OpenReq: uma Ferramenta para Auxílio à Gerência de Requisitos

Claiton Luís Grings, Miriam Sayão
Pontifícia Universidade Católica do Rio Grande Do Sul
{claiton.grings@acad.pucrs.br, miriam.sayao@pucrs.br}

Abstract

The activity of requirements specification demands considerable efforts from professionals that execute it. To accomplish the activity, it's not enough to obtain the requirements with customers. It is necessary to have a clear and complete understanding of customer's necessities.

Considering the issues associated to requirement's registration and management, and the necessity of process's optimization during product's conception, the OpenReq tool was developed.

The objective of OpenReq is to offer the possibility of requirements managements through an automated tool that is different from the other tools because it is open source and consequently, doesn't have monetary costs for its utilization.

The OpenReq was developed to Web platform, enabling the decentralized management of software requirements in just one base. The creation of this tool looks for the stimulation of the adoption of requirement management practices for small and medium companies and contribute with open source development community too, using open source technology.

1. Introdução

De acordo com as metodologias clássicas de desenvolvimento, o ciclo de desenvolvimento de um produto de software, tem início pelo Processo de Requisitos. Esse processo constitui-se, basicamente, de um conjunto de atividades divididas em Elicitação, Modelagem e Análise de Requisitos [1].

A atividade de elicitar (ou identificar) requisitos demanda considerável esforço dos profissionais envolvidos em sua execução, pois exige uma completa compreensão do problema, sendo imprescindível não apenas obter um bom entendimento, como também definir requisitos de forma clara.

A complexidade apresentada para a execução dessa tarefa determina que esta seja uma das atividades mais difíceis de um projeto de software; erros cometidos nesta etapa podem implicar em alterações não apenas nos requisitos, mas também em muitos outros artefatos gerados no processo de desenvolvimento. É fato conhecido que a correção de erros originados em requisitos implica em retrabalho, cujos custos são da ordem de 10 a 100 vezes o custo da correção desses mesmos erros, caso houvessem sido detectados nas etapas iniciais do processo de desenvolvimento [2].

Além do surgimento de erros, não é incomum que os requisitos já definidos sofram alterações ao longo do processo de desenvolvimento. Mudanças no contexto onde o software está inserido, novas expectativas por parte dos clientes e usuários, negociação entre clientes e desenvolvedores, são apenas alguns exemplos de motivos que levam um requisito a ser modificado. Desta forma, mesmo que se faça um bom trabalho de elicitação e gerência de requisitos, à medida que aumenta o número de requisitos que sofrem modificações, o esforço necessário para manter e gerenciar esses artefatos tende a crescer.

Dentro desse contexto procuramos, através da criação do *OpenReq*, oferecer a possibilidade de realizar o gerenciamento de requisitos através de uma ferramenta específica para o atendimento das necessidades derivadas dessa atividade. A ferramenta aqui apresentada possui as principais funcionalidades encontradas nas ferramentas comerciais mais utilizadas atualmente, porém diferencia-se delas por possuir código aberto e consequentemente, não apresentar custo para a sua utilização. A ferramenta deve apoiar atividades associadas à Gerência de Requisitos, que engloba o Controle de Mudanças, a Gerência de Configuração, a Rastreabilidade e Gerência da Qualidade de Requisitos. O uso de uma ferramenta própria a tais atividades pode contribuir de forma significativa na melhoria da qualidade do processo de requisitos e na redução dos esforços necessários à sua execução.

2. Gerência de Requisitos

A Gerência de Requisitos está associada ao processo de Engenharia de Requisitos, uma subárea da Engenharia de Software, que engloba todas as atividades relacionadas com a produção de requisitos de software.

Suas principais atividades incluem:

- **Controle de Mudanças:** Pressupõe um processo de mudança controlado e bem definido. Deve existir um mecanismo formal para solicitação de mudanças nos requisitos do software.
- **Gerência de Configuração:** Apresenta duas atividades principais: Controle de versões e Controle de Configuração. O *Controle de versões* é a atividade relacionada à evolução dos artefatos, dando atenção aos artefatos gerados no processo de desenvolvimento, e às diferentes revisões e versões geradas por modificações. Já o *Controle de configuração*, está relacionado às *releases* do software e o seu foco está voltado a diferentes configurações do software, considerando possíveis combinações de artefatos.
- **Rastreabilidade:** Está no centro da atividade de gerenciamento de requisitos. É definida como a habilidade de se acompanhar a vida de um requisito durante todo o processo de construção.
- **Gerência da Qualidade de Requisitos:** A qualidade de um artefato de requisitos pode ser avaliada através da utilização de métricas, indicadores ou testes. Uma análise crítica desses artefatos pode ser realizada através de inspeções.

Uma das principais justificativas para empregar o esforço necessário para a realização dessas atividades é garantir que mesmo executando um bom processo de análise, construção e verificação de qualidade, não se obtenha, como resultado final, um software que não atenda às necessidades do cliente.

2.1. Foco deste trabalho

As atividades a serem atendidas por uma ferramenta de apoio ao processo de requisitos são muitas, e nesse trabalho priorizamos atividades relacionadas ao processo de elicitação e modelagem de requisitos. A ferramenta desenvolvida, descrita na seção 4 deste artigo, mantém a base de requisitos através de um editor próprio, registra requisitos modelados em sentenças, executa controle de versões, permite o acesso a sua base através de um navegador (ou *browser)* e apóia o processo de controle de mudanças.

