Uma Apoio Sistematizado à Implementação do Processo de Desenvolvimento de Requisitos do MPS.BR e CMMI a partir do Uso de Ferramentas de Software Livre

Ewelton Y. C. Yoshidome[1], Maurício R. de A. Souza[1], Wallace M. P. Lira[1], Sandro R. B. Oliveira[1], Alexandre M. L. de Vasconcelos[2].

[1] Programa de Pós-Graduação em Ciência da Computação – Instituto de Ciências Exatas e Naturais – Universidade Federal do Pará (UFPA)
Belém – PA – Brasil.
[2]Programa de Pós-Graduação em Ciência da Computação – Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Recife –PE – Brasil
ewelton.yoshio@gmail.com, mauricio.ronny@uol.com.br, wallace.lira@gmail.com, srbo@ufpa.br, amlv@cin.ufpe.br

Abstract. To meet the market demands and ensure competitiveness, the search for quality has become a strategic goal for many companies, so the search for quality models becomes necessity. In this context, this paper proposes a tool support for implementation of Requirements Development process included in MPS.BR and CMMI programs, exclusively using free software tools, analyzing adherence to the maturity programs and trying to reduce the time and cost of its implementation.

Keywords: Software Quality, MPS.BR, CMMI, Requirements Development, Free Software Tools.

Resumo. Para atender as demandas de mercado e garantir competitividade, a busca por qualidade tem se tornado objetivo estratégico de muitas empresas, logo, a procura por modelos de qualidade torna-se necessária. Neste contexto, este trabalho apresenta uma proposta de apoio ferramental para a implementação do processo de Desenvolvimento de Requisitos constante nos programas MPS.BR e CMMI utilizando exclusivamente ferramentas de software livre, analisando a aderência aos programas de maturidade e procurando diminuir o tempo e custo de implementação do mesmo.

Palavras-Chave: Qualidade de Software, MPS.BR, CMMI, Desenvolvimento de Requisitos, Ferramentas de Software Livre.

1 Introdução

As organizações enfrentam constantes mudanças motivadas por diversos fatores. A alta concorrência exige habilidades para fornecer produtos e serviços mais inovadores. As exigências dos clientes são outros fatores causadores de mudanças, pois as organizações devem se preocupar em criar produtos e prover serviços que

satisfaçam as necessidades dos seus clientes para garantir confiança e satisfação [2]. Neste contexto, foi desenvolvido pela SOFTEX (Associação para Promoção da Excelência do Software Brasileiro) o programa MPS.BR (Melhoria do Processo de Software Brasileiro) que tem como foco principal a implementação da melhoria dos processos de software em empresas brasileiras. Este programa divulga o modelo de referência MR-MPS [2].

Entre os processos presentes no MR-MPS está o processo de Desenvolvimento de Requisitos (DRE), que tem como propósito definir os requisitos do cliente, do produto e dos componentes do produto [2]. Essas características inerentes ao processo de Desenvolvimento de Requisitos são melhores gerenciadas quando se sistematiza o processo a partir da utilização de ferramentas, pois esta prática pode implicar na redução de esforço e tempo, devido a diminuição da necessidade de documentação, agilizando o processo como um todo.

Neste contexto, este trabalho está inserido no escopo do Projeto SPIDER (*Software Process Improvement – DEvelopment and Research*), disponível em: *www.spider.ufpa.br*; o qual tem como objetivo criar um *Suite* de ferramentas de software livre aderente ao MR-MPS e ao CMMI-DEV (*Capability Maturity Model Intergration for Development*) [11]. Software livre é um software o qual pode ser executado, copiado, distribuído, estudado, modificado ou aperfeiçoado, de maneira comercial ou gratuita (Fonte: *www.softwarelivre.org*).

Este trabalho visa descrever uma metodologia de apoio ao processo de Desenvolvimento de Requisitos (DRE) com o uso de quatro ferramentas livres e uma gratuita. Neste trabalho entende-se metodologia como sendo a codificação de um conjunto de práticas recomendadas (por exemplo, os serviços/operações providos a partir do uso de ferramentas), às vezes acompanhada de material de treinamento, programas de educação formal, planilhas, diagramas, tomando como parte um método (processo com uma série de passos, para construir um software) [1]. Esta metodologia procura agregar boas práticas para o uso de ferramentas livres de apoio ao processo de Desenvolvimento de Requisitos.

Além desta seção introdutória, a Seção 2 apresenta a fundamentação teórica do processo de Desenvolvimento de Requisitos no contexto dos modelos MR-MPS e CMMI-DEV. A Seção 3 descreve alguns trabalhos relacionados encontrados durante a pesquisa, em seguida, na Seção 4 é realizada uma descrição das ferramentas utilizadas para a implementação da metodologia proposta. Na Seção 5 é apresentada a metodologia de uso das ferramentas, seguida com uma análise susinta de aderência ao MR-MPS e ao CMMI-DEV na Seção 6 e, finalmente, a Seção 7 apresenta as considerações finais deste trabalho.

2 Processo de Desenvolvimento de Requisitos

A Engenharia de Requisitos tem a função de auxiliar o entendimento do sistema a ser desenvolvido, incluindo tarefas para analisar as necessidades do cliente e a forma de como o sistema irá interagir com os usuários finais [1]. A Engenharia de Requisitos pode ser dividida em duas atividades [2][4]: Desenvolvimento de Requisitos e Gerência de Requisitos.

