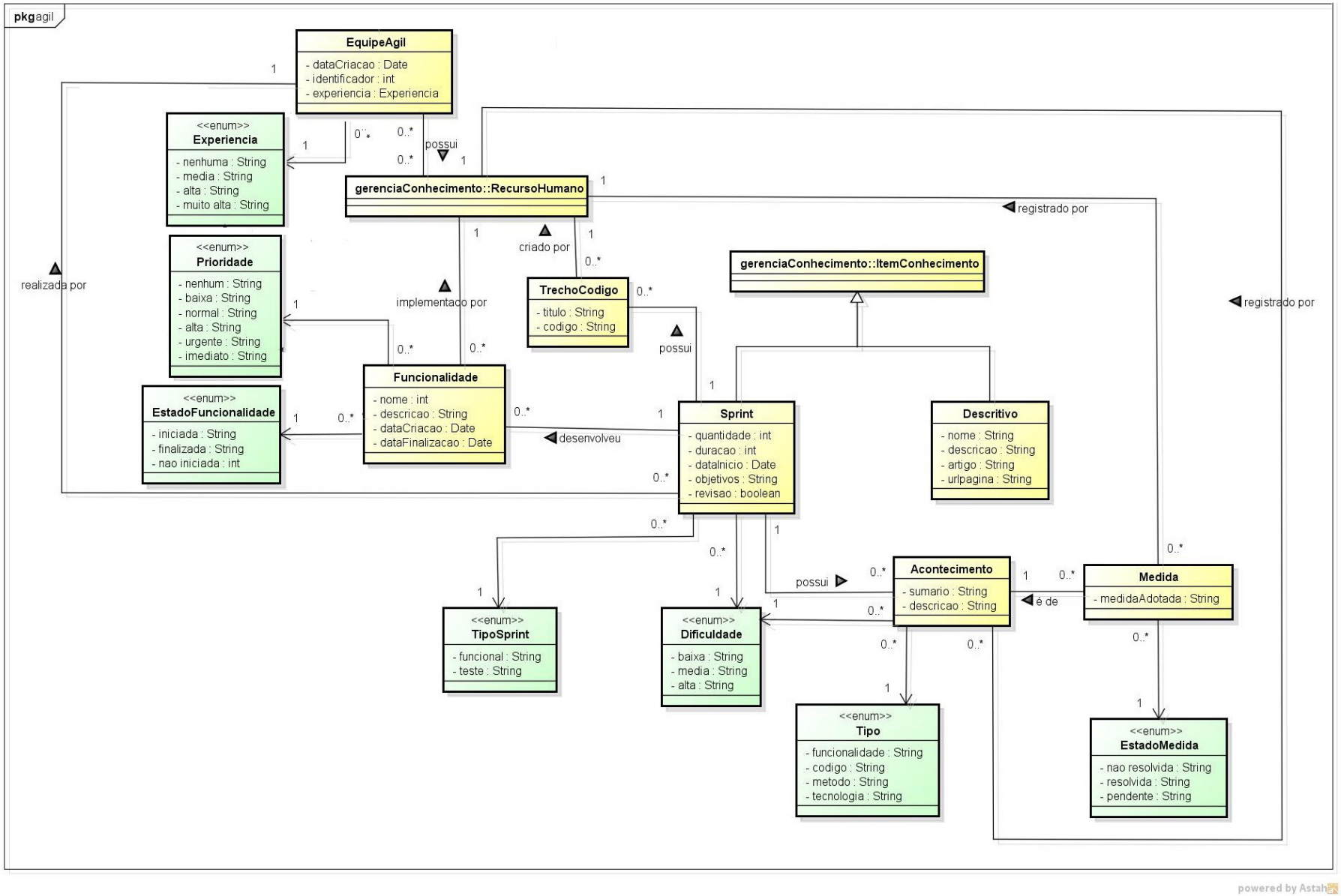

Figura 8: Diagrama de Classes do CDP do Subsistema agil

3.2.1.2 Componente de Gerência de Tarefas (CGT)

Na componente *cgt*, foi uma decisão de projeto procurar criar uma classe gerenciadora de tarefa para cada caso de uso identificado. A única exceção foram as classes *AplCadastrarEquipeAgil*, *AplCadastrarFuncionalidade*, *AplCadastrarSprint* e *AplCadastrarFuncionalidade que, por* possuirem casos de uso de menor complexidade e fortemente relacionados às suas respectivas classes, tiveram esses casos de usos agrupados. A Tabela 3 relaciona as classes do *cgt* e os casos de usos a eles relacionados.

Tabela 1: Classes do CGT e Casos de Uso do Subsistema agil

Classe	Caso de Uso
AplCadastrarAcontecimento	Cadastrar Acontecimento
AplCadastrarDescritivo	Cadastrar Descritivo Visualizar Descritivo Buscar Descritivo
AplCadastrarEquipeAgil	Cadastrar EquipeAgil Incluir EquipeAgil
AplCadastrarFuncionalidade	Cadastrar Funcionalidade Incluir Funcionalidade
AplCadastrarMedida	Cadastrar Medida
AplCadastrarSprint	Cadastrar Sprint Visualizar Sprint Buscar Sprint
AplCadastrarTrechoCodigo	Cadastrar Trecho de Código

3.2.2 Camada de Interface com o Usuário (CIU)

A Figura 9 apresenta o diagrama de classes (parcial) da *ciu* do Subsistema *agil.* As classes com prefixo Apl apresentadas na figura herdam da _infraestruturaCRUD, enquanto a controladora do sistema é uma generalização da controladora base da

_infraestruturaBase do projeto ODE-Web.

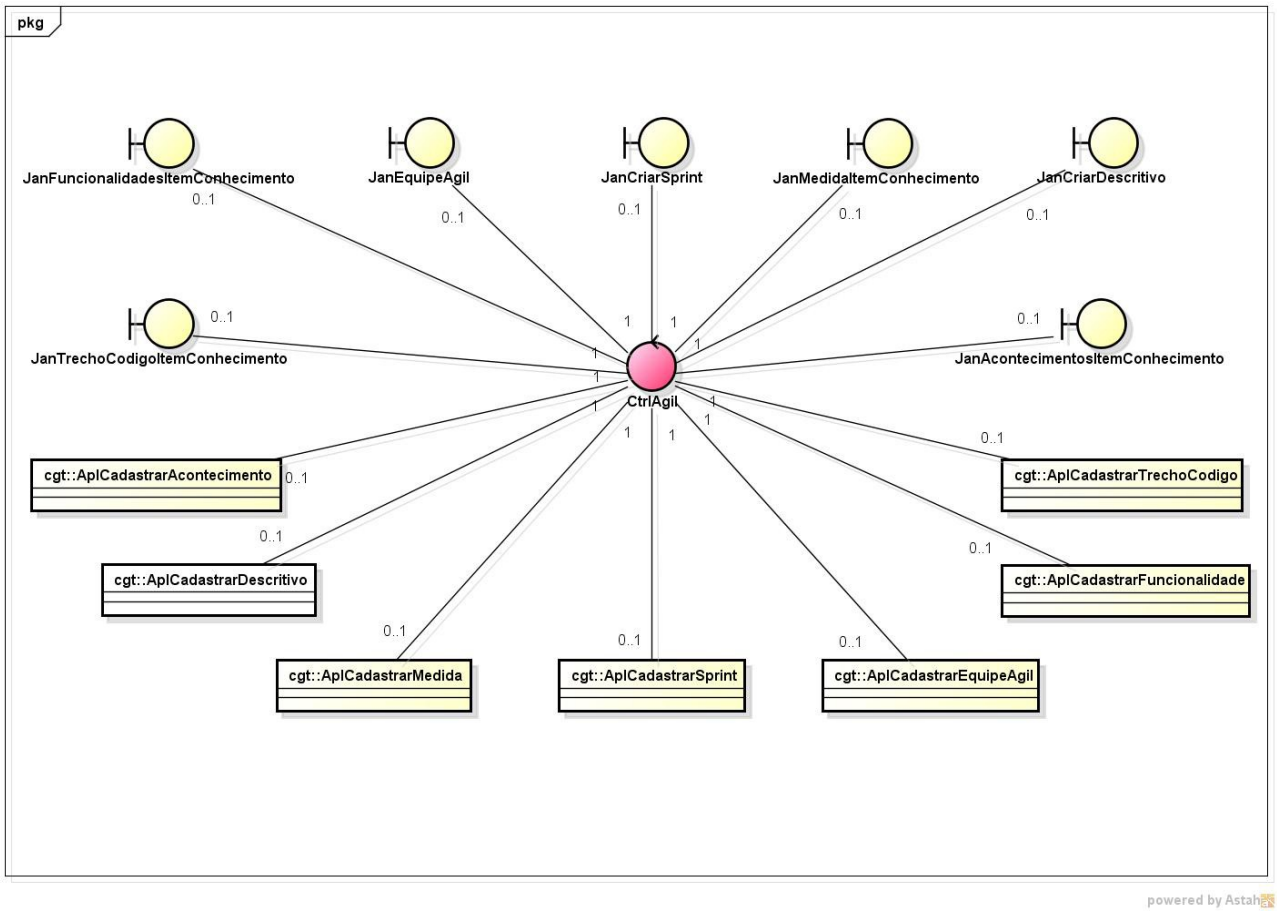

Figura 9: Diagrama de Classes (parcial) do CIU do Subsistema agil

3.2.3 Camada de Gerência de Dados (CGD)

Foi adotada a infraestrutura do projeto *ODE-Web* e a persistência de dados foi realizada no banco de dados relacional Postgres. Para realizar a persistência foi utilizado o *framework* para mapeamento objeto-relacional *Hibernate* (BAUER et al., 2005) adotando o padrão DAO, que prescreve a separação das regras de negócio das regras de acesso ao banco. Seguindo a orientação da estrutura do projeto *ODE-Web,* para cada classe de domínio foi criada uma classe *DAOImpl* e uma interface *DAO* para a mesma classe, dentro da componente *cgd*. A Figura 10 apresenta um resumo da infraestrutura de persistência do projeto *ODE-Web*.

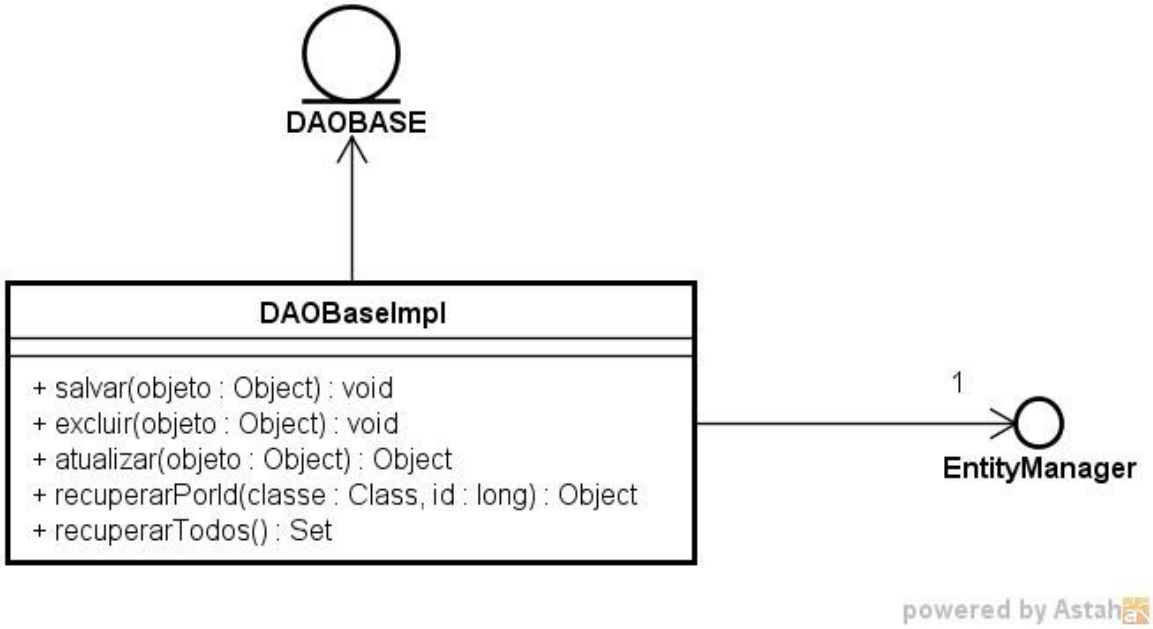

Figura 10: Diagrama da Infraestrutura de Persistência do Projeto ODE-Web

3.3 Apresentação do Sistema

A Figura 11 apresenta a configuração inicial do sistema. No menu ***Arquivos*** estão as funcionalidades gerais do Portal, neste menu é possível realizar os *CRUD* mais básicos (acrônimo de *Create, Read, Update e Delete* na língua inglesa). Dentro do menu existe as seguintes opções: cadastrar recurso humano, cadastrar projetos, cadastrar usuários, cadastrar eventos Scrum, cadastrar equipe ágil, cadastrar funcionalidades e a opção sair, para o usuário finalizar o portal.