As funcionalidades implementadas estão agrupadas em:

- Segurança: componente responsável por validar o acesso dos usuários ao sistema através de login e senha;
- Controle de Acesso: responsável por garantir o acesso apenas aos usuários cadastrados, restringindo suas funcionalidades conforme as políticas de permissão do usuário;
- Gerência de Módulos: permite a adição, configuração e gerência de módulos da ferramenta;
- Gerência de Usuários: permite criar e gerenciar os usuários que terão acesso a ferramenta;
- Controle de Mudanças: gerencia as alterações ocorridas nos artefatos gerenciados pelo sistema;
- Controle de Versões: controla as diversas versões geradas a partir das modificações ocorridas nos artefatos gerenciados pelo sistema.

3. Ferramentas para Controle de Requisitos

Nos últimos anos muito se tem falado a respeito de modelos e normas de qualidade, bem como a importância dos mesmos para o desenvolvimento de softwares que não apenas atendam as necessidades do cliente, mas que também o façam apresentando como resultado produtos de qualidade.

Organizações mundiais tais como ISO, IEEE, PMI e SEI, propõem modelos e padrões visando à melhoria do processo de produção de software. Ao consultar as normas e modelos disponibilizados por essas organizações, é possível identificar a presença de seções que tratam especificamente de requisitos de software, desde a atividade de elicitação de requisitos, até o gerenciamento de suas mudanças. Alguns chegam a apontar que um dos primeiros aspectos a serem tratados na busca pela qualidade do processo é a gerência de requisitos. Isso reforça a importância dessa atividade durante o processo de desenvolvimento.

Segundo Wiegers[3], uma ferramenta de gerenciamento de requisitos traz grandes benefícios em um projeto de desenvolvimento de software, tais como:

- Gerenciar versões e mudanças: O projeto deve definir uma *baseline* de requisitos, o que é importante para manter um histórico das mudanças realizadas em cada requisito. Deve ser possível armazenar cada alteração realizada no requisito, bem como reverter para uma versão prévia tais mudanças, caso necessário;
- Armazenar os atributos dos requisitos: Ferramentas de gerenciamento de requisitos geram vários atributos definidos pelo sistema, tais como: a data em que o requisito foi criado e seu número de versão. Uma ferramenta de gerenciamento de requisitos deve permitir a definição de atributos adicionais de vários tipos de dados;

- Facilitar a análise de impacto: As ferramentas permitem rastrear requisitos através de *links* definidos entre diferentes tipos de requisitos, entre requisitos em diferentes subsistemas, entre requisitos individuais e relacionados. Esses *links* ajudam na análise de impacto que uma mudança proposta terá em um requisito específico, através da identificação de outros elementos do sistema que essa mudança poderá afetar;
- Acompanhamento do estado do requisito: Coletar os requisitos em uma base de dados permite saber quantos requisitos foram especificados para o produto. O acompanhamento do estado dos requisitos durante o desenvolvimento permite o acompanhamento da situação de todo o projeto;
- Controle de acesso: Uma ferramenta de gerenciamento de requisitos define permissões de acesso para indivíduos ou grupos de usuários e compartilha informação entre um time distribuído geograficamente através de uma interface *Web* à base de dados;
- Comunicação com as partes interessadas: Algumas ferramentas permitem que membros de uma equipe discutam, eletronicamente, questões relativas aos requisitos. Mensagens enviadas de forma automática por e-mail notificam as partes afetadas quando uma nova discussão é iniciada ou quando um requisito específico é modificado;
- Reuso de requisitos: Armazenar os requisitos em uma base de dados permite sua reutilização em vários projetos. Requisitos que servem para múltiplas partes da descrição do produto podem ser armazenados e referidos sempre que necessário, para evitar requisitos duplicados.

Como não poderia deixar de ser, a indústria de desenvolvimento de software vem trabalhando com o intuito de oferecer produtos para atender essa demanda, disponibilizando ferramentas que auxiliem nas atividades relativas ao processo de gerência de requisitos.

Durante o processo de concepção do *OpenReq*, foram selecionadas três ferramentas comerciais para controle de requisitos: *RequisitePro*, *CaliberRM* e *DOORS*. Essas ferramentas foram estudadas e analisadas com o objetivo de identificar funcionalidades comuns e especificidades, que pudessem ser incorporadas à nova ferramenta.

As ferramentas comerciais escolhidas foram selecionadas levando em consideração a sua representatividade e a sua atual utilização no mercado.

O *RequisitePro*[10] é uma ferramenta da IBM que possibilita a gerência de requisitos do sistema. Os requisitos definidos possuem atributos que descrevem os mesmos em termos de prioridade, status ou outras características definidas pelo usuário. Através do *RequisitePro*, os requisitos são organizados em uma base de dados centralizada. A ferramenta possui integração com o *Microsoft Word*, provendo uma arquitetura que permite que os documentos de requisito sejam dinamicamente ligados a essa base. A versão da ferramenta que foi analisada para estudo é a versão 7.0.1.

O *CaliberRM*[11] é um sistema de automação de gerenciamento de requisitos cliente servidor desenvolvido pela Borland. A ferramenta possui objetivos similares ao *RequisitePRO* e *DOORS*, porém, ao invés de tratar os requisitos como documentos, o *CaliberRM* os trata como um grupo de objetos integrados e reutilizáveis que são armazenados em repositório central, permitindo que um mesmo requisito possa ser utilizado em vários projetos gerenciados pela ferramenta. A versão do *CaliberRM* que foi utilizada para avaliação, é a versão 10.0.229.