Segundo [2] e [4], o objetivo do processo de Desenvolvimento de Requisitos é interpretar as necessidades do cliente para a criação dos requisitos do sistema a ser

desenvolvido. No CMMI-DEV, a área de processo *Requirements Development* (RD) está estruturada em dez práticas específicas (SP – *Specific Practices*, "é a descrição de uma atividade considerada importante para alcançar uma meta específica associada" [4]), que são:

- **SP1.1:** *Elicit Needs*;
- **SP1.2:** *Transform Stakeholder Needs into Customer Requirements*;
- **SP2.1:** *Establish Product and Product Component Requirements*;
- **SP2.2:** *Allocate Product Component Requirements*;
- **SP2.3:** *Identify Interface Requirements*;
- **SP3.1:** *Establish Operational Concepts and Scenarios*;
- **SP3.2:** *Establish a Definition of Required Functionality and Quality Attributes*;
- **SP3.3:** *Analyze Requirements*;
- **SP3.4:** *Analyze Requirements to Achieve Balance*;
- **SP3.5:** *Validate Requirements*.

No contexto do MR-MPS, o processo de Desenvolvimento de Requisitos (DRE) está estruturado em oito resultados esperados aderentes às práticas específicas do CMMI-DEV [2]. Um resultado esperado "é um resultado observável do sucesso do alcance do propósito do processo" [8]. Em seguida, são listados os resultados esperados do processo de Desenvolvimento de Requisitos:

- **DRE1:** As necessidades, expectativas e restrições do cliente, tanto do produto quanto de suas interfaces são identificadas;
- **DRE2:** Um conjunto definido de requisitos do cliente é especificado a partir das necessidades, expectativas e restrições identificadas;
- **DRE3:** Um conjunto definido de requisitos funcionais e não-funcionais, do produto e dos componentes do produto que descrevem a solução do problema a ser resolvido é definido e mantido a partir dos requisitos do cliente;
- **DRE4:** Os requisitos funcionais e não-funcionais de cada componente do produto são refinados, elaborados e alocados;
- **DRE5:** Interfaces internas e externas do produto e de cada componente do produto são definidas;
- **DRE6:** Conceitos operacionais e cenários são desenvolvidos;
- **DRE7:** Os requisitos são analisados, usando critérios definidos, para balancear as necessidades dos interessados com as restrições existentes;
- **DRE8:** Os requisitos são validados.

Segundo [2] e [4], Gerência de Requisito é o processo responsável por gerenciar todos os requisitos do projeto e identificar as suas inconsistências com os demais produtos de trabalho. Entretanto, o processo de Gerência de Requisitos está fora do escopo desta pesquisa, focando-se apenas no processo de Desenvolvimento de Requisitos. Existe um trabalho publicado no Workshop de Engenharia de Requisitos de 2010, apresentado pelo mesmo grupo do projeto SPIDER, que descreve uma metodologia com apoio ferramental para atendimento às recomendações do processo de Gerência de Requisitos [7].

3 Trabalhos Relacionados

Este trabalho foca-se em propor uma metodologia para apoiar o Processo de Desenvolvimento de Requisitos e atua como uma extensão à metodologia proposta em [7], no qual seu trabalho descreve uma metodologia de uso de ferramentas livres para o apoio do Processo de Gerência de Requisitos do MPS.BR. Pretende-se, a partir desta extensão, utilizar um *kit* de ferramentas em comum que sejam capazes de contemplar os processos de gerência e desenvolvimento de requisitos.

Em [15] é descrito um processo de levantamento e modelagem de requisitos para sistemas web com o uso do método Web-SEMP (*Web System Elicitation, Modeling and Planning*). Porém, pelo fato de seu trabalho não ser direcionado à adequação a modelos de qualidade de processo de software (CMMI-DEV, MR-MPS), nota-se a ausência de algumas práticas exigidas pelo modelos, tais como o uso de critérios objetivos para verificação e validação de requisitos e a definição de interfaces internas e externas.

Martins *et al.* em [16] descreve uma ferramenta de apoio à Engenharia de Requisitos integrada ao ambiente ODE (*Ontology-based Development Enviroment*). Porém, este trabalho foca-se em apresentar funcionalidades voltadas à gerência e desenvolvimento de requisitos, não apresentando um alinhamento aos modelos de qualidade existentes no mercado. Esse fato pode ser observado pela ausência da funcionalidade/serviço de criação e aplicação de critérios objetivos para realizar o balanceamento entre as necessidades dos interessados às restrições existentes no projeto, e os critérios para executar a validação dos requisitos, junto ao cliente ou usuário final.

Adicionalmente, encontra-se na literatura especializada inúmeros trabalhos que tratam de ferramentas de apoio à gestão e engenharia de requisitos, como [17] e [18]. Entretanto, percebe-se dois aspectos negativos em relação aos trabalhos propostos: primeiro, cita-se o fato destes trabalhos referenciarem ferramentas que não estão disponíveis para uso/operação, o que não apóia a implementação de melhoria efetivamente; segundo, estes trabalhos não apresentam metodologias para apoiar a operação das ferramentas quando da implementação das boas práticas constantes nos modelos de qualidade, o que pode acarretar no seu desuso quando considerado o contexto dos programas de melhoria dos processos organizacionais.