A janela da esquerda (Figura 11) apresenta as ferramentas e o resumo da participação do usuário identificado no portal. Através dessa janela, o usuário vai armazenar os itens de conhecimento da organização, utilizando o botão ***Criar itens de Conhecimento.*** Para recuperar os itens de conhecimentos criados, o usuário deve acessar ***Buscar itens de Conhecimento*** na mesma janela.

A janela central apresenta informações de acordo com a configuração do perfil do usuário atual. Se o usuário atual for um gerente de conhecimento, como representado na Figura 11, serão apresentados os ***Itens Pendentes de Avaliação*** para o gerente aprovar ou rejeitar os itens de conhecimento apresentados na lista. Caso o usuário atual apresente um perfil que não é de gerente de conhecimento, será apresentada a janela de ***Itens Criados*** com todos os itens de conhecimento criados por aquele usuário.

A janela da direita (Figura 11) apresenta um resumo de todo o conhecimento

gerado pelo portal. Esse resumo é importante para o usuário, pois são apresentados os conhecimentos mais relevantes dentro da organização, isto é, aqueles itens de conhecimento que foram mais acessados e estão caracterizados como os mais importantes. Além dos itens mais acessados também é apresentada uma listagem com os itens de conhecimento mais recentes dentro do portal.

Uma vez criado algum item de conhecimento ele não estará disponível para os outros usuários do portal até que algum gerente de conhecimento avalie esse item. Quando avaliado o item estará disponível no portal. Somente recursos humanos com o perfil de gerente de conhecimento podem avaliar, aprovar e rejeitar e itens de conhecimentos.

Figura 11: Configuração inicial do Portal Gerência de Conhecimento

Quando acionado o botão para criar itens de conhecimento, o usuário poderá criar itens de conhecimento dos seguintes tipo: ***Lição Aprendida, Conhecimento Relativo a uma Discussão, Descritivo e Sprint.*** A janela da Figura 12 apresenta o conhecimento referente ao item ***Criar Descritivo***. Esse item de conhecimento serve para armazenar o conhecimento referente aos métodos que serão usados, por exemplo, por se tratar de um portal sobre metodologia ágil baseado no *Scrum*, pode-se armazenar informações explicando sobre o processo e algum artigo ou página web servido como base para adquirir esse conhecimento. Uma vez armazenado esse conhecimento, novos membros do portal não necessitam buscar na literatura a explicação da metodologia usada dentro

daquela organização, já que todo o esse conhecimento estará disponível no portal.

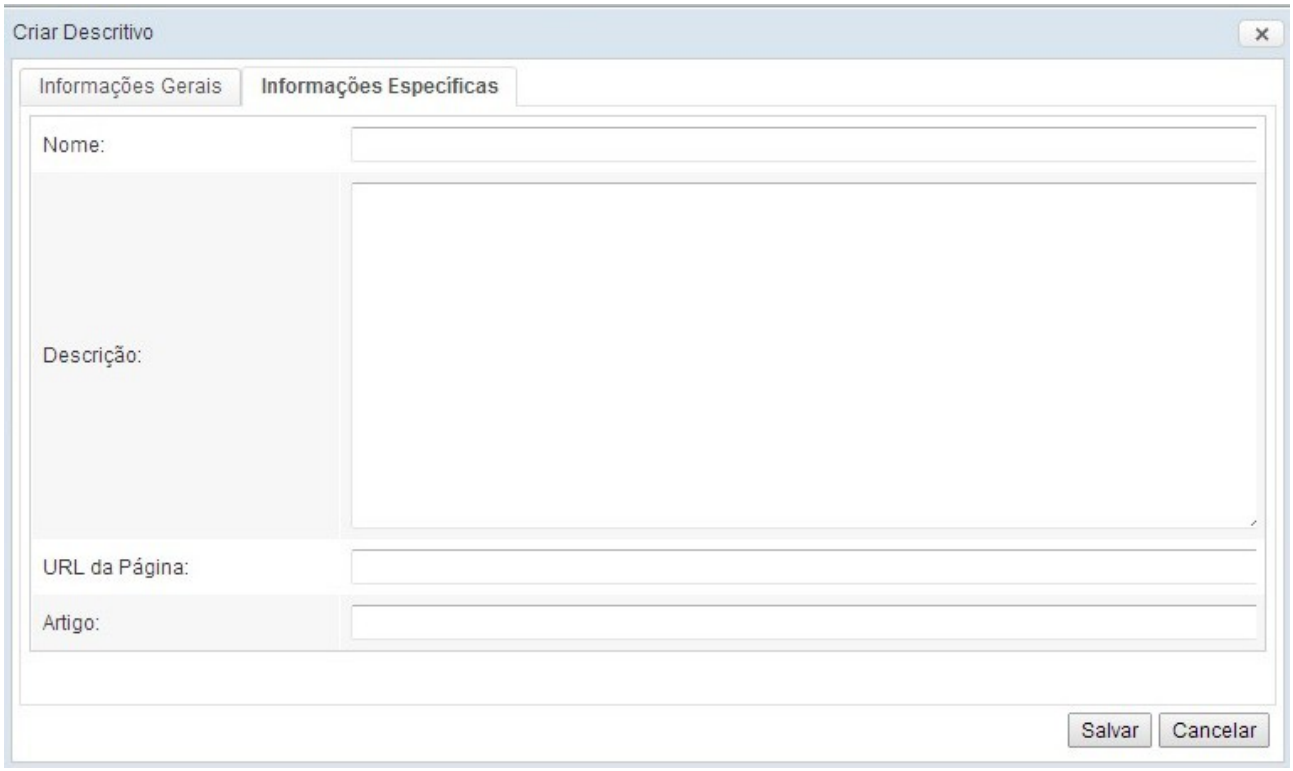

Figura 12: Janela Criar Item de Conhecimento Descritivo

Se o item de conhecimento for do tipo *Sprint* será apresentada a janela ***Criar Sprint.*** A Figura 13 apresenta o resumo dessa janela. A aba "*informações gerais*" apresenta as informações que estão presentes em todos os quatro tipos de itens de conhecimento disponíveis para serem criados, nela são armazenadas informações tais como: título, resumo, aplicabilidade e data de criação. A aba "*informações específicas*" apresenta as informações específicas de cada item de conhecimento. A Figura 13, apresenta as informações específicas da *Sprint* tais como: tipo da *Sprint* (funcional ou teste), objetivos, equipe ágil, se a *Sprint* foi ou não revisada, duração e dificuldade.

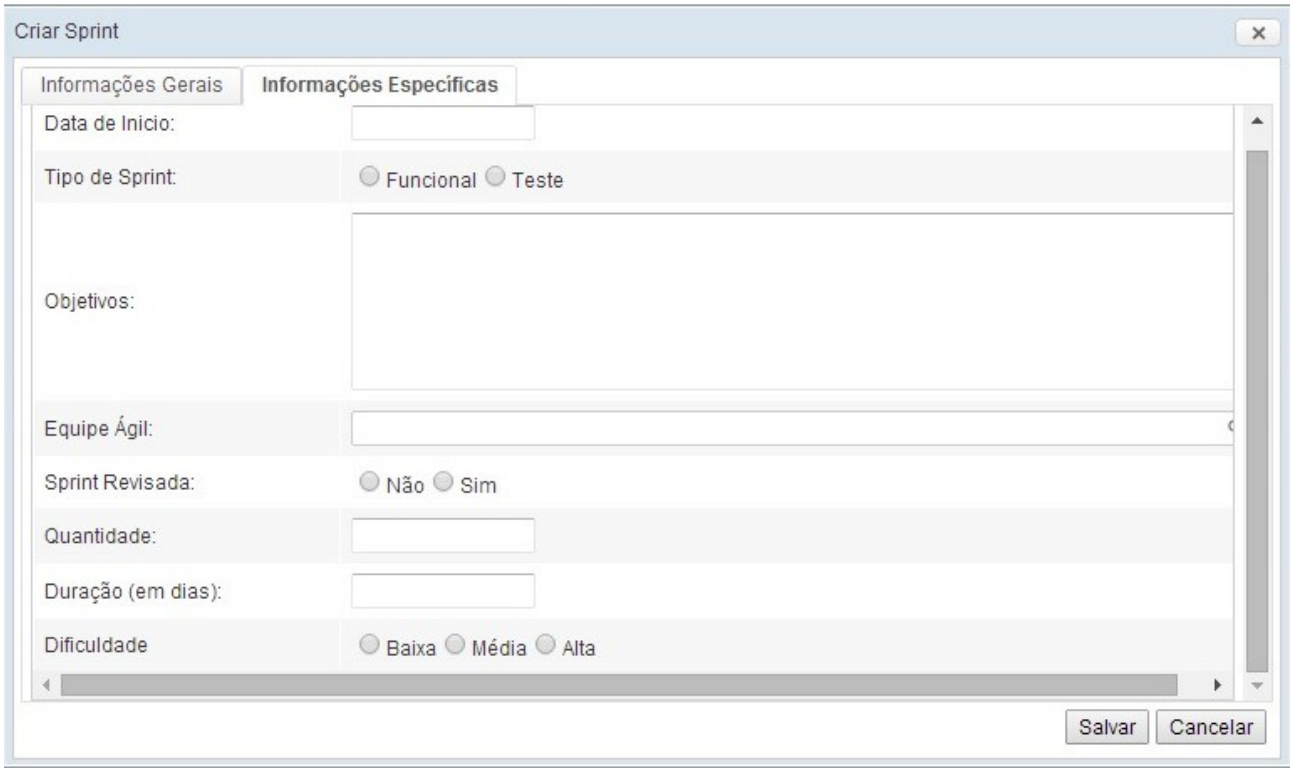

Figura 13: Janela Criar Item de Conhecimento Sprint

A Figura 14 apresenta uma ***Visualização da Sprint*** com todas as operações que são possíveis sobre este item de conhecimento. Na aba "*informações gerais*" é apresentado o conteúdo das informações na etapa da criação da *Sprint*. As abas Avaliações, Valorações, Equipe Ágil, Acontecimentos, Funcionalidades e Trecho de Código são as operações que podem ser efetuadas sobre a *Sprint* e serão explicadas de maneira mais detalhada a seguir.