O *DOORS*[12] é um sistema de gerenciamento de requisitos cliente servidor desenvolvido pela *Telelogic*. A ferramenta trata os requisitos como documentos da mesma forma que o *RequisitePro*, porém, com uma interface de edição própria, não havendo a necessidade de utilização de um editor externo. O *DOORS* não oferece a funcionalidade de fórum de discussões, o que o difere das ferramentas previamente analisadas. A versão 8.0 do *DOORS* foi utilizada para avaliação.

Na seção seguinte, será apresentada a ferramenta *OpenReq*, bem como os detalhes relativos ao seu processo de concepção e uso e, em seguida, na seção 5, será apresentado um estudo comparativo entre o *OpenReq* e as ferramentas comerciais analisadas, buscando apontar suas diferenças e pontos comuns.

4. Desenvolvimento da ferramenta *OpenReq*

O *OpenReq* foi inicialmente concebido com a proposta de disponibilizar, através de uma ferramenta de código aberto, as principais funcionalidades existentes nas ferramentas comerciais de gerência de requisitos mais utilizadas no mercado.

Porém, logo se percebeu que esta proposta poderia ser ampliada para permitir não somente a gerência de requisitos, mas também auxiliar no controle das demais etapas de um processo de construção de software. Assim, o projeto inicial foi modificado para o formato de uma estrutura modular, que permite adicionar novas funcionalidades através do conceito de *plug-in*. Desta forma, o *OpenReq* fornece uma estrutura básica de funcionalidades onde novas extensões (plug-ins) podem ser adicionadas, ampliando suas funções.

A Figura 1 apresenta os componentes que formam a estrutura básica da ferramenta.

Figura 1: OpenReq - estrutura básica

O componente de *Controle de Acesso e Segurança* é responsável por validar o acesso dos usuários ao sistema através de *login* e senha (autenticação) e por restringir o acesso dos mesmos, conforme as regras descritas na política de permissão de cada perfil de usuário (autorização).

O componente de *Administração de Usuários* permite criar e gerenciar os usuários que poderão utilizar a ferramenta.

A *Administração de Projetos* permite realizar o cadastro e acompanhamento dos projetos de desenvolvimento de software que serão gerenciados com o auxilio da ferramenta. Neste componente também é realizada a definição do papel (função) que cada usuário irá desempenhar dentro do projeto.

O *Controle de Mudanças* gerencia as alterações ocorridas nos artefatos (como por exemplo, um requisito) gerenciados pelo sistema.

O componente de *Controle de Versões* gerencia e controla as diversas versões geradas a partir das modificações ocorridas nos artefatos gerenciados pelo sistema.

E por fim, o componente de *Gerência de Plug-ins* permite a adição e configuração dos módulos que podem ser adicionados à ferramenta, com o objetivo de estender as suas funcionalidades.

Assim, as funcionalidades de apoio ao processo de gerência de requisitos, que foram inicialmente previstas como funcionalidades principais do sistema, foram implementadas na forma de um *plug-in* para a nova ferramenta. Para permitir ao usuário, a personalização de determinadas configurações do sistema, também foi desenvolvido o *plug-in* denominado *Configurações do Usuário* que é responsável por armazenar e gerenciar as opções de personalização da ferramenta.

4.1. Utilização da ferramenta

Para possibilitar o acesso e o gerenciamento descentralizado dos requisitos do software, em uma base única e centralizada, a ferramenta foi construída para a plataforma *Web*. Seu projeto foi concebido em uma arquitetura de três camadas e desenvolvido em linguagem Java. A Figura 2 representa o ambiente de operação da aplicação.

O *Servidor Web* gerencia as requisições dos clientes, vindas da Internet, e as repassa ao *Servidor de Regras de Negócio* que possui toda a lógica da aplicação. O *Servidor de Dados* abriga a base de informações que alimenta o sistema e o *Servidor de Arquivos versionados* é o repositório utilizado para armazenar os arquivos contendo os textos dos requisitos, de maneira que cada alteração sofrida no seu conteúdo gere uma nova versão do arquivo.

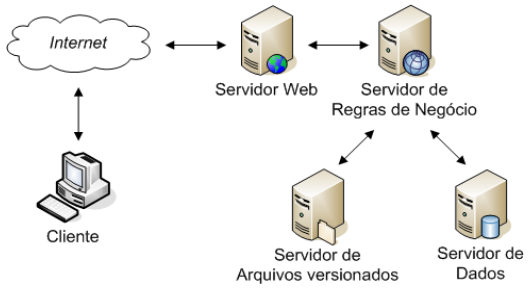

Figura 2: OpenReq - ambiente de operação

Para acessar o *OpenReq* é necessário possuir uma conta de acesso ativa, que é criada pelo administrador do sistema. Após efetuar o acesso, é exibida a tela principal do sistema, onde são listados os módulos aos quais o usuário conectado possui acesso. A Figura 3 demonstra a tela inicial de um usuário com permissão de acesso ao *plug-ins* "Requisitos" e "Preferências do usuário". Na parte superior direita da tela, são exibidos dados referentes ao usuário conectado, o nome do projeto selecionado e o seu perfil dentro do mesmo.

Como o sistema *OpenReq* foi concebido para operar sobre diversos projetos de software, para acessar os módulos que operam sobre um projeto, é necessário primeiramente, indicar o projeto no qual as operações serão realizadas. Caso o usuário não tenha indicado um projeto, o sistema exibe a mensagem "Nenhum projeto selecionado".