4 Ferramentas de Apoio

Para a implementação da metodologia do processo de Desenvolvimento de Requisitos, foram utilizadas as ferramentas Openproj (apoio à Gerência de Projetos), OSRMT (apoio à Gerência de Requisitos), Redmine (apoio ao Controle de Mudanças), Astah Community (apoio à modelagem UML - *Unified Modeling Language*) e Spider-CL (apoio a Geração de *Checklists*). Vale salientar que o uso destas ferramentas só fará sentido se o fluxo da metodologia for seguido. As subseções a seguir descrevem com mais detalhes cada uma das ferramentas.

4.1 OSRMT

O OSRMT (*Open Source Requirements Management Tool*) é uma ferramenta, desenvolvida na linguagem Java, projetada para apoiar o processo de Gerência de Requisitos. Licenciada sob os termos da GPL (*General Public License*), hoje possui a versão 1.5 como sua versão mais estável (atualmente no "patch 2") [9].

A ferramenta possui suporte ao processo de Desenvolvimento de Requisitos do MR-MPS no que se refere ao armazenamento e registro dos requisitos gerados durante a fase de elicitação.

4.2 Openproj

O Openproj [10] é uma ferramenta *desktop*, livre e *open source*, voltada para apoiar a Gerência de Projetos. Esta ferramenta é composta por um grande número de funcionalidades que dão suporte ao cronograma, gestão de recursos humanos e riscos. Outra característica presente na ferramenta é a possibilidade de gerar representações gráficas como WBS (*Work Breakdown Structure*), RBS (*Resource Breakdown Structure*), CPM (*Critcal Path Method*) e *Gantt Chart*.

A versão utilizada na metodologia foi modificada pelo projeto SPIDER com a adição de funcionalidades de suporte a estimativas, riscos e definição do escopo do projeto. O objetivo das alteraçãoes foi permitir que a ferramenta possua uma maior aderência ao MR-MPS. A versão customizada encontra-se disponível em *http://www.spider.ufpa.br/index.php?id=resultados*.

O propósito do uso da ferramenta Openproj na metodologia é realizar o registro e armazenamento do método de elicitação de requisitos.

4.3 Redmine

Redmine [11] é uma ferramenta *web* para *bugtracking*, desenvolvida em Ruby, com o objetivo de gerenciar mudanças nos produtos de trabalho de um projeto. Além de realizar a gestão de mudanças, a ferramenta Redmine proporciona suporte à Gerência de Projetos [12].

Para o contexto da metodologia, a ferramenta de controle de mudanças é utilizada para o controle do ciclo de vida das tarefas de verificação, validação e implementação dos requisitos.

4.4 Spider-CL

A Spider-CL [13] é uma ferramenta desenvolvida no projeto SPIDER, com o propósito de criar *checklists* compostos por critérios objetivos para utilização em diversos contextos, provendo mecanismos para a aplicação destes *checklists*, mantendo histórico e registrando seus resultados. *Checklist* é um conjunto de atributos que servem para avaliar determinado produto de trabalho. Cada atributo possui uma lista de possíveis alternativas das quais apenas uma pode ser escolhida.

No contexto da metodologia proposta, os *checklists* gerados pela ferramenta Spider-CL são utilizados para a definição de critérios objetivos necessários para a implementação de alguns resultados esperados.

3.5 Astah Community

Astah Community [14] é uma ferramenta gratuita, mas não *open source*, voltada para a modelagem de diagramas UML – *Unified Modeling Langua*ge. Além do Astah Community, existem outras três versões: Astah UML, Astah Professional e Astah Share, que disponibilizam outras funcionalidades além da modelagem UML, porém, sua licença é comercial.

Na metodologia, a ferramenta Astah Community é utilizada para o desenvolvimento dos diagramas necessários para representar o projeto dos requisitos.

5 Metodologia Proposta

Esta seção descreve a metodologia para implementar o processo de Desenvolvimento de Requisitos com o uso das ferramentas propostas. Esta proposta pretende estar aderente aos modelos constantes nos programas MPS.BR e CMMI.

Salienta-se que para a implementação de um programa de melhoria organizacional, são requeridas ações que tratem de maturidade e capacidade. Para as ações referentes à maturidade, pode-se citar a definição de um processo, institucionalização de uma política, entre outros. Para realizar a evidência de ações que tratem da capacidade, pode-se verificar a aplicação prática do processo definido na organização, treinamento/capacitação nos ativos constantes no processo, entre outros. Uma dessas boas práticas recomenda o uso de ferramentas de software para sistematizar a execução de atividades constantes no processo.

Não é foco deste trabalho definir um processo para a implementação dos resultados esperados/práticas específicas do processo de Desenvolvimento de Requisitos, mas sim propor um conjunto de ferramental de apoio para auxiliar na implementação das boas práticas constantes nos modelos. Assim, qualquer análise de aderência descrita na metodologia de uso proposta neste trabalho recai na análise do atendimento dos serviços descritos nas ferramentas discutidas de acordo com as características recomendadas pelo processo de Desenvolvimento de Requisitos.

Na Figura 1 pode ser visualizado o fluxo das atividades da metodologia (definido usando o padrão BPMN), junto com o nome das ferramentas as quais serão utilizadas para executar cada atividade constante na metodologia. Nas subseções seguintes serão descritas a metodologia proposta. Para facilitar o entendimento da metodologia, as atividades referentes ao processo de Desenvolvimento de Requisitos foram divididas em três, a saber: Desenvolver Requisitos do Cliente; Desenvolver Requisitos do Produto; e Verificar e Validar os Requisitos.

