Wikis para suporte à documentação de processo de software livre

Adalberto Gonzaga da Silva Filho

Wikis para suporte à documentação de processo de software livre

Adalberto Gonzaga da Silva Filho

Orientadora: **Profa. Dra. Renata Pontin de Mattos Fortes**

Dissertação apresentada ao Instituto de Ciências Matemáticas e de Computação - ICMC-USP, como parte dos requisitos para obtenção do título de Mestre em Ciências - Ciências de Computação e Matemática Computacional. *VERSÃO REVISADA*

USP – São Carlos
Julho de 2011

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados fornecidos pelo(a) autor(a)

Gonzaga da Silva Filho, Adalberto
G586w Wikis para suporte à documentação de processo de software livre / Adalberto Gonzaga da Silva Filho; orientadora Renata Pontin de Mattos Fortes -- São Carlos, 2011.
67 p.

Dissertação (Mestrado - Programa de Pós-Graduação em Ciências de Computação e Matemática Computacional) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2011.

1. processo de software. 2. wiki. 3. design rationale. 4. software de código aberto. I. Pontin de Mattos Fortes, Renata, orient. II. Título.

Agradecimentos

Agradeço primeiramente a Deus, por ter me dado oportunidade e forças para a execução deste trabalho.

Agradeço à toda minha família, e em especial aos meus pais, Neuza e Adalberto, pelo apoio incondicional durante toda a minha vida e também durante a condução deste mestrado. Não tenho palavras para descrever o quanto são importantes para mim.

Muito obrigado, professora Dra. Renata Pontin de Mattos Fortes, pelo apoio, orientação e paciência comigo: tais ingredientes foram imprescindíveis para a conclusão desta pesquisa.

Em especial à minha noiva, Letícia Rossetto: desde a época da graduação sempre me incentivando a estudar. É uma pessoa muito especial, de extrema importância e influência em minha vida.

À empresa Ícaro Technologies, por permitir diversas vezes que minha jornada de trabalho fosse flexibilizada para que eu pudesse me dedicar a este trabalho de mestrado.

A todos os meus amigos e professores pelo bom convívio que tivemos, pelas confraternizações e os ótimos momentos que passamos juntos.

Ao Projeto QualiPSo e ao CNPq, pelo apoio financeiro disponibilizado para a realização desta pesquisa.

Resumo

Um processo de software é definido pelas diversas atividades comumente efetuadas durante o desenvolvimento de software, considerando que tais atividades sejam realizadas sob políticas bem definidas e procedimentos bem estabelecidos. Assim, existem diversos modelos de processo de desenvolvimento de software, que visam garantir a qualidade do produto desenvolvido por meio de tal processo. Identificando a necessidade de processos de software específicos para o desenvolvimento de software livre, devido aos modelos tradicionais não considerarem as características do desenvolvimento deste tipo de software, foi proposto no contexto do Projeto QualiPSo o modelo OMM (*Open Source Maturity Model*). Com o intuito de contribuir com o modelo OMM e a comunidade de software livre, esse trabalho teve como um de seus objetivos a análise do emprego de wikis no processo de desenvolvimento de software. Outro objetivo dessa pesquisa foi o desenvolvimento de uma ferramenta para mensurar o quanto de documentação e registro de *Design Rationale* tem sido realizado em uma wiki.

Palavras chave: processo de software, wiki, design rationale, software de código aberto

Abstract

A software process is defined by several activities commonly performed during the software development, whereas such activities are conducted under well defined policies and and well established procedures. Thus, there are various models of software development process designed to ensure the quality of the product developed through this processes. By identifying the need for software processes specific to the development of free software, due to traditional models do not consider the characteristics of this type of software, the OMM Model (*Open Source Maturity Model*) was proposed in the the QualiPSo's project context. Aiming to contribute with the OMM model and free software community, one of this work objectives was the analysis of using wikis in the software development process. The other objective of this research was to develop a tool to measure how much documentation and registration of *Design Rationale* has been performed in a wiki.

Keywords: software process, wiki, design rationale, open source software

Sumário

Lista de Figuras

Lista de Tabelas

CAPÍTULO

1

Introdução

O desenvolvimento de software possui a característica de ser dividido em várias fases. Desde o planejamento de um software até a sua finalização, são executadas diversas atividades que, quando executadas sob políticas bem definidas e a procedimentos bem estabelecidos, caracterizam o processo de software.

Uma das preocupações das pessoas envolvidas com desenvolvimento de software é poder avaliar o desempenho e o esforço empregados durante a sua construção, com o intuito de garantir que todas as atividades definidas no processo o qual estão seguindo sejam executadas corretamente.