Para facilitar a atividade de definição de projeto, a ferramenta oferece a opção de definir um projeto padrão, que será automaticamente selecionado, sempre que o usuário acessar o sistema.

Figura 3: OpenReq - tela principal

Sempre que o usuário tentar acessar um módulo que opera sobre um projeto, sem o mesmo estar definido, o sistema irá apresentar a tela de seleção de projeto de trabalho, conforme apresentado na Figura 4.

Para alterar o projeto selecionado, é necessário escolher a opção "Alterar" disponível ao lado do nome do projeto atual.

Figura 4: OpenReq - seleção de projeto de trabalho

Assim como apenas os módulos que o usuário tem acesso são listados na tela principal do sistema, os projetos disponíveis para o usuário também são restringidos conforme as suas permissões.

Para que um usuário possa ter acesso aos dados de um determinado projeto, é necessário realizar a definição da função que este irá executar no mesmo. Essa definição é realizada pelo gerente do projeto, no módulo de *Manutenção de Projetos*, e é resultado da associação entre um usuário e um perfil. O conceito de "Perfil" é utilizado pelo *OpenReq* para definir quais operações um usuário poderá efetuar no sistema. A Figura 5, demonstra o modelo conceitual que implementa este comportamento no sistema.

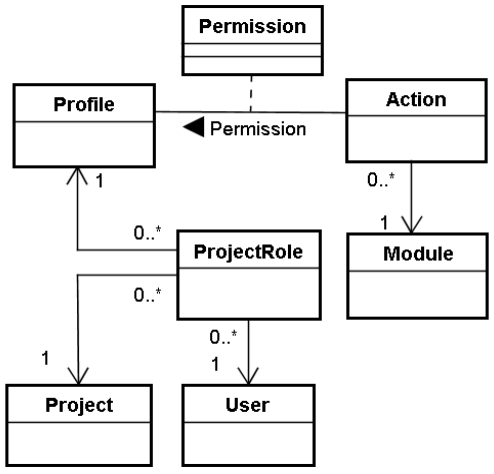

Figura 5: OpenReq - políticas de permissões

A classe *ProjectRole* representa um perfil (*Profile*) de um usuário (*User*) dentro de um projeto (*Project*). Cada usuário poderá ter apenas um perfil em um projeto, mas não necessariamente o mesmo em todos os projetos do sistema. Essa definição é realizada a cada novo projeto, permitindo que um único usuário possa exercer diferentes funções em vários projetos. A classe de permissão (*Permission*) define o conjunto de ações (*Action*) que cada perfil poderá efetuar no projeto.

As políticas de permissão do sistema são bastante flexíveis, e podem ser modificadas pelo administrador do sistema de acordo a necessidade.

O módulo de requisitos do *OpenReq*, dispõe os artefatos em uma estrutura de árvore que possibilita a organização dos requisitos em uma hierarquia de níveis. No nível mais alto dessa árvore, localizam-se os tipos de requisitos e abaixo deste, é possível incluir requisitos, ou ainda, criar grupos de requisitos.

Um grupo de requisitos é um tipo especial de registro criado com o objetivo de agrupar outros requisitos na estrutura da árvore.

Por padrão, o *OpenReq* disponibiliza três tipos de requisitos: *Requisitos Funcionais*, *Requisitos Não Funcionais* e *Requisitos Inversos*. A cada projeto, o gerente define quais os tipos de requisitos estarão disponíveis para o mesmo.

A Figura 6 apresenta o formulário de inclusão de requisitos, onde um novo requisito funcional está sendo criado dentro de um grupo de requisitos previamente criado.