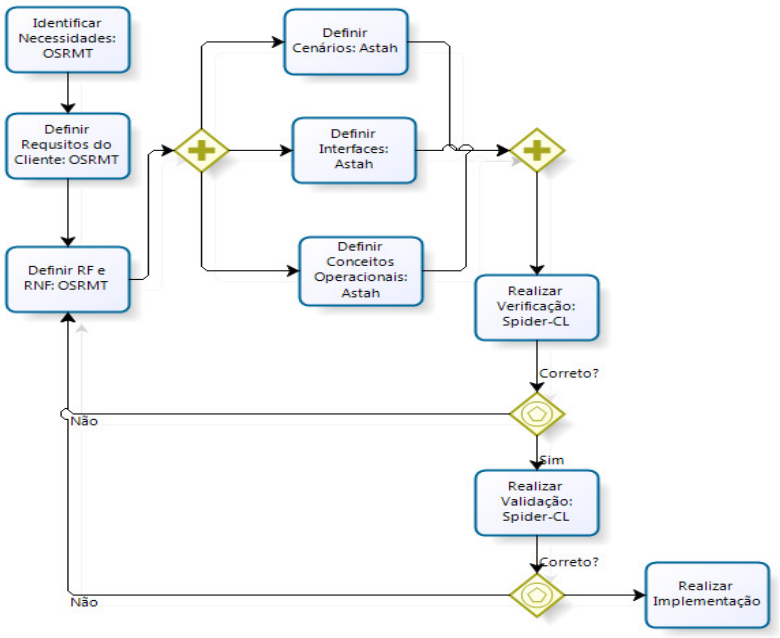


Fig. 1. Fluxo das atividades da metodologia

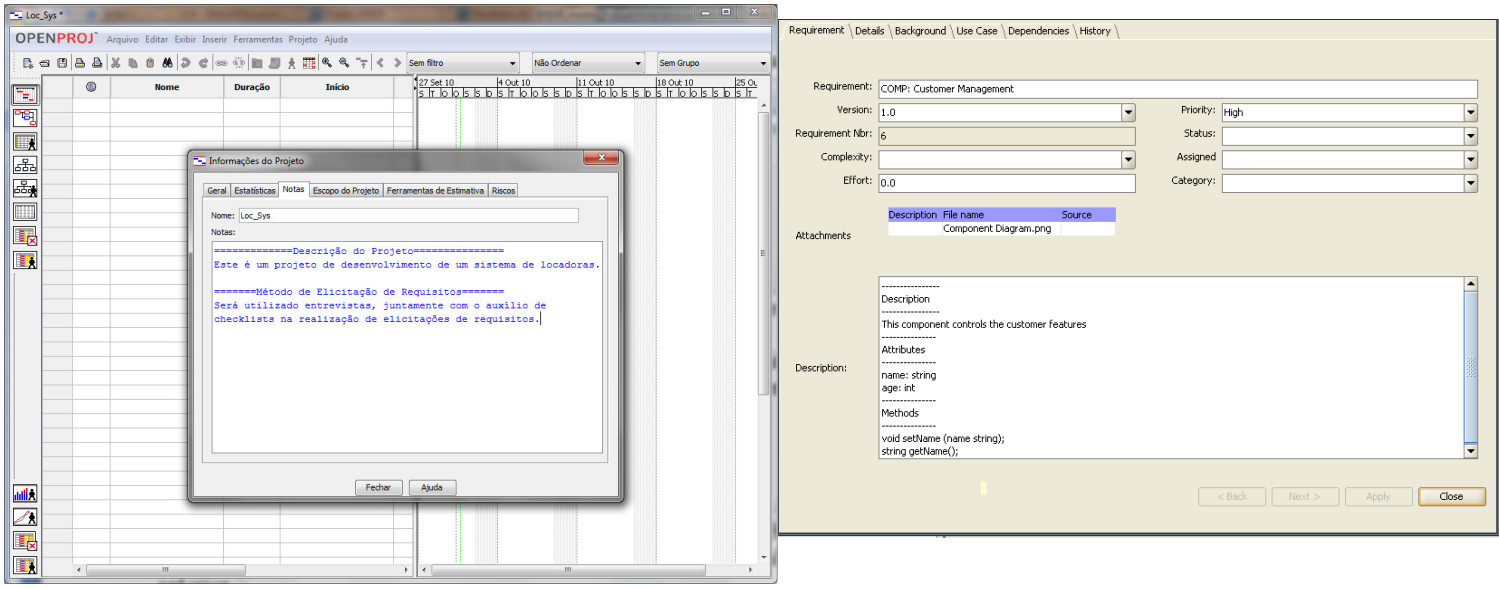
5.1 Desenvolver Requisitos do Cliente

O alcance do resultado esperado DRE1 do MR-MPS, compatível com a prática específica SP1.1 da área de processo RD do CMMI-DEV, envolve a utilização de métodos adequados para identificar necessidades, expectativas, restrições e interfaces do cliente [3][4]. A coleta desses requisitos pode ser feita a partir de métodos de elicitação de requisitos como entrevistas, questionários, *brainstorms*, uso de protótipos, casos de uso, entre outras. Uma boa prática que pode ser adotada é o registro e a descrição da forma de elicitação de requisitos utilizada na empresa, para que estas informações possam ser acessadas por todos da organização.