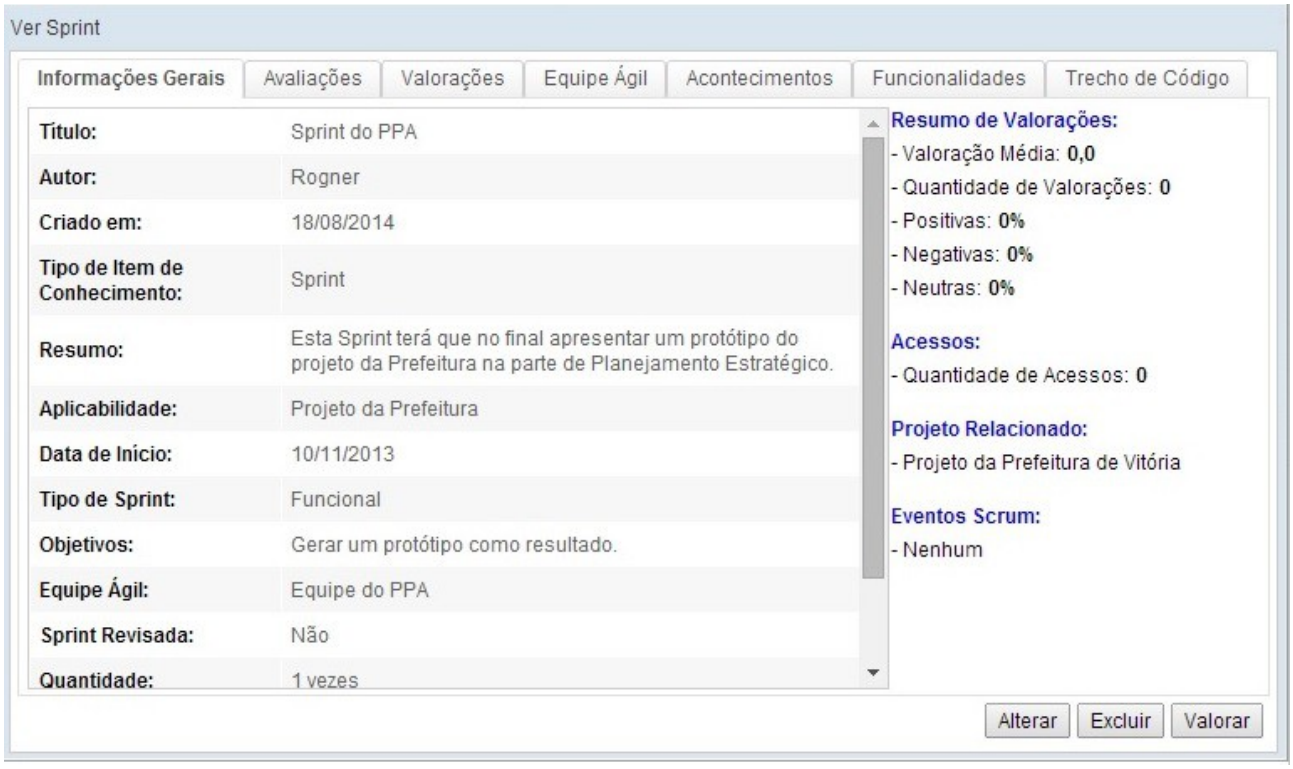

Figura 14: Janela Visualizar Item de Conhecimento Sprint

Quando acionada a aba Funcionalidades, o usuário poderá incluir as funcionalidades que serão desenvolvidas em cada *Sprint*. A Figura 15 apresenta o cadastro de funcionalidades. Esse cadastro é feito no Menu ***Arquivos*** da tela inicial, ou seja, é criado antes mesmo do item de conhecimento *Sprint* existir. Como a funcionalidade ainda não pertence a nenhuma *Sprint,* ela é criada com o estado de não iniciada. Além do seu estado, são armazenadas as seguintes informações: nome, descrição e prioridade.

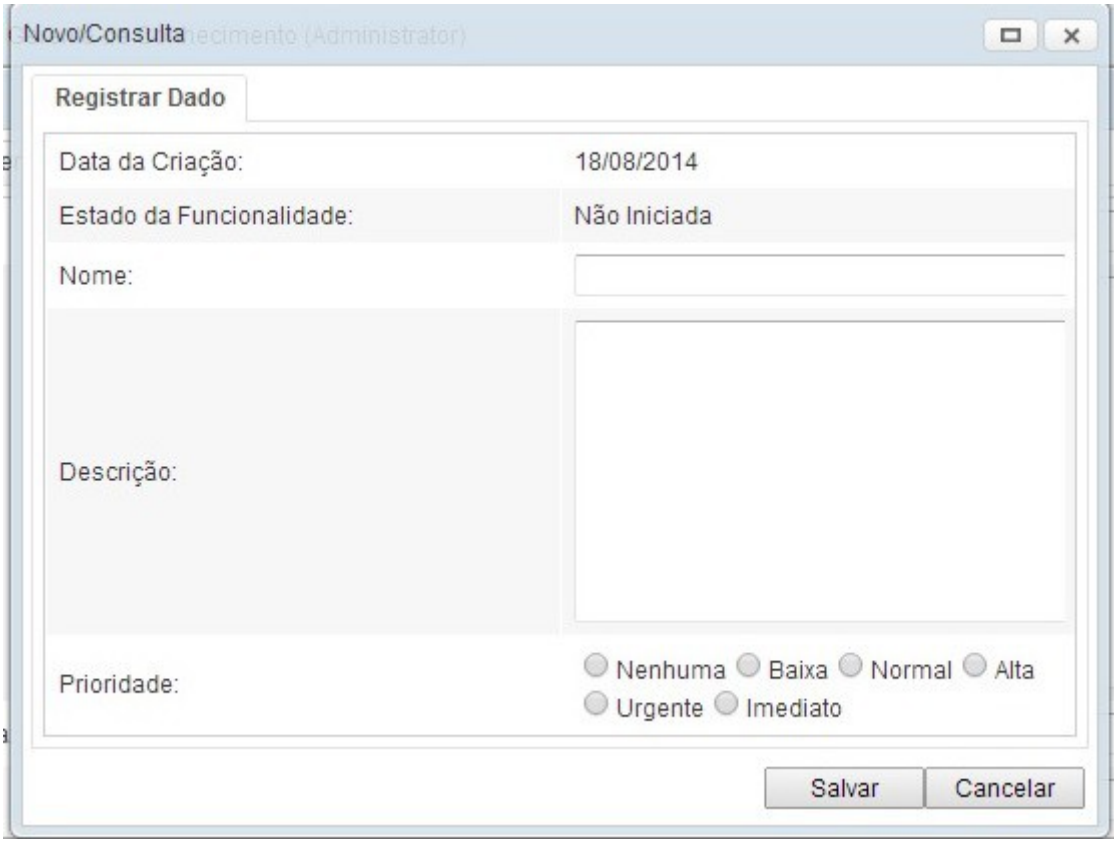

Figura 15: Janela Criar Nova Funcionalidade

Uma vez criada, a funcionalidade será apresentada na lista de ***Funcionalidades Criadas,*** apresentada na Figura 16. Essa janela representa uma simulação do *Product Backlog* do processo *Scrum*, isto é, o usuário que representa o papel do *Product Owner* da organização preenche essa lista de acordo com o risco, prioridade e necessidade da funcionalidade.

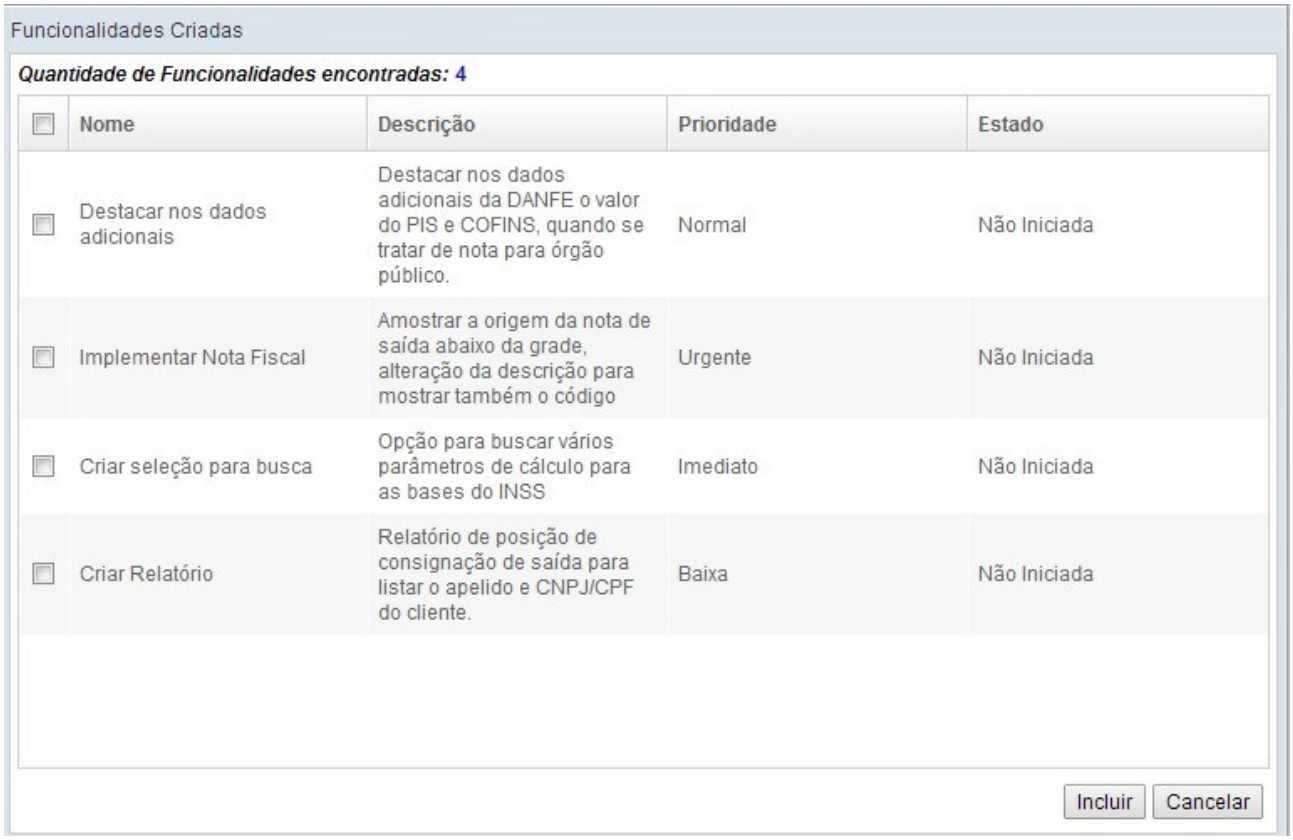

Figura 16: Janela Visualizar Funcionalidades Criadas

Na aba "*Funcionalidades*" da Figura 17, o usuário poderá ***Incluir Funcionalidades*** para aquela *Sprint*. Nesta janela o usuário poderá selecionar as funcionalidades do "*Product Backlog*" (Figura 16) que serão implementadas dentro daquela *Sprint*. A Figura 17 representa o "*Sprint Backlog*", ou seja, é a lista de funcionalidades que o "*Scrum Team*" se compromete a fazer em uma *Sprint*.

A Figura 17 apresenta o resultado da seleção das funcionalidades escolhidas pelo desenvolvedor. Ao ser incluída em uma *Sprint,* a funcionalidade altera seu estado de não iniciada para iniciada, observe que na lateral esquerda de uma funcionalidade na Figura 17, existe uma barra vermelha. A barra vermelha representa funcionalidades iniciadas e a barra verde representada funcionalidades finalizadas. Ao escolher alguma funcionalidade da lista de funcionalidades o usuário executará a janela ***Finalizar Funcionalidade*** representada na Figura 18. Nesta janela é apresentado um resumo com todas as informações daquela funcionalidade: data de criação, nome, descrição, prioridade, estado da funcionalidade, data da finalização e a opção "*Finalizar*", caso o usuário deseje finalizar aquela funcionalidade. Caso o usuário clique em uma funcionalidade que já foi finalizada, na lista de funcionalidades aparecerá um botão para "*Retomar*" aquela funcionalidade e

realizará a ação de alterar o estado da funcionalidade de finalizada para iniciada. A Figura 17 apresenta o resultado do controle de funcionalidades. Funcionalidades com o estado de finalizada apresentam informações adicionais tais como: finalizada por recurso humano e data de finalização. Essas informações permitem ao recurso humano que representa o papel de gerente de projeto da organização, um relatório do que está sendo feito e por quem foi feito aquela funcionalidade, portanto armazenando conhecimento sobre o controle de tarefas da organização.