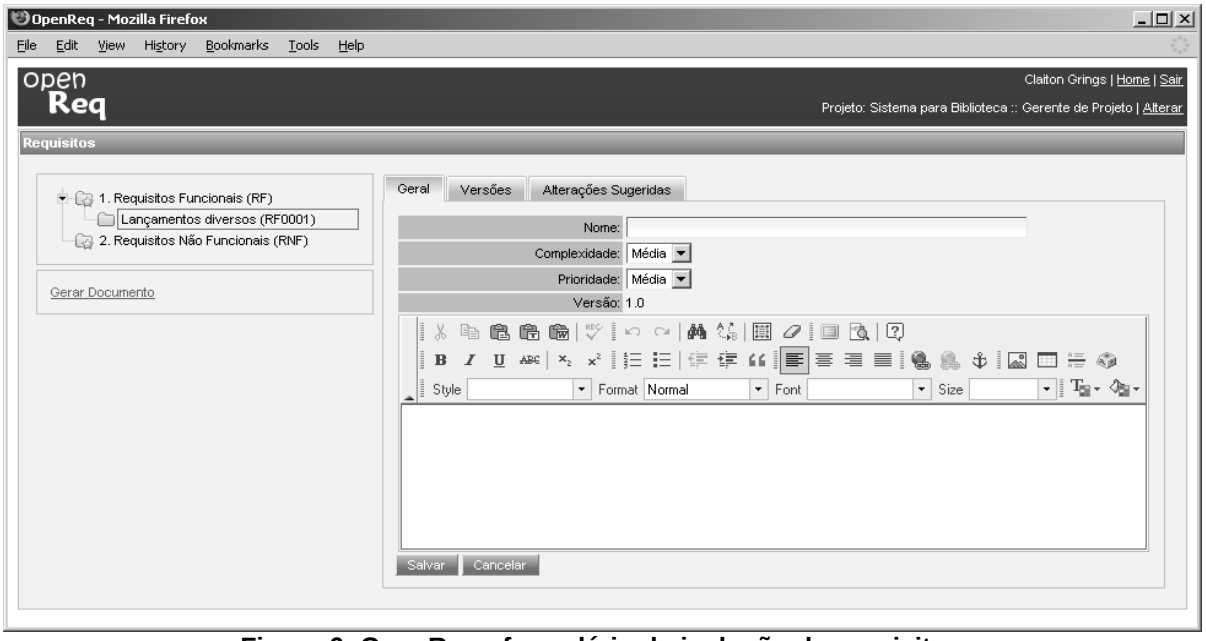

Figura 6: OpenReq - formulário de inclusão de requisitos

Após a criação do requisito, o sistema passa a controlar as mudanças efetuadas no mesmo. A cada alteração sofrida por um requisito, seu número de versão é modificado e novos registros são gerados com o objetivo de manter uma base histórica das modificações realizadas. A Figura 7 demonstra os registros gerados por um requisito que sofreu duas alterações após a sua inclusão.

O controle de versionamento dos requisitos ocorre por uma parte inteira e outra decimal do número da versão. Qualquer modificação realizada nos atributos, sem alteração do texto do requisito, resultará no incremento da parte decimal do número, ao passo que, quando o texto do requisito for modificado, independente dos atributos serem modificados ou não, a unidade do número da versão será incrementada para o próximo número inteiro. Assim, quando o texto de um novo requisito é modificado, o mesmo passa a ter o número de versão 2.0.

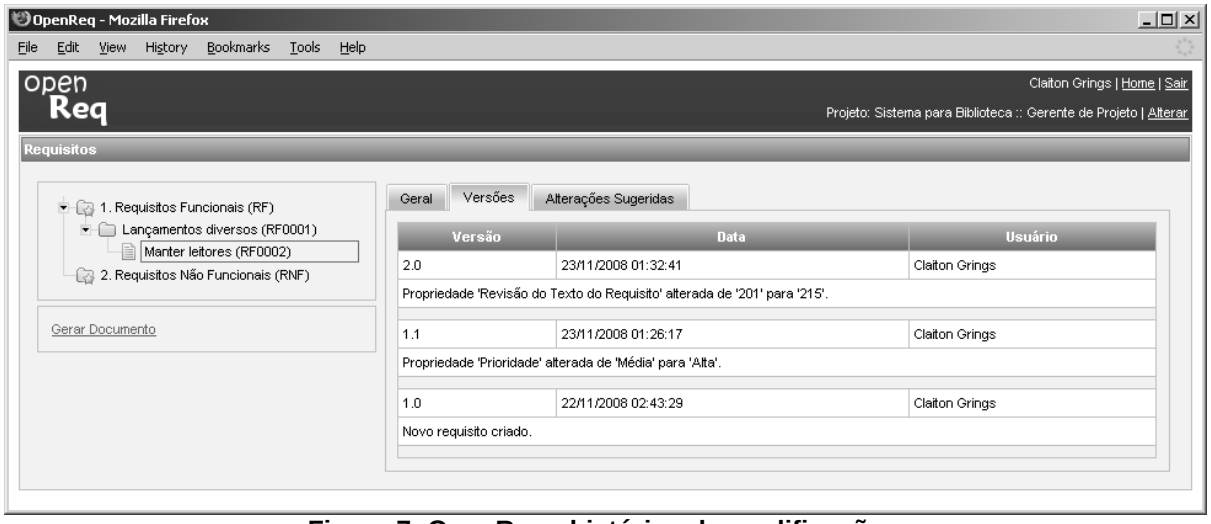

Figura 7: OpenReq - histórico de modificações

A Figura 8 demonstra o modelo conceitual das classes referentes ao módulo de requisitos do *OpenReq*.

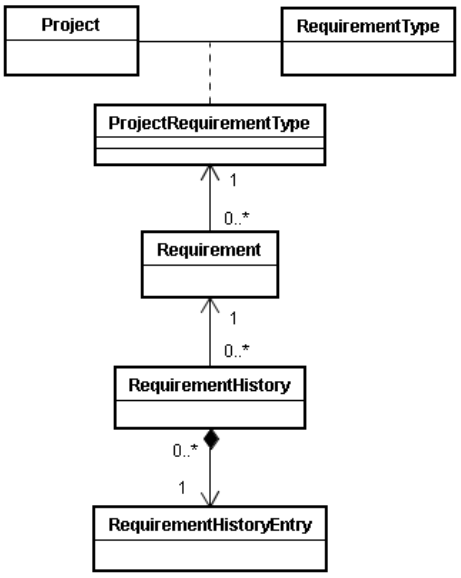

Figura 8: OpenReq - módulo de requisitos

Os tipos de requisitos existentes no sistema são representados pela classe *RequirementType* e a classe *ProjectRequirementType* representa quais tipos de requisitos estarão disponíveis em um projeto.

A classe *Requirement* representa os requisitos gerenciados pelo sistema. Como já foi mencionado, cada modificação de um requisito gera um novo registro de histórico, que no modelo é representado pela classe *RequirementHistory*, e cada atributo modificado gera um registro representado pela classe *RequirementHistoryEntry* que armazena o valor anterior e atual do atributo alterado.