A descrição da forma de elicitação de requisitos é feita utilizando o campo de descrição do projeto na ferramenta Openproj, como mostra a Figura 2(a). Com o objetivo de separar a descrição do projeto e a forma de coleta de requisitos, pode ser definido um separador, nomeando-se um título para cada um desses itens. Para finalizar a implementação do DRE1/SP1.1, a funcionalidade de *Features* na ferramenta OSRMT é utilizada, onde são cadastradas três *features*: uma representando as necessidades; outra para expectativas; e a última para as restrições do cliente.

Quando as necessidades, expectativas e restrições estão devidamente registradas e detalhadas, é o momento de utilizá-las como insumo para a geração dos requisitos do cliente, para isso, a funcionalidade de *Requirement* da ferramenta OSRMT é utilizada. Na metodologia, define-se a criação de um item denominado “Requisitos do Cliente” com o uso da funcionalidade *requirement*, o qual servirá de recipiente para todos os requisitos derivados a partir das necessidades, expectativas e restrições do cliente.

Para cada requisito, um *requirement* referente será criado o qual estará ligado hierarquicamente a “Requisitos do Cliente”.



(a) (b)

Fig. 2 (a). Descrição do método de elicitação de requisitos na ferramenta Openproj; **(b).** Tela de detalhamento do componente com a descrição de suas interfaces.

Cada requisito do cliente gerado deverá ser mapeado às necessidades, expectativas ou restrições que o originou, além de sua descrição, prioridade, complexidade, estado, versão. Caso existam dependências múltiplas (proveniente de um conjunto de necessidades/expectativas/restrições), estas dependências também devem ser evidenciadas. Após esse procedimento, o resultado esperado DRE2 e a prática específica SP1.2 estarão contemplados.

5.2 Desenvolver Requisitos do Produto

Para alcançar o DRE3 e o SP2.1, os requisitos funcionais e não-funcionais são derivados a partir dos requisitos do cliente, que são devidamente registrados, detalhados e mapeados pela ferramenta OSRMT. Salienta-se que a prática específica SP2.1 trata do estabelecimento e refinamento dos requisitos funcionais e não-funcionais.

O registro dos requisitos funcionais e não-funcionais é feito de forma similar ao registro dos requisitos do cliente, o qual deverá ser criado um item denominado de “Requisitos Funcionais e Não-Funcionais”, onde estarão aninhados os requisitos funcionais e não-funcionais. A utilização de critérios para o agrupamento destes requisitos é uma boa prática sugerida pela SOFTEX [3].

Na metodologia, o critério de agrupamento dos requisitos é feito com o uso do conceito de componente. Componente é um bloco construtivo modular para software

de computador [1]. Para a representação de um componente, é criado um item (*requirement*) com o nome no formato "COMP: NOME_DO_COMPONENTE" o qual é registrado dentro do item "Requisitos Funcionais e Não-Funcionais". Em cada componente são cadastrados os requisitos funcionais e não-funcionais do sistema. Com o objetivo de preservar a legibilidade é adotado o uso das *tags* (rótulos) "RF:" e "RNF:", antes do nome de cada requisito, os quais significam requisitos funcionais e requisitos não-funcionais, respectivamente.

Após o cadastro de um requisito funcional ou não-funcional, é imprescindível o seu detalhamento, por isso deve-se descrevê-lo informando sua versão, o nível de sua complexidade e urgência. Além de caracterizar o requisito, é necessário fazer o mapeamento ao componente em que este pertence. O propósito deste mapeamento é evidenciar que o conjunto de requisitos faz parte de um determinado componente, quando uma atividade de rastreabilidade for realizada.

Quando todos os requisitos funcionais e não-funcionais forem detalhados e mapeados ao seu componente, este componente deve, também, ser detalhado. Primeiramente, deve ser feita uma descrição do componente com informações de sua interface (métodos utilizados e dados de entrada e saída), além de descrever suas funcionalidades. Com o objetivo de organizar as informações presentes no campo de descrição, são criados tópicos para separar a descrição, os métodos e os dados de entrada e saída do componente, como visto na Figura 2 (b). A utilização de tópicos não precisa se restringir apenas na separação desses conteúdos, para cada novo conteúdo, um novo tópico pode ser criado.

Depois do registro e detalhamento do componente, faz-se necessário refiná-lo, ou seja, devem ser utilizadas técnicas de modelagem para o desenvolvimento de diagramas com o objetivo de proporcionar uma análise do projeto com diferentes perspectivas. A ferramenta responsável pela modelagem, utilizada na metodologia, é a *Astah Community*, a qual permite a elaboração de diagramas no modelo UML.

No momento em que os diagramas são elaborados, é necessário disponibilizá-los e mapeá-los aos requisitos que os geraram. Para isso, duas abordagens são adotadas na metodologia: (1) os diagramas produzidos são anexados aos componentes ou requisitos que os geraram; (2) no campo "Design" é criado *designs* que representam os diagramas. Na ferramenta OSRMT, *design* é o componente que representa e descreve os dados arquiteturais do sistema. Para cada *design*, é anexado apenas um tipo de diagrama e seu nome é associado a este tipo, ou seja, se o diagrama anexado ao *design* for um diagrama de classes, este item será rotulado como "Diagrama de Classes".