O acompanhamento de um processo de software tem como objetivo garantir que as atividades estão sendo seguidas de acordo com a maneira como foram especificadas, pois entende-se que a qualidade de um produto depende diretamente da qualidade do processo utilizado para construí-lo.

Os modelos de processo de software definem quais são as fases do desenvolvimento, as atividades de cada uma dessas fases e o que se espera como resultado após terem sido completadas. Um dos resultados obtidos com a execução das atividades de um processo de software é a produção de artefatos de software, informações que podem servir de base para o a execução das fases seguintes.

Assim, garantir que os artefatos de software sejam bem gerenciados e que a qualidade dos mesmos possa ser medida, são uma das preocupações que tem demandado grandes esforços e atenção dos envolvidos com o desenvolvimento de software.

1.1 Contexto do trabalho

Considerando os modelos existentes de processo de desenvolvimento de software bem como as atividades contempladas e exigidas em cada um deles, foi constatado que os atuais modelos não suprem as necessidades e características do processo de desenvolvimento de software livre.

Devido ao crescente sucesso dos projetos de software livre nos últimos anos, os mesmos têm sido empregados em diversas áreas, criando a demanda e expectativa por mais qualidade (Wittmann et al., 2010).

Nesse contexto, foi proposto pelo projeto QualiPSo um modelo de desenvolvimento de software livre, denominado OMM (*Open Source Maturity Model*). O principal objetivo é garantir que o desenvolvimento de software livre também siga um processo bem definido e que a sua qualidade possa ser equiparada à qualidade percebida de software desenvolvido da maneira convencional.

O projeto QualiPSo (*Quality Platform for Open Source Software*) agrega interessados de vários países da comunidade européia, além de China e Brasil, visando propor um reconhecimento e disseminação das práticas adotadas nos projetos OSS (*Open Source Softwares*).

Dentre as necessidades do projeto QualiPSo, identificadas durante a sua fase de elaboração, uma delas é a de ferramentas que possam ser utilizadas para a obtenção de métricas acerca do processo a ser seguido (Petrinja et al., 2010b). O intuito é de que tais ferramentas possam prover dados que permitam verificar a aderência ao modelo proposto.

O modelo OMM foi dividido em diversas áreas, denominadas TWEs (*trustworthy elements*), ou, elementos de confiabilidade. A idéia é a de que executando as práticas propostas em cada um dos TWEs e garantindo que as mesmas sejam aderentes ao modelo proposto, se alcance a qualidade no desenvolvimento de software livre (Petrinja et al., 2008; Wittmann et al., 2010)

1.2 Objetivos

Conforme reportado por Raymond (2001), o desenvolvimento de softwares livre tem como característica a descentralização, sendo comum o cenário no qual desenvolvedores em distintas localizações se comuniquem pela Internet. O autor também menciona a utilização de ferramentas web para o apoio ao desenvolvimento de software livre.

Partindo da premissa que os desenvolvedores de software livre estão ambientados com utilização ferramentas web e que as wikis são softwares web que permitem a edição de conteúdo de forma colaborativa, o emprego de wikis para apoiar o processo de software livre foi investigado no presente trabalho.

Portanto, como um dos objetivos da pesquisa reportada na presente dissertação de mestrado, foi proposta a análise do emprego de wikis na atividade de documentação do processo de software, e em específico, o software livre, contribuindo também com as necessidades do projeto QualiPSo.

Outro objetivo deste trabalho foi o desenvolvimento de uma ferramenta que permitisse mensurar o quanto está sendo documentado e produzido de *Design Rationale* no contexto de um projeto de desenvolvimento de software. As discussões realizadas durante o decorrer de um projeto com o intuito de serem tomadas decisões, bem como as razões pelas quais as decisões foram ou não tomadas, ou ainda, informações que foram utilizadas para embasar tal atividade, são denominadas *Design Rationale* (Burge e Brown, 2000).

1.3 Estrutura da dissertação

No próximo capítulo (Capítulo 2) são introduzidas as definições de processo de software, descritos alguns dos modelos de processo existentes e apresentado o modelo OMM. No Capítulo 3 são descritos os conceitos relativos a documentação de softwares bem como apresentada a abordagem do modelo OMM para obtenção de métricas de acompanhamento de processo de software, destacando as atividades de documentação. O Capítulo 4 contém as definições acerca de wikis e nele também são relatados trabalhos relacionados com o presente trabalho. Em seguida, no Capítulo 5, é apresentada a ferramenta desenvolvida no contexto deste trabalho, intitulada WikiRat. Ao final, no Capítulo 6, são apresentadas as conclusões e os resultados obtidos com o presente trabalho, bem como são elencadas as limitações da ferramenta desenvolvida e os possíveis trabalhos futuros derivados desta pesquisa.