Conforme demonstra a Figura 2 o *OpenReq* utiliza dois servidores para armazenamento de dados. O *Servidor de Dados* abriga toda a base de informações do sistema e o *Servidor de Arquivos versionados*, que é um sistema de controle de versão[1], utilizado para armazenar e controlar os arquivos contendo os textos dos requisitos. Todo novo requisito criado gera um arquivo no formato *HTML* contendo o texto do mesmo. Esse arquivo é mantido nesse repositório e a cada modificação realizada no texto, uma nova versão do arquivo é gerada. Por padrão, o *OpenReq* utiliza o sistema de controle de versão *Subversion*, entretanto, pode ser utilizado qualquer outro sistema, como o *CVS* por exemplo. Porém, para isso é necessário modificar a implementação padrão, de acordo com as especificações do sistema que se deseja utilizar.

Ainda sobre o armazenamento de dados, cabe destacar que o *OpenReq* não está limitado a utilização de apenas um SGBD[2] pois o sistema faz uso de um *framework* de persistência que lhe confere a possibilidade de utilizar de forma transparente diferentes SGBDs.

Outro benefício disponível no *OpenReq* é a possibilidade de operar em diferentes idiomas que são definidos em arquivos de internacionalização. Inicialmente o sistema oferece suporte aos idiomas

[1] Um sistema de controle de versão (VCS - do inglês *Version Control System*) é um software com a finalidade de gerenciar diferentes versões de um artefato. Entre os sistemas mais conhecidos para essa funcionalidade encontram-se as soluções livres: *Subversion* e *CVS*.

[2] Sistema Gerenciador de Banco de Dados ou SGBD é um conjunto de programas responsáveis pelo gerenciamento de uma base de dados.

"Português do Brasil" (pt_BR) e "Inglês Americano" (en_US). Para adicionar suporte a outro idioma no sistema, não é necessária nenhuma mudança no código fonte, basta fornecer o novo arquivo de internacionalização e realizar os ajustes de configuração necessários.

5. Comparativo entre as ferramentas

Nesta seção é apresentada uma análise comparativa da ferramenta *OpenReq* com as ferramentas comercias (RequisitePro, CaliberRM e DOORS) citadas na seção 3 deste artigo.

Cada ferramenta foi analisada sob dez aspectos: tipos de requisitos e seus atributos, cadastro e edição de requisitos, controle e histórico de alterações, *baseline* e controle de versão, fórum de discussões, integração com outras ferramentas, segurança, acesso concorrente, acesso *Web* e restrições de ambiente.

Os aspectos analisados foram selecionados com base no estudo intitulado "*Requirements for Requirements Management Tools*" [4], que tem como propósito auxiliar os usuários na atividade de comparar e selecionar ferramentas de gerência de requisitos, assim como fornecer subsídios a desenvolvedores no projeto e definição de novas ferramentas que possam vir a ser construídas.

Baseado nesse estudo segue uma breve descrição da análise realizada sobre os aspectos citados acima, do ponto de vista de cada ferramenta. No caso do *OpenReq*, também é possível constatar quais aspectos foram contemplados pela ferramenta e foram implementados durante o desenvolvimento deste trabalho.

i) Tipos de requisitos e seus atributos: No *RequisitePro* e no *CaliberRM* requisitos são organizados por tipo. O usuário pode configurar os tipos de requisitos a utilizar em cada projeto. Artefatos tais como glossário e documento de visão, também são gerenciados e tratados como tipos de requisitos. Em ambas as ferramentas os requisitos são representados através de sentenças de requisitos; é possível a definição dos tipos a utilizar em cada projeto. No *RequisitoPro* é possível representar requisitos como casos de uso. No *CaliberRM* as modificações dos atributos de um determinado tipo são replicadas para todos os projetos. No *DOORS* requisitos são organizados por módulos; o sistema apresenta uma série de módulos pré-definidos e permite que sejam adicionados módulos customizados. No *OpenReq* os requisitos são representados através de sentenças de requisitos. Os requisitos são organizados por tipo e o gerente de cada projeto pode definir quais os tipos de requisitos serão utilizados no projeto.

ii) Cadastro e edição de requisitos: as ferramentas analisadas possibilitam a identificação unívoca dos requisitos; também permitem a importação de requisitos armazenados em documentos do *Microsoft Word*. O *RequisitePro* permite ainda a importação de requisitos de arquivos no formato CSV, e o *DOORS* permite importar requisitos de arquivos de texto ou planilhas. O *CaliberRM* e o DOORS possuem uma interface própria para edição dos requisitos. No *DOORS* é possível realizar a cópia de requisitos entre módulos do projeto. No *OpenReq*, os requisitos são identificados pelo tipo, seguido de uma numeração sequencial por projeto. O *OpenReq* oferece uma interface própria para edição de requisitos.

iii) Controle e histórico de alterações: todas as ferramentas geram um novo registro de histórico a cada alteração nos requisitos (texto ou atributos). O *CaliberRM* utiliza números com casa decimal para representar a versão, que é incrementada quando o texto do requisito é alterado; se apenas atributos são modificados, apenas a parte decimal do número é incrementada. O *RequisitePro* armazena informações de data, hora e autor, além dos valores antigo e atual dos campos modificados. Apenas o *DOORS* não oferece número de versão para requisitos. No *OpenReq* um novo registro de histórico é gerado a cada alteração (texto ou atributos). São armazenadas informações de data, hora e autor, além do valor antigo e atual dos campos modificados. Utiliza números com casa decimal para representar a versão. Quando o texto do requisito é alterado, a unidade do número da versão é incrementada para o próximo número inteiro, caso contrário, apenas a parte decimal do número é incrementada.