A abordagem (1) é utilizada nos casos em que os diagramas gerados representam uma parte específica do projeto, como um componente ou requisito. Em casos em que o diagrama possui um escopo mais abrangente, como a representação da comunicação entre componentes, é feito uso da abordagem (2).

Cada diagrama desenvolvido permite uma análise em uma determinada perspectiva, porém, para o MR-MPS e o CMMI-DEV, devem ser gerados diagramas que modelem cenários, conceitos operacionais e interfaces internas e externas, obviamente não se restringindo apenas a essas visões.

Na metodologia, são modelados casos de uso para evidenciar que os cenários foram desenvolvidos. Cada caso de uso produzido é anexado ao componente ou ao requisito que o gerou. Depois de realizado o anexo, devem ser descritas suas pré e

pós-condições, juntamente com seus fluxos principal e alternativo no guia "Use Case", na tela de detalhamento do requisito.

Os conceitos operacionais podem ser implementados com o uso de diagramas de implantação, navegabilidade ou atividades. Esta metodologia faz uso do diagrama de atividades. Cada organização está livre para escolher a melhor forma de representar os conceitos operacionais do projeto, não estando restritos a apenas um tipo de diagrama.

Primeiramente, deve-se criar um item, utilizando a funcionalidade *design*, com o nome de "Diagrama de Atividades" para realizar o anexo dos diagramas de atividades. Outra necessidade é o detalhamento de cada atividade que compõe o diagrama, além de seu mapeamento aos requisitos que o geraram. Após a definição dos cenários e conceitos operacionais, o resultado esperado DRE6, juntamente com a prática específica SP3.1, estão totalmente satisfeitos.

Com o propósito de alcançar o DRE5 e o SP2.3, faz-se necessário elaborar as interfaces internas e externas do sistema. Para contemplar este resultado esperado, na metodologia são modelados os diagramas de componente contendo suas interfaces de comunicação. Este diagrama deve ser anexado a um *design* denominado "Diagrama de Componentes", além de realizar seu detalhamento. Também é importante que o diagrama de componentes esteja mapeado ao componente ou requisitos funcionais e não-funcionais que serviram de insumo para seu desenvolvimento.

A evidência da implementação do DRE5 e SP2.3 é feita quando for presenciado o diagrama de componentes, juntamente com suas interfaces e descrição. Outra evidência é a presença da lista de métodos e dados de entrada e saída no campo de descrição dos componentes.

Após definir os cenários, os conceitos operacionais e as interfaces, o projeto possuirá insumos suficientes para evidenciar um atributo denominado de Análise Funcional, permitindo contemplar a prática específica SP3.2. O resultado esperado DRE4 está parcialmente contemplado devido ao fato das alocações das tarefas ainda não terem sido realizadas. Análise Funcional é o exame de uma determinada função para a identificação de todas as subfunções necessárias para a sua execução, incluindo a identificação de seus relacionamentos e interfaces [4].

5.3 Verificar e Validar os Requisitos

Quando todos os diagramas necessários são modelados, descritos e mapeados, deve-se alocar algum membro responsável para realizar a verificação, validação e implementação dos componentes, junto aos seus diagramas. A alocação é feita atribuindo o nome do responsável no campo "Assigned", na tela de detalhamento do componente na ferramenta OSRMT.

Além de designar um responsável na ferramenta OSRMT, uma *issue* deve ser instanciada na ferramenta Redmine com o objetivo de informar a alocação da tarefa à pessoa responsável, além de possibilitar o acompanhamento da evolução da *issue* alocada. Uma *issue* é a representação de uma solicitação de mudança, tarefa, defeito ou problema que tem sua evolução acompanhada desde sua criação até seu desfecho.

Nesta metodologia, para cada tarefa de verificação, validação e implementação, é instanciada uma *issue* do tipo tarefa com o nome seguindo a forma de nomenclatura:

- *Verificação do componente "Nome_do_componente":* Para tarefa de verificação;

- *Validação do componente "Nome_do_componente":* Para tarefas de validação;
- *Implementação do componente "Nome_do_componente":* Para tarefas de implementação.

Um componente, antes de ser implementado, deve passar pelo processo de verificação e, em seguida, validação. Com o objetivo de auxiliar no processo de verificação e validação, é utilizado o conceito de *checklist*. Na metodologia, a notificação e acompanhamento das tarefas de verificação, validação e implementação são feitas a partir da ferramenta Redmine, enquanto que os critérios objetivos são cadastrados e aplicados pela ferramenta Spider-CL.

O primeiro processo que deve ser executado é a verificação, com o objetivo de detectar erros, inconsistências, ambiguidades, omissões no documento de requisitos. Na metodologia, é criada uma tarefa solicitando que um componente seja verificado, juntamente com a disponibilização do *checkilist* a ser aplicado. Caso seja detectado algum defeito, o estado da tarefa é modificado para "Feedback" e ocorre a alocação de um responsável para realizar as devidas correções, além de proceder o anexo do *checklist* aplicado (em formato pdf) à tarefa. Por fim, o componente verificado, retorna para a fase de análise de requisitos.

Caso não tenha sido detectado nenhum defeito no componente, o estado da tarefa é modificado para "Resolved" e o *checklist* aplicado é anexado. Além disso, uma tarefa de validação para o componente é instanciada no Redmine.