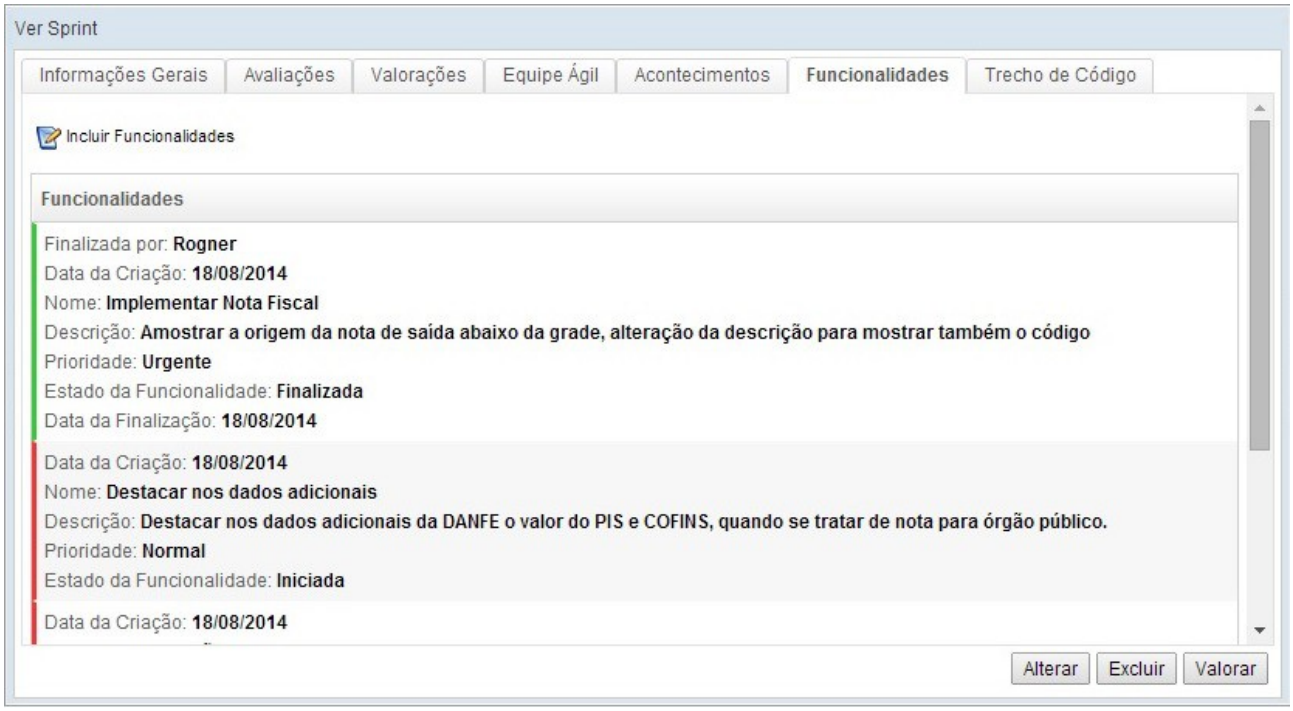

Figura 17: Janela Visualizar Item de Conhecimento Funcionalidades da Sprint

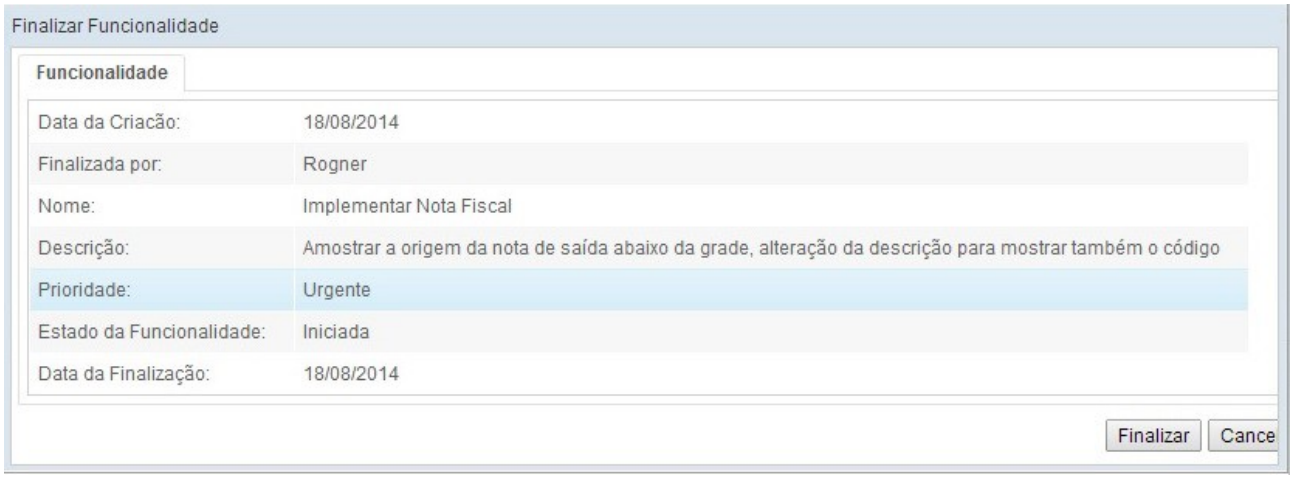

Figura 18: Janela Finalizar Funcionalidade

No menu ***Arquivos*** da tela inicial é possível criar uma equipe ágil. A Figura 19 apresenta o cadastro de uma equipe ágil**.** Os dados apresentados são: data de criação, identificador e experiência. Quando uma equipe ágil é criada ela aparece na lista de equipes da Figura 20.

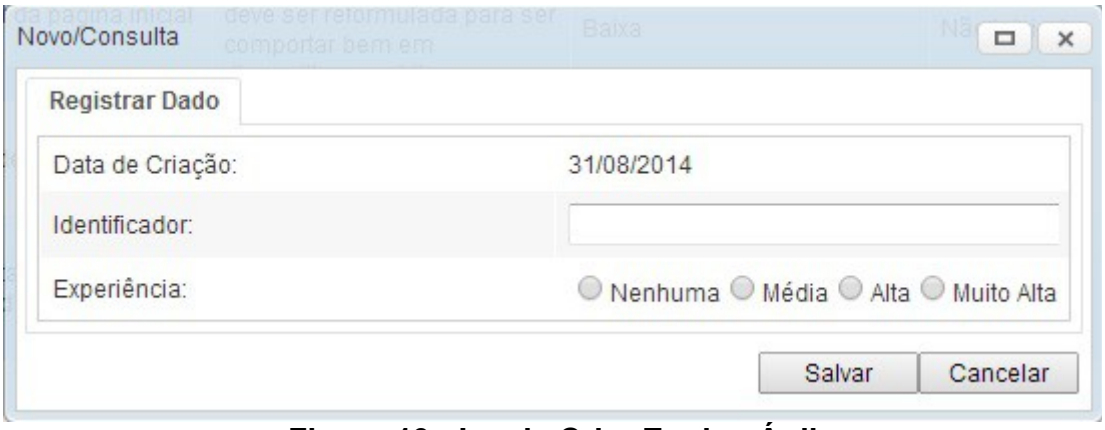

Figura 19: Janela Criar Equipe Ágil

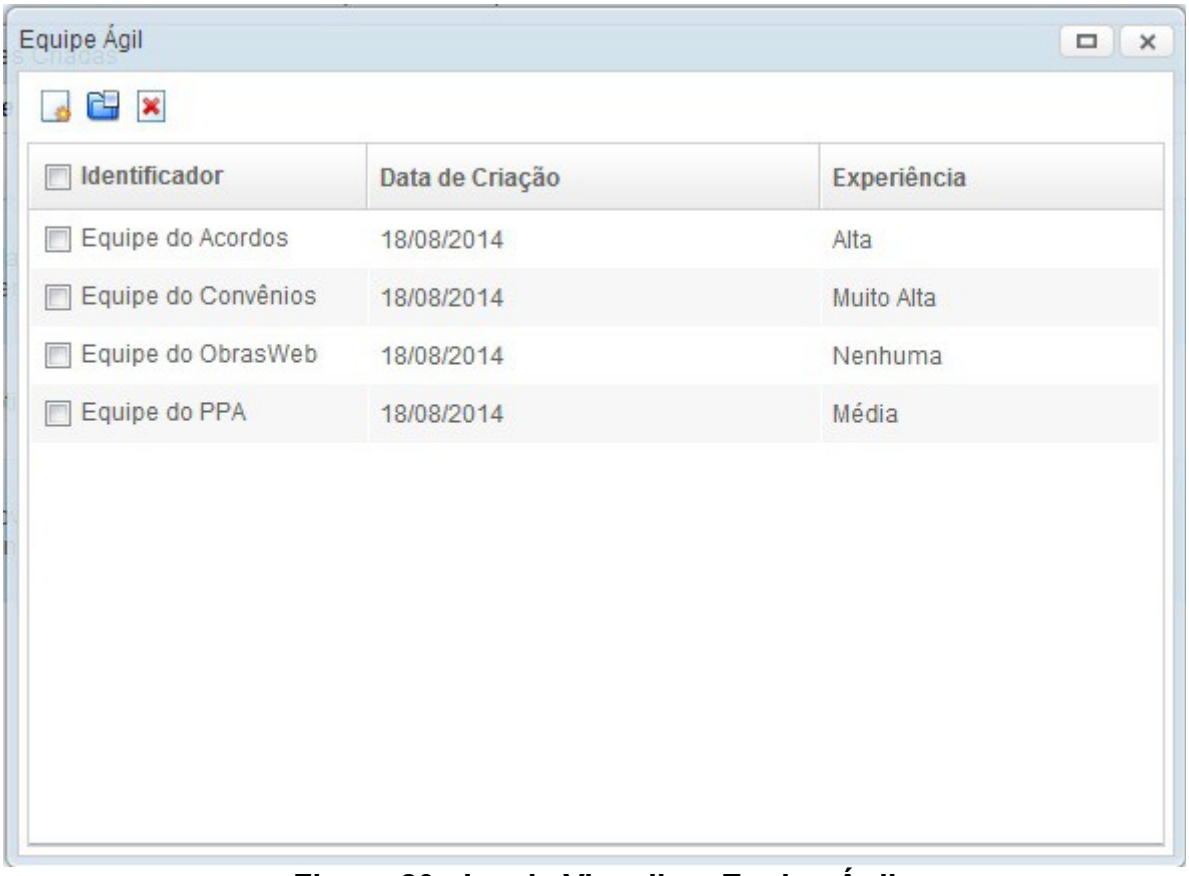

Figura 20: Janela Visualizar Equipe Ágil

A aba "*Acontecimentos*" da Figura 21 apresenta informações da lista de acontecimentos que foram incluídas na *Sprint* e a opção para incluir um novo acontecimento. Ao clicar no botão incluir acontecimento será executada a ação **Incluir Acontecimento** representado na Figura 22. Acontecimentos são informações que foram caracterizadas como importantes e devem ser registradas dentro da *Sprint*. Os dados a serem informados são: descrição, tipo, sumário e grau de dificuldade.

Quando um acontecimento é criado ele será apresentando na lista de acontecimentos na Figura 21. A lista de Acontecimentos possui uma legenda colorida na lateral. A barra de cor verde representa um acontecimento que possui uma medida cadastra. A barra de cor vermelha significa que ainda não foi cadastrada nenhuma medida para o acontecimento.

Para incluir medida para algum acontecimento ocorrido dentro da *Sprint*, o usuário deve clicar no acontecimento escolhido na lista de acontecimentos, que será executada a ação ***Incluir Medida*** apresentada na Figura 23. A janela para incluir uma medida apresenta a descrição do acontecimento escolhido pelo usuário e as informações

específicas da medida: autor, complexidade, solução adotada e complexidade.