iv) *Baseline* e controle de versões: O *RequisitePro* utiliza as funcionalidades do Microsoft Word para realizar o controle e revisão dos requisitos, mas somente é possível visualizar as modificações a partir do histórico de modificações. O *CaliberRM* permite inicializar uma *baseline* a partir de outra já existente, porém não compara o texto dos requisitos e das discussões. Realiza o *lock* de uma *baseline*, mas não permite o *unlock*; também possibilita assinatura eletrônica de usuários para aprovação da *baseline*. *DOORS* permite a duplicação de uma *baseline* existente. Nesta fase de desenvolvimento do *OpenReq* a criação de *baseline* não foi implementada. Os textos dos requisitos são armazenados em um software de controle de versão, o que permite realizar a comparação entre as versões geradas.

v) Fórum de discussões: esta facilidade não é oferecida pelo *DOORS*. *RequisitePro* e *CaliberRM* possuem esta funcionalidade, e o primeiro ainda possibilita conversas privadas e a definição de prioridades para as discussões. Nessas ferramentas

requisitos que possuem uma discussão associada são identificados por uma marca visual. No *OpenReq*, a funcionalidade de fórum de discussões não foi contemplada na fase inicial de desenvolvimento da ferramenta.

vi) Integração com outras ferramentas: O *RequisitePro* possibilita a integração com as demais ferramentas da família *IBM Rational*, permitindo ainda a exportação de informações para o *Microsoft Project*. *CaliberRM* permite a integração com as demais ferramentas da *Borland*, suportando ainda exportação para diversos formatos de arquivos e a geração de documentos para o *Microsoft Word*. Possui integração com as *IDEs Microsoft Visual Studio* e *Eclipse*, além da integração com a ferramenta *Quality Center* da *Mercury*. Por sua vez, o *DOORS* permite integração apenas com algumas ferramentas da *Telelogic*, e oferece a possibilidade de exportar requisitos para arquivos texto, arquivos do pacote *Microsoft Office* e para arquivos *MIF*. No *OpenReq*, inicialmente não foram previstas integrações com outras ferramentas.

vii) Segurança: *RequisitePro* utiliza o conceito de papéis, limitando as ações dos usuários em cada projeto. A base de usuários não é compartilhada, sendo necessário o recadastramento dos usuários a cada novo projeto. Já o *CaliberRM* utiliza grupos associados a perfis de segurança, possibilitando a configuração de uma política abrangente de permissões para cada tipo de requisito. No *DOORS* as permissões são concedidas para usuários ou grupos de usuários. O controle de acesso é realizado em cada um dos itens do sistema (módulos, requisitos, *baselines*, etc.). Oferece um mecanismo de herança que concede as permissões do item "pai" ao novo item criado. O *OpenReq* utiliza o conceito de "perfis", onde são definidas as ações que podem ser efetuadas pelos usuários em cada projeto. A base de usuários é única, e cada usuário pode desempenhar diferentes papéis em diferentes projetos, porém, um único papel em cada projeto. A definição de papel ocorre pela vinculação de um usuário a um perfil em um determinado projeto.

viii) Acesso concorrente: o *RequisitePro* permite o acesso concorrente de múltiplos usuários. Possibilita a abertura de projetos em modo de edição exclusivo ou somente leitura. *CaliberRM* permite o acesso concorrente de múltiplos usuários porém, não permite a abertura de projetos em modo exclusivo de edição ou somente leitura. O *DOORS* realiza o controle de acesso no nível de módulo, e permite a abertura de um módulo em modo de edição compartilhada, exclusiva ou somente leitura.

ix) Acesso Web: o *RequisitePro* permite acesso via internet através do *IBM Rational RequisiteWeb*. O *CaliberRM* oferece as ferramentas *CaliberRM Web* e *CaliberRM WebView*. O *DOORS* oferece o *DOORS/Net* que permite acessar a base de requisitos através de um navegador *Web*. O *OpenReq*, por ser desenvolvido para plataforma *Web*, permite o acesso a todas as suas funcionalidade através dos principais navegadores existentes.

x) Restrições de ambiente: o *RequisitePro* pode ser instalado apenas na plataforma *Windows*. Para a edição de documentos é necessária a instalação do *Microsoft Word*. Os bancos de dados suportados são: *IBM DB2*, *Oracle*, *Microsoft SQL Server* e *Microsoft Access*. O *CaliberRM* executa apenas na plataforma *Windows*. O seu servidor pode ser instalado apenas no *Windows Server* (2000 ou 2003), e o cliente nos *Windows* 2000, 2003, XP ou Vista. No caso do *DOORS*, o *DOORS Database Server* possui suporte aos sistemas operacionais das plataformas *Windows* (2003 Server, XP e Vista), *Sun Solaris*, *HP-UX* e *Linux Red Had Enterprise*. O seu cliente possui suporte apenas a plataforma *Windows*. O *OpenReq* foi desenvolvido em Java, e por isso oferece suporte a todos os sistemas operacionais suportados pela linguagem. Além disso, oferece suporte aos principais bancos de dados existentes.

A Tabela 1 apresenta um resumo da comparação dos aspectos avaliados entre as ferramentas.

Cabe destacar que a avaliação efetuada não teve o objetivo de realizar uma comparação entre as ferramentas, mas sim apoiar o processo de identificar as funcionalidades a serem implementadas no *OpenReq*.