O processo de validação tem o objetivo confirmar se o produto ou componente do produto atende o seu uso pretendido quando colocado no ambiente para qual foi desenvolvido [3]. Diferente da verificação, a validação é executada pelo cliente externo, por esse motivo, deve ser disponibilizado um usuário de acesso na ferramenta Spider-CL com o perfil de cliente externo.

Quando o componente receber a aprovação do cliente, a tarefa deve ser modificada para o estado "Resolved" e o *checklist* aplicado deve ser anexado. Enfim, uma tarefa solicitando a implementação do componente deve ser instanciada, juntamente com a alocação de um responsável para resolvê-la.

Caso o componente não seja validado, o estado da tarefa é modificado para "Feedback" e o *checklist* aplicado é anexado. O componente reprovado na validação é enviado para a fase de análise, visando corrigir os problemas detectados. Como consequência, o componente passa pela fase de verificação novamente, com o objetivo de detectar possíveis novos defeitos provenientes das modificações realizadas.

No momento em que as tarefas de refinamento, detalhamento e alocação dos requisitos funcionais e não-funcionais são feitas, o resultado esperado DRE4 e a prática específica SP2.2 são alcançados completamente. Deve-se salientar que a prática específica SP2.2 trata apenas de alocação de responsáveis para as tarefas de verificação, validação e implementação. O estabelecimento e refinamento de requisitos funcionais e não-funcionais é de responsabilidade do SP2.1.

Quando os procedimentos para gerenciar os ciclos de vida das tarefas de verificação e validação forem executados, os resultados esperados DRE7 e DRE8 serão contemplados, respectivamente. No caso do CMMI-DEV, as práticas específicas SP3.3 e SP3.4 são satisfeitas no momento que forem gerenciadas as

tarefas de verificação, e a prática específica SP3.5 será contemplado quando as atividades de validação forem gerenciadas.

6 Análise de Aderência

A análise de aderência da metodologia proposta na Seção 5 é evidenciada a partir dos mapeamentos das funcionalidades e serviços presentes no conjunto de ferramentas com as práticas sugeridas pelos resultados esperados e práticas específicas do Processo de Desenvolvimento de Requisitos. A análise de aderência é sintetizada no Quadro 1 o qual mostra a relação entre os resultados esperados (RE), as práticas específicas (SP) e as ferramentas propostas, juntamente com uma breve descrição das funcionalidades e práticas que implementam cada resultado esperado. A aderência dos resultados esperados do MR-MPS às práticas específicas do CMMI-DEV está baseada nos mapeamentos publicados em [19].

Quadro 1. Aderência entre as funcionalidades das ferramentas, as práticas específicas e os resultados esperados do processo DRE

RE	SP	Ferramentas	Funcionalidades/Práticas
DRE1	SP1.1	Openproj/ OSRMT	Definição do método de elicitação de requisitos no campo de notas; Registro e descrição das necessidades, expectativas e restrições no campo *Features*.
DRE2	SP1.2	OSRMT	Registro e descrição dos requisitos do cliente no campo *Requirements*.
DRE3	SP2.1	OSRMT	Registro e descrição dos requisitos funcionais e não-funcionais no campo *Requirements*.
DRE4	SP2.2 SP3.2	OSRMT/ Redmine/ Astah Community	Anexo de diagramas aos componentes e requisitos; Descrição dos diagramas nos componentes e requisitos; Alocação de um responsável na tela de detalhamento do requisito ou componente; Instanciação e acompanhamento de uma *issue* para alocação de tarefa.
DRE5	SP2.3	OSRMT/ Astah Community	Definição dos métodos e dados de entrada e saída dos componentes no campo *description;* Criação de diagramas de componente contendo suas interfaces.
DRE6	SP3.1	OSRMT/ Astah Community	Criação do diagrama de atividades; Descrição de cada atividade no *design* que corresponde ao diagrama de atividades; Criação dos diagramas de casos de uso; Descrição dos fluxos principal e alternativo,

RE	SP	Ferramentas	Funcionalidades/Práticas
			junto com as pré e pós-condições no guia *Use Case*, da tela de detalhamento do requisito.
DRE7	SP3.3 SP3.4	Redmine/ Spider-CL	Instanciação e acompanhamento da tarefa (*issue*) de verificação dos requisitos; Criação do *checklist* contendo os critérios objetivos para realizar a verificação.
DRE8	SP3.5	Redmine/ Spider-CL	Instanciação e acompanhamento da tarefa (*issue*) de validação dos requisitos; Criação do *checklist* contendo os critérios objetivos para realizar a validação.

7 Considerações Finais

Como apresentado no Quadro 1, as ferramentas analisadas neste trabalho possibilitam a implementação sistêmica do processo de Desenvolvimento de Requisitos, em conformidade com as práticas sugeridas pelo MR-MPS e CMMI-DEV, de forma a estender a metodologia de apoio à Gerência de Requisitos, anteriormente proposta em [7]. O processo de Desenvolvimento de Requisitos constitui uma fase crítica durante o desenvolvimento do produto, sendo fundamental para garantir a definição e a descrição correta dos requisitos, base para o alcance da qualidade de software.