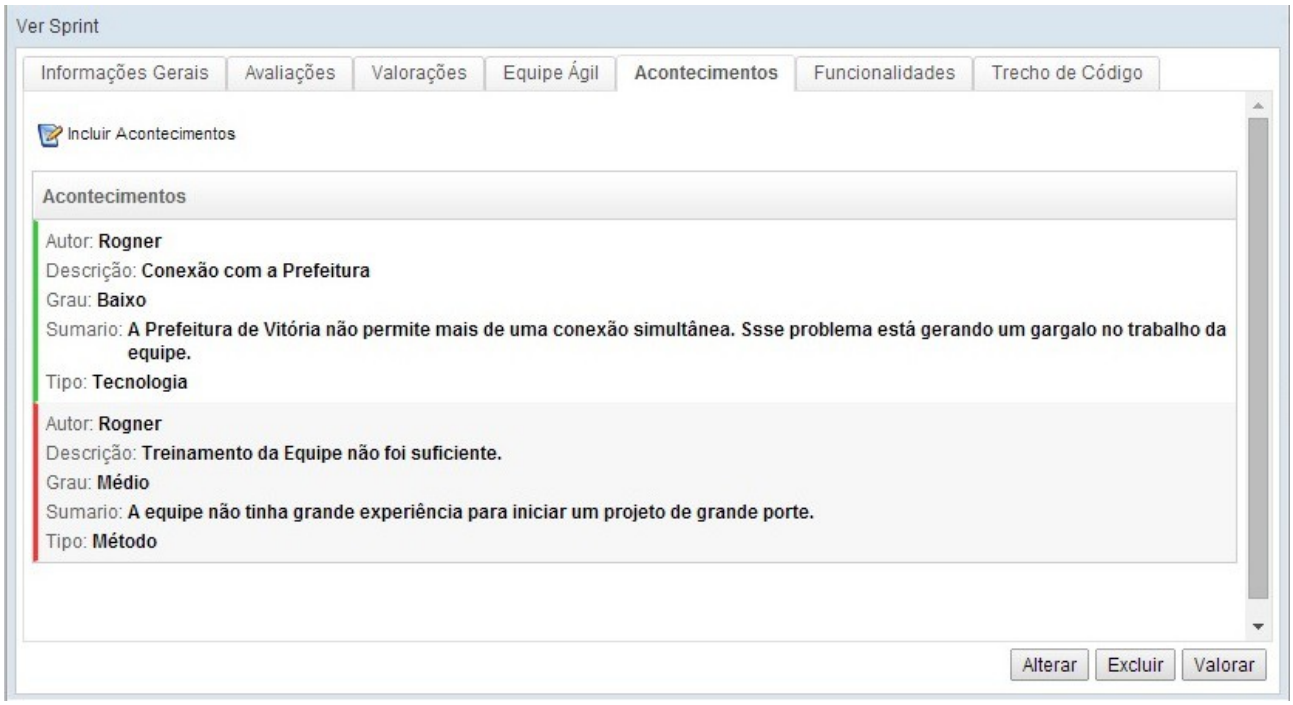

Figura 21: Janela Visualizar Acontecimentos da Sprint

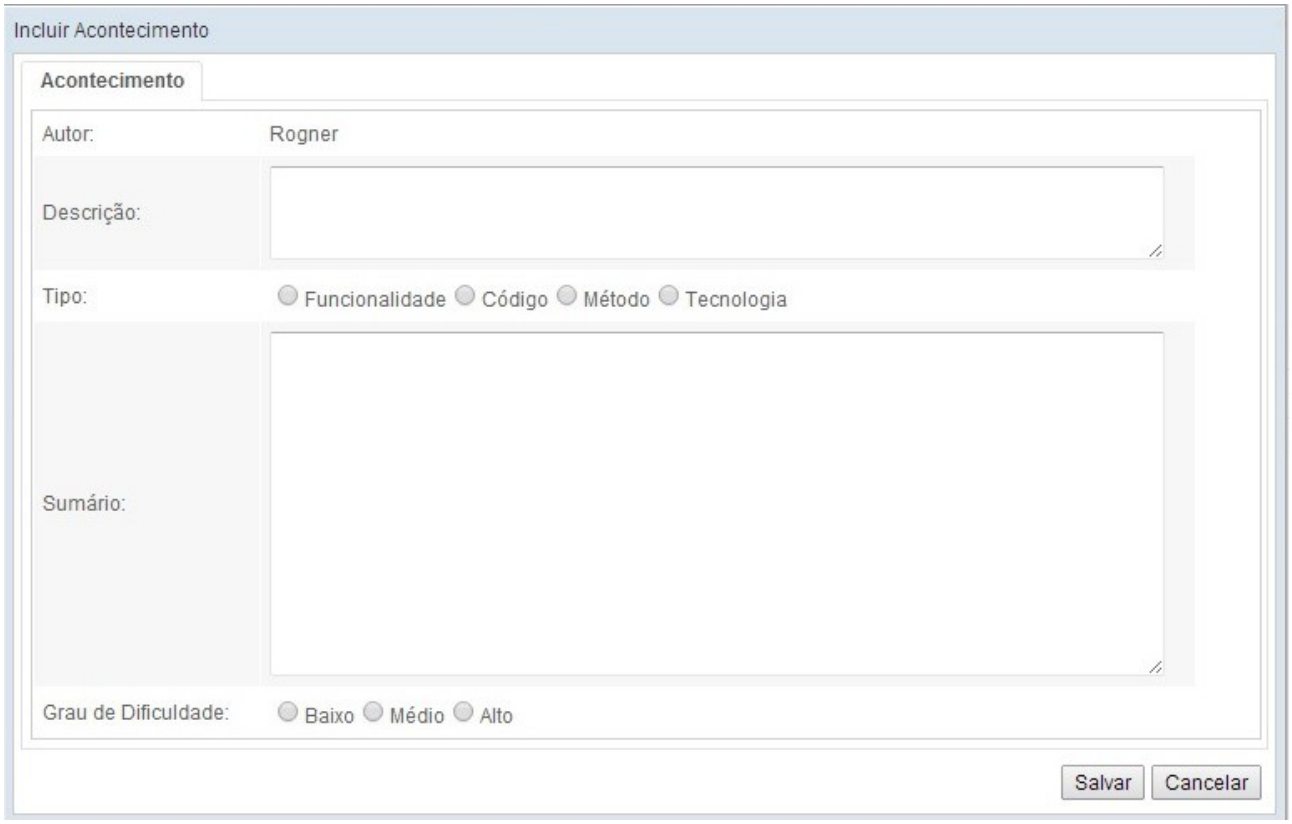

Figura 22: Janela Incluir Acontecimento na Sprint

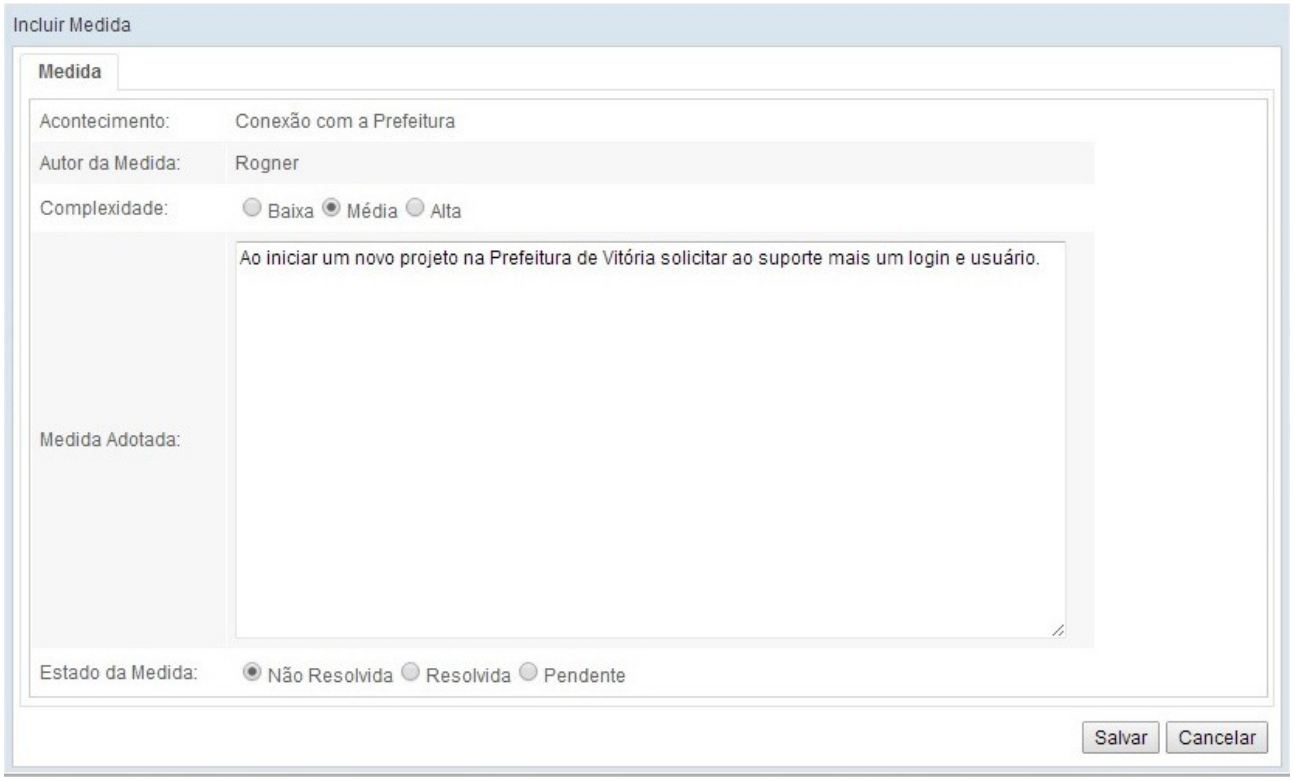

Figura 23: Janela incluir Medida para um Acontecimento

A aba "*Trecho de Código*" da Figura 24 apresenta as informações da lista de trechos de códigos mais importantes, que foram incluídas na *Sprint* e a opção para incluir um novo código. Ao clicar neste botão será executada a ação ***Incluir Trecho de Código*** representado na Figura 25. Trecho de Código é uma funcionalidade para armazenar as informações referentes aos trechos de códigos mais importantes da *Sprint*, como por exemplo: abrir uma conexão com o banco de dados, gerar um relatório, armazenar e recuperar imagens do banco, isto é, trechos que vão servir de auxílio para novos membros da organização ao realizar uma funcionalidade semelhante. Os dados a serem informados são: título e código.

Figura 24: Janela Visualizar Trecho de Código

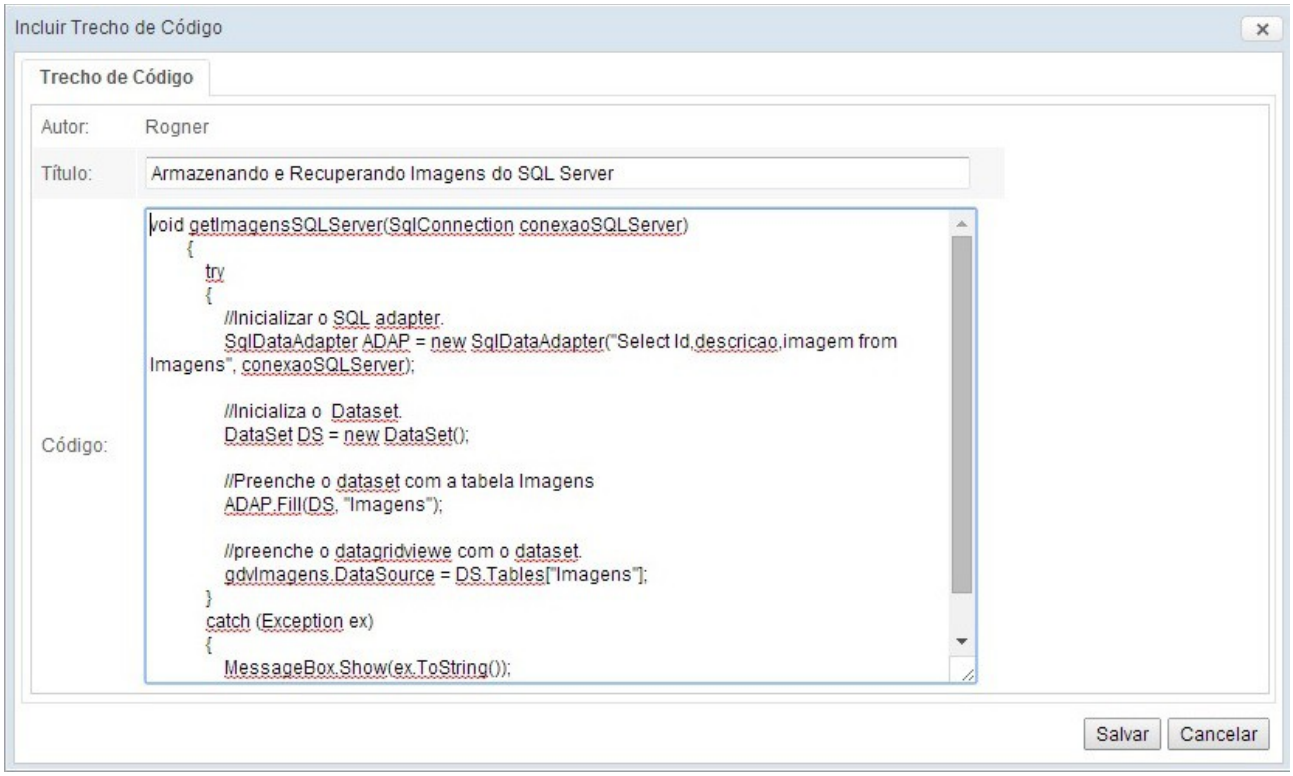

Figura 25: Janela Incluir Trecho de Código

Para auxiliar o usuário a encontrar e filtrar o conhecimento que deseja naquele

momento, os itens de conhecimentos criados são caracterizados por eventos S*crum*. Caso seja adquirido algum conhecimento sobre uma reunião de planejamento, ao criar um item de conhecimento o usuário seleciona o evento que gerou esse item de conhecimento, no caso uma reunião de planejamento. A Figura 26 apresenta a lista de eventos criado dentro do portal, os dados do evento ágil são: nome e descrição. Nesta janela, é possível alterar e excluir informações do evento ágil.