Tabela 1: Avaliação das ferramentas de requisitos

	RequistePro	CaliberRM	DOORS	OpenReq
i	Sim	Sim	Sim	Sim
ii	Sim	Sim	Sim	Sim
iii	Sim	Sim	Sim	Sim
iv	Sim	Sim	Sim	Não
v	Sim	Sim	Não	Não
vi	Sim	Sim	Sim	Não
viii	Sim	Sim	Sim	Sim
ix	Sim	Sim	Sim	Sim
x	Sim	Sim	Sim	Sim

Como pode ser observado na Tabela 1, o *OpenReq* não implementa todos os aspectos elencados, pois na fase inicial do projeto, optou-se em dar prioridade às funcionalidades diretamente relacionadas à atividade de registro dos requisitos, além das funcionalidades básicas para a utilização da ferramenta.

Contudo, a realização desse estudo, ofereceu subsídios para apoiar o grande diferencial desse trabalho, que é a criação de uma ferramenta de gerência de requisitos de código aberto, que busca

oferecer as principais funcionalidades disponíveis nas ferramentas comerciais existentes.

Outro diferencial do *OpenReq*, em relação às ferramentas comercias avaliadas, é o fato do mesmo disponibilizar uma plataforma básica para gerência de projetos de software, a partir da qual, outras características poderão ser agregadas. Isso confere ao *OpenReq* a possibilidade de ampliar suas funcionalidades, com a adição de novos plug-ins que poderão ser desenvolvidos por qualquer pessoa que tenha interesse em expandir o conjunto de funcionalidades disponíveis na ferramenta.

6. Conclusões

A motivação para a realização desse trabalho baseia-se na dificuldade de encontrar ferramentas de gerenciamento de requisitos desenvolvidas em código aberto, apresentando facilidades e flexibilidade para customização e ampliação do seu leque de funcionalidades. Embora a atividade de gerência de requisitos seja de grande importância no processo de desenvolvimento de software, são poucas as ferramentas de código aberto disponíveis para essa finalidade. Tal fato acarreta dificuldades para empresas de pequeno e médio porte carentes da utilização de uma ferramenta de gerência de requisitos de software. Em uma área que busca por qualidade nos produtos de software, é indispensável que possam existir alternativas, que ofereçam às empresas ou instituições com poucas condições de acesso a ferramentas comerciais de alto custo, ferramentas livres que apóiem seus processos de desenvolvimento de software.

Tem-se o pleno conhecimento de que melhorias são necessárias para o amadurecimento e a estabilização da ferramenta proposta, de forma a torná-la mais adequada para utilização, não apenas como ferramenta de gerência de requisitos, como também, para transformá-la em uma solução completa no que diz respeito ao processo de desenvolvimento de software. Neste sentido, algumas ações como o aprimoramento da gerência de módulos com a utilização de OSGi[3] (que visa facilitar a adição e o gerenciamento de *plug-ins*) e a construção de módulo de *Métricas sobre Requisitos de Software*, já estão previstas e se encontram em processo de implementação.

A partir desse trabalho foi possível dar início ao desenvolvimento de um recurso que pode vir a proporcionar, através da adição de outros módulos, tais como: rastreabilidade e validação, uma solução abrangente que possa atender muitas das atividades de processos de desenvolvimento de software.

[3] OSGi é uma plataforma de serviços para o desenvolvimento de aplicativos que vem se tornando o padrão para o desenvolvimento de aplicativos modulares em Java.

7. Bibliografia

[1] Leite, Julio C. S. P. Gerenciando a Qualidade de Software com Base em Requisitos. In: A. R. C. Rocha, J. C. Maldonado e K. C. Weber. *Qualidade de Software Teoria e Prática*. São Paulo: Prentice - Hall, 2001, pp. 238-246.

[2] Sayão, M., A. v. Staa e J. C. S. P. Leite, "Qualidade em Requisitos", Monografia em Ciência da Computação, DI/PUC-Rio, Rio de Janeiro, 2003.

[3] Wiegers, Karl E., *Software Requirements*, Microsoft Press, Redmond, 2003.

[4] M. Hoffmann, N. Kühn e M. Weber. "Requirements for Requirements Management Tools". IEEE International Requirements Engineering Conference, IEEE Computer Society, 2004.

[5] Sommerville, I., *Software Engineering*, Addison Wesley, Harlow, 2004.

[6] IEEE, C. S., "Guide to the Software Engineering Body of Knowledge" - SWEBOK. Los Alamos, IEEE Computer Society, 2004.

[7] Pressman, R. S., *Software Engineering: A Practitioner's Approach*, McGraw-Hill Series in Computer Science, New York, 2001.

[8] Brooks, F. P., *The Mythical Man-Month: Essays on Software Engineering*, Addison-Wesley, 1995.

[9] Dorfman, M.; Thayer, R. H. Standards, *Guidelines and Examples of System and Software Requirements Engineering*, IEEE Computer Society, 1990.

[10] IBM Rational RequisitePro, IBM International Business Machines Corp., acessado em 25 de março de 2009, http://www-306.ibm.com/software/awdtools/reqpro.

[11] Borland CaliberRM, Borland Software Corp., acessado em 10 de março de 2009, http://www.borland.com/us/products/caliber/rm.html.

[12] Telelogic DOORS, Telelogic AB, acessado em 26 de março de 2009, http://www.telelogic.com/products/doors.