O conjunto de ferramentas livres proposto é capaz de implementar totalmente os resultados esperados do processo de Desenvolvimento de Requisitos do MR-MPS e as práticas específicas da área de processo *Requirements Development* do CMMI-DEV, desde que seja utilizado de forma planejada, conforme a metodologia apresentada. Também é importante ressaltar que a metodologia apresentada constitui boas práticas de uso das ferramentas para atender às sugestões do MR-MPS e CMMI-DEV sem, no entanto, estabelecer um processo de Desenvolvimento de Requisitos.

Como trabalhos futuros, pretende-se realizar a integração das ferramentas propostas no escopo do Projeto SPIDER para o desenvolvimento de um *Suite* de ferramentas de software livre para apoio aos programas MPS.BR e CMMI, além de um estudo de caso da metodologia, realizando implementações em ambientes de desenvolvimento das empresas/instituições de desenvolvimento de software parceiras do Projeto SPIDER para a análise dos pontos fortes, fracos e oportunidades de melhoria.

Deve-se salientar que, inicialmente, a implementação da metodologia proposta acarretará em mudanças de cultura na organização com uma possível diminuição em sua produtividade. Além disso, poderá ser necessária a adaptação do processo organizacional para se adequarem ao uso de ferramentas. Porém, a médio e longo prazo, a produtividade da organização poderá sofrer enormes melhorias, pelo fato das ferramentas serem capazes de substituir grande parte das documentações geradas. Outra vantagem seria o custo de implantação das ferramentas ser nulo, pelo fato da gratuidade/disponibilidade das ferramentas.

Referências

1. Pressman, R. S. Engenharia de software. Tradução Rosângela Velloso Penteado. São Paulo: McGraw-Hill, 2006.
2. Associação para Promoção da Excelência do Software Brasileiro. MPS.BR – Guia Geral: 2011". Disponível em: www.softex.br. Último acesso em 07 de agosto de 2011.
3. Associação para Promoção da Excelência do Software Brasileiro. MPS.BR – Guia de Implementação – Parte 4: 2011. Disponível em: www.softex.br. Último acesso em 07 de agosto de 2011.
4. Software Engineering Institute. CMMI® for Development. Version 1.3, Carnegie Mellon University, Software Engineering Institute, Pittsburgh, 2010.
5. Sommerville, I. Software Engineering. Addison Wesley, 8ª edição, 2007.
6. Oliveira, S. R. B. et al. SPIDER - Um *Suite* de Ferramentas de Software Livre de Apoio à Implementação do modelo MPS.BR. Anais do VIII Encontro Anual de Computação – ENACOMP 2010. Catalão, Goiás, 2010.
7. Cardias Junior, A. B. et al. Uma Análise Avaliativa de Ferramentas de Software Livre no Contexto da Implementação do Processo de Gerência de Requisitos do MPS.BR. Anais do WER10 - Workshop em Engenharia de Requisitos. Cuenca, Equador, 2010.
8. International Organization for Standardization/ International Eletrotechnical Commission. ISO/IEC 12207 Systems and software engineering – Software life cycle processes. Geneve: ISO, 2008.
9. Sourceforge. Open Source Requirements Management Tool. Disponível em: http://sourceforge.net/projects/osrmt/. Último acesso em 03/11/2010.
10. Serena. OpenProj. Disponível em: http:// http://openproj.org/openproj. Último acesso em 03/11/2010.
11. Rubyforge. Redmine. Disponível em http:// http://rubyforge.org. Último acesso em 03/11/2010.
12. Yoshidome, E. Y. C. et al. Uma Implementação do Processo de Gerência de Projetos Usando Ferramentas de Software Livre. Anais do VI Workshop Anual do MPS – WAMPS 2010. Campinas, São Paulo, 2010.
13. Barros, R. S., Oliveira, S. R. B. Spider-CL: Uma Ferramenta de Apoio ao Uso de Critérios Objetivos no Contexto da Qualidade de Software. Anais da II Escola Regional de Informática – ERIN 2010. Manaus, Amazonas, 2010.
14. Astah. Astah Community. Disponível em http://astah.change-vision.com. Último acesso em 03/11/2010.
15. Martins, A. F. et al. ReqODE: Uma Ferramenta de Apoio à Engenharia de Requisitos Integrada ao Ambiente ODE. Anais da XIII Sessão de Ferramentas. XX Simpósio Brasileiro de Engenharia de Software. Florianópolis-SC, Brasil, 2006.
16. Zaniro, D. e Fabbri, S. Um Processo Guiado para o Levantamento e Modelagem de Requisitos de Aplicações Web Baseado em Objetivos e Casos de Uso. XI Workshop em Engenharia de Requisitos. Catalonia, Espanha, 2008.
17. Silva, W. C. e Martins, L. D. G. PARADIGMA: Uma Ferramenta de Apoio à Elicitação e Modelagem de Requisitos Baseados em Processamento de Linguagem Natural. XI Workshop em Engenharia de Requisitos. Catalonia, Espanha, 2008.
18. Zanlorenci, E. P. e Burnett, R. C. Ferramenta de Apoio aos Processos da Engenharia de Requisitos, na Fase de Projetos. III Workshop em Engenharia de Requisitos. Rio de Janeiro, Brasil, 2000.
19. Associação para Promoção da Excelência do Software Brasileiro. MPS.BR – Guia de Implementação – Parte 11: 2011". Disponível em: www.softex.br. Último acesso em 07 de agosto de 2011.