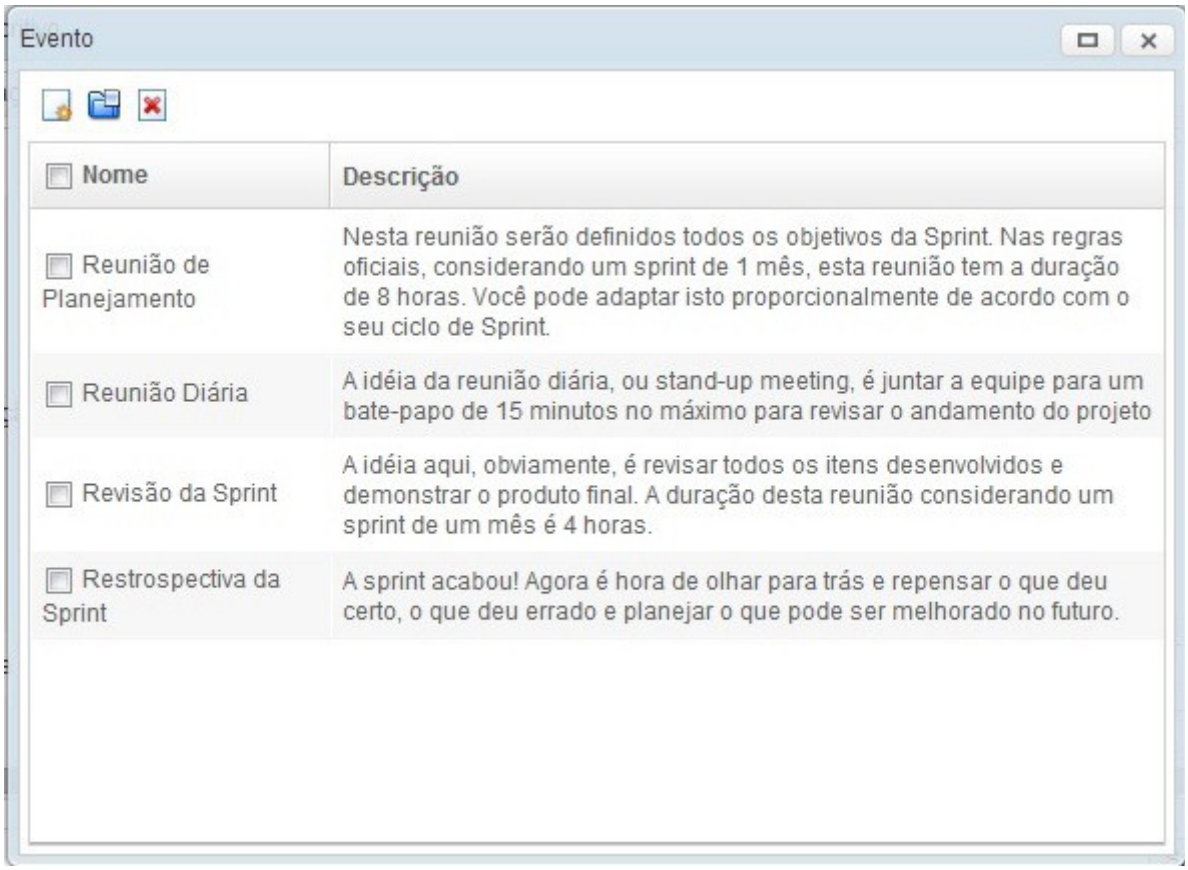

Figura 26: Janela Visualizar Eventos Scrum

A aba "*Equipe Ágil"* da Figura 27 apresenta a lista de recursos humanos membros daquela *Sprint*. Ao clicar no botão ***Incluir recurso Humano na Equipe Ágil*** será apresentada ao usuário a janela da Figura 28. Esta janela é responsável por incluir um membro na equipe ágil: deve-se informar o nome do recurso humano e salvar os dados. Caso o gerente necessite excluir algum membro da equipe, é necessário clicar sobre o recurso humano que a mensagem de exclusão será apresentada ao usuário, informando se realmente deseja excluir o membro da equipe.

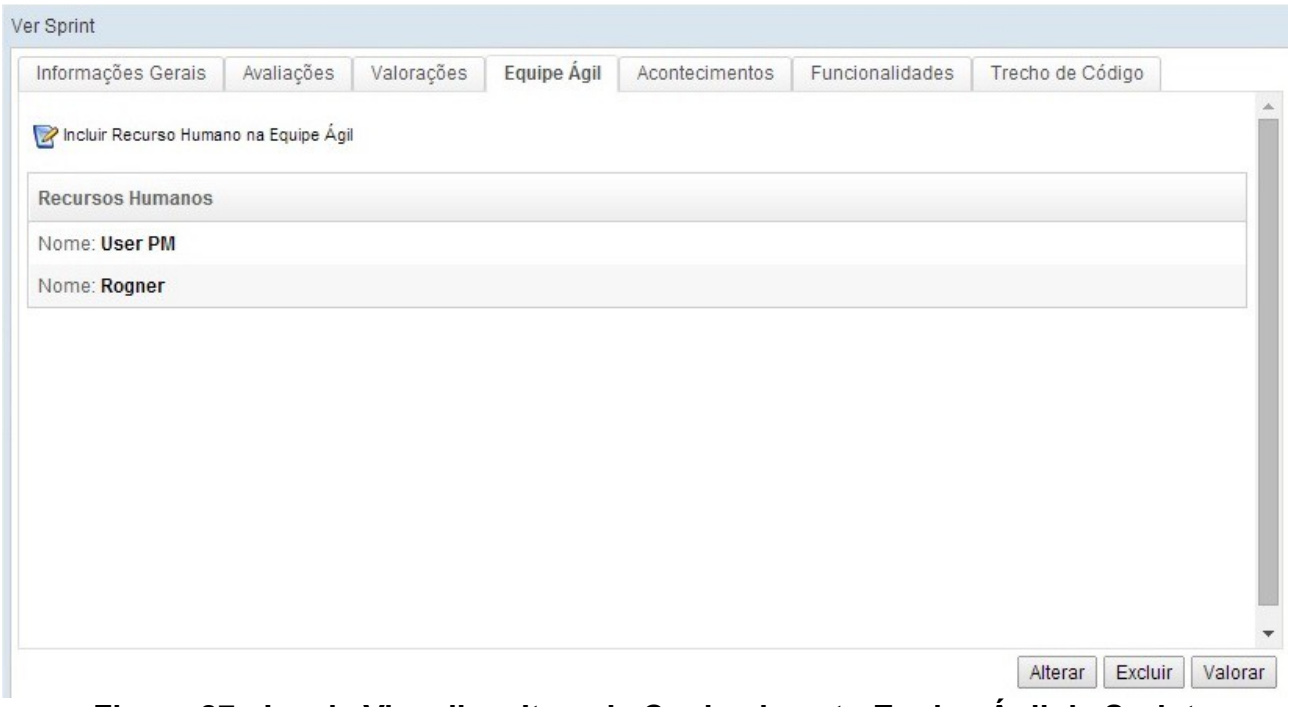

Figura 27: Janela Visualizar Item de Conhecimento Equipe Ágil da Sprint

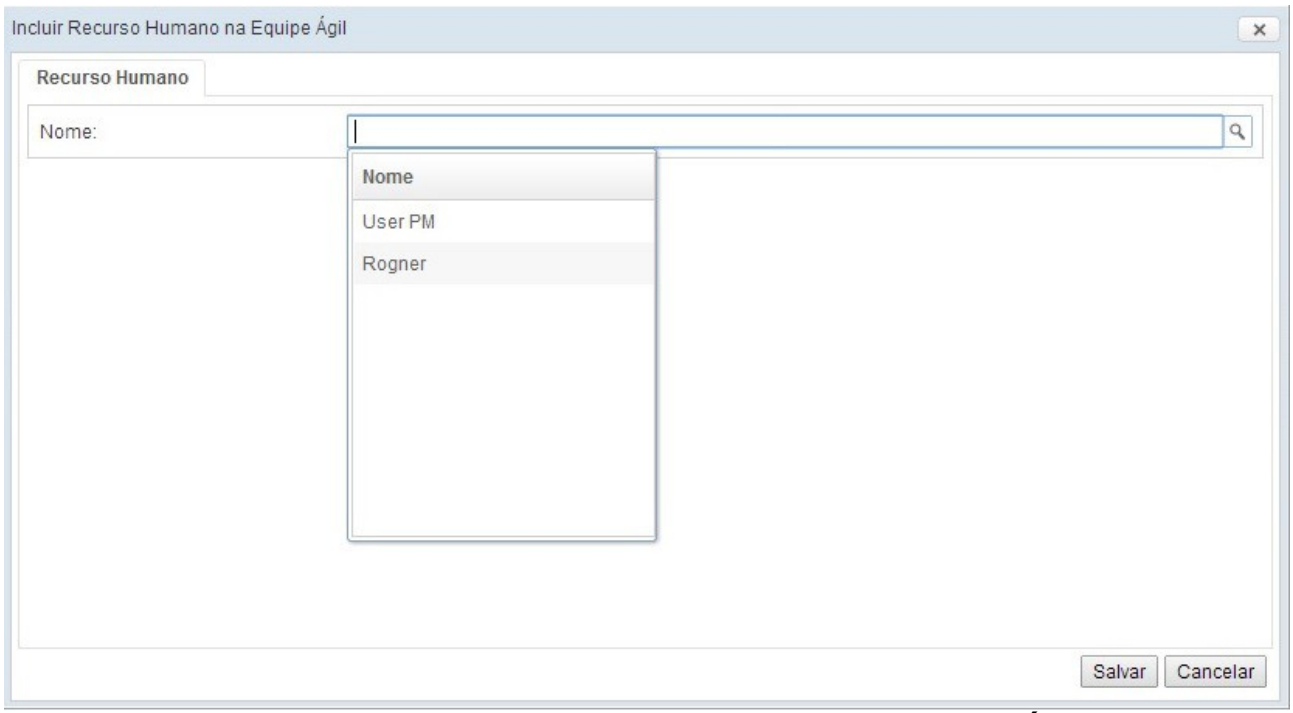

Figura 28: Janela Incluir Recurso Humano na Equipe Ágil

4 CONSIDERAÇÕES FINAIS

Este capítulo apresenta as conclusões do trabalho realizado, mostrando suas contribuições. Por fim, são apresentadas suas limitações e perspectivas de trabalhos futuros.

4.1 Conclusões

A substituição da abordagem clássica pela metodologia ágil no processo de desenvolvimento de software tem como objetivo antecipar e facilitar a visualização do produto final. A metodologia ágil proporciona essa dinâmica pois sua estrutura de iteração permite realizar entregas curtas e rápidas, pelo fato do cliente estar sempre envolvido nesse processo. Essas características permitem que os projetos adaptem-se e evoluam rapidamente em resposta às mudanças encontradas, mantendo alinhadas as informações entre o cliente e a equipe de desenvolvimento (AMBLER, 2003).

O desenvolvimento ágil propõe, na sua essência, evitar que o tempo seja gasto com tarefas excessivas de planejamento e documentação. Seu princípio parte da premissa que vale mais um software funcionando do que uma documentação extensa. Pelo lado do cliente, é mais fácil analisar o sistema através de um protótipo, do que revisar diagramas de classes, casos de uso e projetos (AMBLER, 2003). A proposta não é abandonar o processo de documentação e análise realizado na abordagem tradicional de desenvolvimento e sim deixar essas tarefas em segundo plano.

Essa carência de planejamento excessivo e documentação utilizado na metodologia tradicional poderá ser suprida e fortalecerá ainda mais o processo *Scrum* se a organização adotar a gerência de conhecimento em sua estrutura organizacional. Um sistema de gerência de conhecimento tem o objetivo de garantir que o conhecimento desejado possa estar disponível naquele exato momento em que for solicitado. Um sistema de gerência de conhecimento além de suprir essa necessidade, vai facilitar a comunicação entre os membros da unidade organizacional, uma vez que a comunicação entre os membros é uma atividade fundamental para o sucesso de projetos ágeis *Scrum*.

A adoção de um sistema de gerência de conhecimento em projetos que utilizam agilidade *Scrum* tem como potencial facilitar o compartilhamento e aplicar o reúso do conhecimento, uma vez que o *Scrum* é uma fonte capaz de gerar diversos tipos de conhecimento, seja a partir das diversas reuniões existentes tais como *Daily Scrum*

Meeting, Sprint Planning Meeting, Sprint Review e Sprint Retrospective, ou até mesmo da *Sprint,* ciclo do *Scrum* onde pode ser adquirido conhecimento através das funcionalidades implementadas, dos acontecimentos gerados, das soluções e medidas que foram utilizadas para o tratamento desse acontecimento, das lições aprendidas.

Diante do cenário apresentado e da grande quantidade de informação produzida na elaboração do *Scrum,* foi construído um portal de gerência do conhecimento um dos objetivos estabelecidos nesse trabalho. A ferramenta proporciona para organização, um ambiente onde pessoas possam adquirir conhecimento através de experiências e da resolução de vários problemas já enfrentados pela organização. Além disso o portal foi construído visando adquirir conhecimentos sobre novas tecnologias, domínios e etapas da execução dos projetos *Scrum*, procurando registrar o que deu certo.

4.2 Limitações e Perspectivas Futuras

A partir dos resultados alcançados, algumas limitações podem ser observadas, o que dá margem para a realização de trabalhos futuros, sendo assim alguns trabalhos surgirão a partir deste. Essas limitações são apresentadas nos itens abaixo.

- A criação de um portal que ofereça suporte a outro tipos de metodologias ágeis que não seja somente o *Scrum*, tais como *Extreme Programming, DAS, DSDM* e *FDD*. Cada uma dessas metodologias tem a suas particularidades e práticas sugeridas, uma política recomendada por autores é a adoção de modelos híbridos, que são mesclas dessas metodologias, onde as melhores práticas de cada metodologia deve ser aplicada ao processo.
- Realizar a integração de outras ferramentas que apoiam o desenvolvimento colaborativo à infraestrutura de gerência de conhecimento, tais como um sistema que controle a execução de tarefas e de funcionalidades pelos membros da organização.
- Acrescentar agentes de software ao portal de gerência de conhecimento aplicado ao desenvolvimento ágil, criando um portal que responda de maneira pró-ativa na identificação e disseminação dos itens de conhecimentos. Assim como disponibilizar mecanismos automatizados que podem responder dúvidas e colaborar na solução de um problema.

REFERÊNCIAS BIBLIOGRÁFICAS

ABECKER, A., MENTZAS, G., LEGAL, M., **"Business-Process Oriented Delivery of Knowledge through Domain Ontologies"**. In: Proceedings of 12th International Workshop on Database and Expert Systems Applications, pp. 442-446, Munich, Germany, Sep. 2001.

AMBLER, S. W., **Modelagem Ágil: Práticas eficazes para a Programação eXtrema e o Processo Unificado.** Porto Alegre: Bookman, 2004.

AMARAL, D.C., CONFORTO, E.C., BENASSI, J.L.C., ARAÚJO, C., **Gerenciamento Ágil de Projetos – aplicação em produtos inovadores**. Saraiva: São Paulo, 2011.

BAUER, C., KING, G., **Hibernate em Ação, Tradução 1ª Ed. Ciência Moderna**, 2005.

CAMARA, F., **Processos Ágeis e MSF.** 2005. Disponível em: <http://www.linhadecodigo.com.br/artigos.asp?id_ac=833>. Acessado em 6 outubro 2014.

BECK, K., FOWLER, M., **"Planning Extreme Programming"**, Addison- Wesley, 2000.

COELHO, A., G. N., **"Uma Infraestrutura de Gerência de Conhecimento em Organizações de Software Aplicada à Gestão de Riscos"**. Dissertação de Mestrado (Mestrado em Informática), Universidade Federal do Espírito Santo, Vitória, Agosto 2010.

COHN, M., **Agile Estimating and Planning**. New Jersey: Prentice Hall, 2005. 368p.

FAGUNDES, P. B., **Framework para Comparação e Análise de Métodos Ágeis.** Dissertação de Mestrado. Universidade Federal de Santa Catarina. Florianópolis: 2005.

FALBO, R. A., ARANTES, D. O., NATALI, A.C.C., **"Integrating Knowledge Management**

and Groupware in a Software Development Environment". Proceedings of the 5th International Conference on Practical Aspects of Knowledge Management – PAKM 2004, Karagiannis, pp. 94-105, 2004.

FALBO, R.A.; Natali, A. C. C.; Mian, P. G.; Bertollo, G.; Ruy, F. B. **"ODE: Ontology-based software Development Environment"**, In: IX CONGRESO ARGENTINO DE CIENCIAS DE LA COMPUTACIÓN, La Plata, Argentina, 2003, p. 1124-1135.

FILHO, E. G. C., PENTEADO, R., SILVA, J. C., BRAGA, R. T. C., **Padrões e Métodos Ágeis: Agilidade no Processo de Desenvolvimento de Software.** 2005.

FOWLER, M., **Patterns of Enterprise Application Architeture**. Addison-Wesley, 2003.

HIGHSMITH, J., **Agile project management: creating innovative products**. Boston: Addison-Wesley, 2004. 312 p.

LIMA, K.V.C., **Definição e Construção de Ambientes de Desenvolvimento de Software Orientados a Organização**. Tese de Doutorado, COPPE/UFRJ, Rio de Janeiro, Brasil, 2004.

NETO, O. N. S., **Análise Comparativa das Metodologias de Desenvolvimento de Softwares Tradicionais e Ágeis.** Trabalho de Conclusão de Curso. Universidade da Amazônia. Belém: 2004.

PRESSMAN, R. S., **Engenharia de Software,** McGraw-Hill, 5ª Ed., 2001

RUS, I., LINDVALL, M., "Knowledge Management in Software Engineering", IEEE Software, v. 19, n. 3, pp. 26-38, May/Jun. 2002.

OLIVEIRA, J. F., **"Abordagem para Implantação de Gerência do Conhecimento com Apoio de um Ambiente de Desenvolvimento de Software Centrado em Processos"**, Programa de Pós-Graduação em Ciência da Computação da Universidade Federal do

Pará, Dissertação, Mestrado em Ciência da Computação, 2009.

SCHWABER, K., **Agile project management with Scrum**. Washington: Microsoft, 2004.

Scrum. desenvolvimentoagil. 2014. Disponível em < http://desenvolvimentoagil.com.br/scrum/ >. Acessado em 6 outubro de 2014.

SOMMERVILLE, Ian**. Engenharia de Software**. São Paulo: Addison-Wesley, 2003.

SOUZA NETO, S., **Um modelo para gerência de conhecimento em projeto de software baseado no fluxo de conhecimento**, Rio Grande do Norte, 2006.

SOUZA NETO, V., **Aplicação de um Processo Ágil com foco em Gestão de Riscos.** Recife, 2008.

SPECIMILLE, M. S., **"TKMP: Um Portal para Gerência de Conhecimento em Teste de Software".** Monografia de Graduação (Graduação em Informática), Universidade Federal do Espírito Santo, Vitória, Fevereiro 2014.

VILLELA, K., "**Definição e Construção de Ambientes de Desenvolvimento de Software Orientados a Organização**", Tese de Doutorado, Programa de Engenharia de Sistemas e Computação, COPPE/UFRJ, 2004.

ZANATTA, A., **xScrum: Uma Proposta de Extensão do xScrum para Adequação ao CMMI**. Dissertação de Mestrado. Universidade Federal de Santa Catarina. Florianópolis: 2004.

Apêndice A

A Tabela 4 apresenta os requisitos funcionais do sistema identificados na fase de análise.

Tabela 2: Requisitos Funcionais

Identificador	Descrição	Prioridade	Depende de
RF01	O sistema deve registrar Projetos.	Média	RNF07, RNF08
RF02	O sistema deve controlar a alteração de Projetos.	Média	RF01
RF03	O sistema deve registrar Eventos *Scrum*.	Baixa	RNF07, RNF08
RF04	O sistema deve controlar a alteração de Eventos *Scrum*.	Baixa	RF03
RF05	O sistema deve registrar Equipes Ágil.	Média	RF01, RNF07, RNF08
RF06	O sistema deve controlar a alteração de Equipes Ágil.	Média	RF01, RF05
RF07	O sistema deve registrar Itens de Conhecimento de Métodos Ágeis.	Alta	RNF07, RNF08
RF08	O sistema deve registrar de Itens de Conhecimento do tipo Sprint.	Alta	RF07
RF09	O sistema deve registrar Itens de Conhecimento do tipo Descritivo.	Média	RF07
RF10	O sistema deve registrar Itens de Conhecimento do tipo Lição Aprendida.	Alta	RF07
RF11	O sistema deve registrar Itens de Conhecimento do tipo Conhecimento Relativo a uma Discussão.	Alta	RF07
RF12	O sistema deve registrar os Acontecimentos da *Sprint*.	Alta	RF07, RF08, RNF10
RF13	O sistema deve registrar a Medida adotada da *Sprint* que possui um Acontecimento.	Alta	RF07, RF08, RF12, RNF07, RNF10
RF14	O sistema deve registrar Funcionalidades para uma *Sprint*.	Alta	RF07, RF08
RF15	O sistema deve registrar o Recurso Humano que implementou a Funcionalidade.	Média	RF07, RF08, RF14
RF16	O sistema deve permitir a busca de Itens de Conhecimento.	Alta	R07
RF17	O sistema deve controlar a alteração de Itens de Conhecimento.	Alta	R07
RF18	O sistema deve permitir a avaliação de Itens de Conhecimento.	Média	R07
RF13	O sistema deve registrar os Trechos de Códigos mais importantes da *Sprint*.	Alta	RF07, RF08

A Tabela 5 apresenta os requisitos não funcionais do sistema identificados na fase de análise.

Tabela 3: Requisitos Não Funcionais

Identificador	Descrição	Categoria	Escopo	Prioridade	Depende de
RNF01	A ferramenta deve prover controle de acesso às funcionalidades.	Segurança de Acesso	Sistema	Média	
RNF02	A ferramenta deve ser de aprendizado fácil, não sendo necessário nenhum treinamento especial para seu uso.	Facilidade de Aprendizado	Sistema	Média	
RNF03	A ferramenta deve ser de fácil operação, não sendo necessário uso contínuo para uma boa operação do sistema.	Facilidade de Operação	Sistema	Alta	
RNF04	A ferramenta deve ser implementada na plataforma definida para o Projeto ODE-Web.	Manutenibilidade	Sistema	Alta	
RNF05	A persistência das informações deve ser implementada usando o Sistema Gerenciador de Bancos de Dados estabelecido na plataforma de implementação do Projeto ODE-Web. Contudo, as ferramentas devem ser projetadas para, no futuro, ser possível utilizar outra tecnologia de bancos de dados.	Portabilidade	Sistema	Alta	RNF04
RNF06	A ferramenta deve estar disponível como uma aplicação Web, acessível a partir dos principais navegadores disponíveis no mercado.	Portabilidade	Sistema	Alta	
RNF07	O sistema deve ser fácil de manter, de modo a acomodar novas funcionalidades ou até mesmo adaptação para organizações específicas.	Manutenibilidade	Sistema	Alta	
RNF08	O desenvolvimento do sistema deve explorar o potencial de reutilização de componentes, tanto no que se refere ao desenvolvimento com reúso quanto ao desenvolvimento para reúso.	Reusabilidade	Sistema	Média	
RNF10	O sistema deve ser fácil de usar, devendo-se evitar a digitação desnecessária de informações, de modo a dar agilidade ao processo.	Facilidade de Operação	Sistema	Alta	

Apêndice B

Descrição de Caso de Uso

Projeto: Portal de Gestão de Conhecimento – Métodos Ágeis.
Subsistema: agil
Caso de Uso: Visualizar Descritivo.

Descrição Sucinta: Este caso de uso permite que o desenvolvedor visualize as informações de um item de conhecimento do tipo descritivo.

Curso Normal: O desenvolvedor informa o item de conhecimento do tipo descritivo. Em seguida, a data de último acesso do item é registrada como sendo a data atual e as seguintes informações gerais do item de conhecimento são exibidas: título, autor, data de criação, resumo, aplicabilidade, avaliações, valorações, projetos relacionados, eventos relacionados e outros itens de conhecimento relacionados. Em seguida, as seguintes informações específicas do item de conhecimento do tipo descritivo são exibidas: nome, descrição, url da página e artigo de referência.

Classes Relacionadas: ItemConhecimento, Descritivo, RecursoHumano, Avaliacao, Valoracao, Projeto e Evento.

Descrição de Caso de Uso

Projeto: Portal de Gestão de Conhecimento – Métodos Ágeis.
Subsistema: agil
Caso de Uso: Visualizar Sprint.

Descrição Sucinta: Este caso de uso permite que o desenvolvedor visualize as informações de um item de conhecimento do tipo *Sprint*.

Curso Normal: O desenvolvedor informa o item de conhecimento do tipo *Sprint*. Em seguida, a data de último acesso do item é registrada como sendo a data atual e as seguintes informações gerais do item de conhecimento são exibidas: título, autor, data de criação, resumo, aplicabilidade, avaliações, valorações, projetos relacionados, eventos relacionados e outros itens de conhecimento relacionados. Em seguida as seguintes informações específicas do item de conhecimento do tipo *Sprint* são exibidas: quantidade, duração, data de início, dificuldade, tipo de *Sprint*, equipe, listagem de funcionalidades, listagem de acontecimentos, medidas e trechos de códigos.

Classes Relacionadas: ItemConhecimento, Sprint, Funcionalidade, EquipeAgil, Acontecimento, Avaliacao, Valoracao, Projeto, TrechoCodigo, Medida e Recurso Humano.

Descrição de Caso de Uso

Projeto: Portal de Gestão de Conhecimento – Métodos Ágeis.
Subsistema: agil
Caso de Uso: Buscar Descritivo.

Descrição Sucinta: Este caso de uso permite que o desenvolvedor busque itens de conhecimento do tipo descritivo de acordo com parâmetros informados.

Curso Normal:

O desenvolvedor informa os dados que devem ter os itens de conhecimento disponíveis (com estado *Disponível*) que deseja recuperar. Esses dados podem ser: expressão contida no título ou resumo do item de conhecimento, data inicial e final de criação, última valoração e último acesso, quantidade mínima e máxima de acessos e de valorações, percentual mínimo e máximo de valorações positivas e negativas. Para buscar item de conhecimento do tipo descritivo, deve-se selecionar o item de conhecimento do tipo descritivo. Em seguida, os itens contendo os dados informados são recuperados e apresentados.

item apresentado: ponto de inclusão

- Caso o desenvolvedor decida visualizar as informações de um item de conhecimento do tipo descritivo, o caso de uso *Visualizar Item Conhecimento* é realizado.

Classes Relacionadas: ItemConhecimento, RecursoHumano, Valoracao, Projeto e Evento.

Descrição de Caso de Uso

Projeto: Portal de Gestão de Conhecimento – Métodos Ágeis.
Subsistema: agil
Caso de Uso: Buscar Sprint.

Descrição Sucinta: Este caso de uso permite que o desenvolvedor busque itens de conhecimento do tipo *Sprint* de acordo com parâmetros informados.

Curso Normal:

O desenvolvedor informa os dados que devem ter os itens de conhecimento disponíveis (com estado *Disponível*) que deseja recuperar. Esses dados podem ser: expressão contida no título ou resumo do item de conhecimento, data inicial e final de criação, última valoração e último acesso, quantidade mínima e máxima de acessos e de valorações, percentual mínimo e máximo de valorações positivas e negativas. Para buscar item de conhecimento do tipo *Sprint*, deve-se selecionar o item de conhecimento do tipo *Sprint*. Em seguida, os itens contendo os dados informados são recuperados e apresentados.

item apresentado: ponto de inclusão

- Caso o desenvolvedor decida visualizar as informações de um item de conhecimento do tipo *Sprint*, o caso de uso *Visualizar Item Conhecimento* é realizado.

Classes Relacionadas: ItemConhecimento, RecursoHumano, Valoracao, Projeto e Evento.

Descrição de Caso de Uso

Projeto: Portal de Gestão de Conhecimento – Métodos Ágeis.
Subsistema: agil
Caso de Uso: Cadastrar Sprint.

Descrição Sucinta: Este caso de uso permite que o desenvolvedor crie as informações de um item de conhecimento do tipo *Sprint*.

Curso Normal: O desenvolvedor informa o item de conhecimento do tipo *Sprint*. Em seguida o desenvolvedor preenche a quantidade, duração, a data de início, dificuldade, tipo da *Sprint* (*funcional ou teste*), objetivos e se a *Sprint* foi revisada. Após preencher as informações o item é salvo e uma mensagem informando que o item foi salvo com sucesso é apresentada ao usuário.

item apresentado: ponto de inclusão

- Caso o desenvolvedor queira incluir funcionalidades para a *Sprint* o caso de uso *Incluir Funcionalidade* é realizado.
- Caso o desenvolvedor queira incluir equipe para a *Sprint* o caso de uso *Incluir Equipe* é realizado.
- Caso o desenvolvedor queira incluir acontecimentos para a *Sprint* o caso de uso *Cadastrar Acontecimento* é realizado.
- Caso o desenvolvedor queira incluir trecho de código mais importantes para a *Sprint* o caso de uso *Cadastrar Trecho de Código* é realizado.

Curso Alternativo: Os dados informados são inválidos: Uma mensagem de erro é exibida, solicitando a correção dos mesmos.

Classes Relacionadas: ItemConhecimento, Sprint, Funcionalidade, EquipeAgil, Acontecimento, TrechoCodigo, Medida e Recurso Humano.

Descrição de Caso de Uso

Projeto: Portal de Gestão de Conhecimento – Métodos Ágeis.
Subsistema: agil
Caso de Uso: Cadastrar Descritivo.

Descrição Sucinta: Este caso de uso permite que o desenvolvedor crie as informações de um item de conhecimento do tipo Descritivo.

Curso Normal: O desenvolvedor informa o item de conhecimento do tipo Descritivo. Em seguida o desenvolvedor preenche o nome do método que deseja guardar, a descrição do método, url da página e artigo de referência. Após preencher as informações o item é salvo e uma mensagem informando que o item foi salvo com sucesso é apresentada ao usuário.

Curso Alternativo: Os dados informados são inválidos: Uma mensagem de erro é exibida, solicitando a correção dos mesmos.

Classes Relacionadas: ItemConhecimento, Descritivo.

Descrição de Caso de Uso

Projeto: Portal de Gestão de Conhecimento – Métodos Ágeis.
Subsistema: agil
Caso de Uso: Cadastrar Acontecimento.

Descrição Sucinta: Este caso de uso permite que o desenvolvedor crie acontecimentos dentro do item de conhecimento referente a uma *Sprint*.

Curso Normal: Durante a criação da *Sprint* o desenvolvedor pode incluir algum acontecimento que ocorreu dentro da *Sprint*. O desenvolvedor deve informar o sumário, descrição, tipo e grau de dificuldade do acontecimento.

item apresentado: ponto de inclusão

- Caso o desenvolvedor queira incluir uma medida para o acontecimento o caso de uso *Incluir Medida* é realizado.

Curso Alternativo:
Os dados informados são inválidos: Uma mensagem de erro é exibida, solicitando a correção dos mesmos.

Classes Relacionadas: ItemConhecimento, Sprint, RecursoHumano e Acontecimento.

Descrição de Caso de Uso

Projeto: Portal de Gestão de Conhecimento – Métodos Ágeis.
Subsistema: agil
Caso de Uso: Cadastrar Medida.

Descrição Sucinta: Este caso de uso permite que o desenvolvedor crie uma medida dentro do item de conhecimento referente a uma *Sprint* de um acontecimento que ocorreu.

Curso Normal: Durante a criação da *Sprint* o desenvolvedor pode incluir algum acontecimento que ocorreu dentro da *Sprint*. Caso o desenvolvedor cadastre algum acontecimento para o ocorrido, o desenvolvedor pode também criar uma medida para esse acontecimento. O desenvolvedor deve informar: a medida adotada, a complexidade e o estado da medida (*Não Resolvida, Resolvida ou Pendente*).

Curso Alternativo:
Os dados informados são inválidos: Uma mensagem de erro é exibida, solicitando a correção dos mesmos.

Classes Relacionadas: ItemConhecimento, Sprint, RecursoHumano, Medida e Acontecimento.

Descrição de Caso de Uso

Projeto: Portal de Gestão de Conhecimento – Métodos Ágeis.
Subsistema: agil
Caso de Uso: Incluir Funcionalidade.

Descrição Sucinta: Este caso de uso permite que o desenvolvedor inclua funcionalidades dentro do item de conhecimento referente a uma *Sprint*.

Curso Normal: Durante a criação da *Sprint* o desenvolvedor pode incluir funcionalidades que podem ser desenvolvidas dentro da *Sprint*. O desenvolvedor deve informar o nome, descrição, prioridade e estado da funcionalidade. Quando a funcionalidade for marcada como resolvida, deve-se incluir o recurso humano que a desenvolveu.

Curso Alternativo:
Os dados informados são inválidos: Uma mensagem de erro é exibida, solicitando a correção dos mesmos.

Classes Relacionadas: Sprint e RecursoHumano.

Descrição de Caso de Uso

Projeto: Portal de Gestão de Conhecimento – Métodos Ágeis.
Subsistema: agil
Caso de Uso: Incluir Equipe Ágil.

Descrição Sucinta: Este caso de uso permite que o desenvolvedor inclua uma equipe dentro do item de conhecimento referente a uma *Sprint*.

Curso Normal: Durante a criação da *Sprint* o desenvolvedor pode incluir uma equipe para a *Sprint*. O desenvolvedor deve selecionar a equipe que vai pertencer à *Sprint*.

Classes Relacionadas: Sprint e EquipeAgil.

Descrição de Caso de Uso

Projeto: Portal de Gestão de Conhecimento – Métodos Ágeis.
Subsistema: agil
Caso de Uso: Cadastrar Trecho de Código.

Descrição Sucinta: Este caso de uso permite que o desenvolvedor crie trechos de códigos mais importantes dentro do item de conhecimento referente a uma *Sprint*.

Curso Normal: Durante a criação da *Sprint* o desenvolvedor pode incluir trechos de códigos mais importantes que ocorreram dentro da *Sprint*. O desenvolvedor deve informar o título e o trecho do código.

Curso Alternativo:
Os dados informados são inválidos: Uma mensagem de erro é exibida, solicitando a correção dos mesmos.

Classes Relacionadas: ItemConhecimento, Sprint, RecursoHumano e TrechoCodigo.