CAPÍTULO

2

Processo de software

2.1 Considerações iniciais

Neste capítulo são apresentadas as principais definições de processo de software. Para tanto, são elencados alguns dos processos existentes, com detalhamento das atividades pertencentes a cada um destes processos. Além disso, são apresentados alguns modelos de referência de processo, que são utilizados para avaliação e melhoria da capacidade dos processos.

2.2 Uma visão geral sobre processos de software

Durante o desenvolvimento de um software, vários passos são definidos e percorridos pelos seus desenvolvedores, com o intuito de que o software resultante deste percurso seja de alta qualidade (Pressman, 2006). Esta sequência de passos, ou roteiro, pode ser chamada de processo de software.

De acordo com Fuggetta (2000), um processo de software pode ser definido como *"um conjunto coerente de políticas, estruturas organizacionais, tecnologias, procedimentos e artefatos que são necessários para conceber, desenvolver, entregar e manter um produto de software"*.

Segundo Sommerville (2003), o processo de software é um conjunto de atividades bem definidas que devem ser realizadas no decorrer do ciclo de desenvolvimento de um software, sendo que tais atividades devem ser, basicamente, a especificação de requisitos, o desenvolvimento, a validação e a evolução do mesmo. Ainda segundo as idéias do autor, o processo de software define como são organizadas as atividades de desenvolvimento e sugere a produção de diversos artefatos.

Para uma melhor compreensão das definições de processo de software citadas, é interessante observar algumas das características que são essenciais para a definição de um bom processo de software. Conforme descrito por Pfleeger (2001) é essencial que os processos:

- possuam razões que suportem as suas definições;
- possuam muitas perspectivas, como funcional, comportamental e organizacional, por exemplo;
- descrevam e integrem diferentes níveis de abstração;
- apresentem sintaxe e semântica definidas formalmente;
- sejam reutilizáveis; e
- sejam compreensíveis em abragência e detalhes.

A adoção de um processo de software é importante devido à melhoria na qualidade dos projetos e, consequentemente, nos resultados dos mesmos. Um processo de software promove uma maior disciplina na execução das diversas tarefas que são pertinentes ao desenvolvimento de software (Fuggetta, 2000).

Os processos de software, segundo Kitchenham e Pfleeger (1996), contribuem para a melhoria da capacidade das empresas e corporações em produzirem sistemas computacionais, e ainda, promovem uma maior garantia de que tais sistemas estejam de acordo com as suas especificações.

Segundo as idéias propostas por Tyrrell (2000), a utilização de um processo de software pode contribuir em diversas tarefas e metas almejadas pelas organizações desenvolvedoras de sistemas computacionais, tais como:

- **alcançar efetividade** - um processo bem definido contribui com a produção de um software que contemple a sua corretitude e esteja de acordo com o que o cliente necessita;
- **melhorar a manutenibilidade** - quando um processo é adotado vários documentos são produzidos, e estes, por sua vez, são cruciais quando mudanças se fazem necessárias;
- **fazer estimativas** - processos contribuem com a identificação de quais são as etapas necessárias durante o desenvolvimento de um produto, promovendo melhores estimativas de tempo de produção e recursos necessários;
- **permitir replicação** - quando bem documentado, um processo pode ser reutilizado em outros e futuros projetos;

- **garantir a qualidade** - um processo auxilia no controle e verificação de quais são os requisitos estipulados pelo usuário e quais os artefatos que estão sendo produzidos;
- **melhorar o próprio processo** - a documentação ajuda na futura identificação de possibilidades de melhorias e aperfeiçoamento do processo;
- **entender o projeto** - um processo bem documentado ajuda aos gerentes, desenvolvedores e clientes a entenderem as atividades de desenvolvimento já estabelecidas;

Ainda segundo as idéias de Tyrrell (2000), é muito importante que sejam definidos alguns procedimentos (*guidelines*), ou "meta-processos", que sejam possíveis de serem executados por pessoas, e não máquinas, e que determinem quais são as atividades e artefatos que precisam ser produzidos.

Com o intuito de estabelecer quais atividades, procedimentos e artefatos são importantes de serem produzidos, foram propostos os modelos de processo de software. De acordo com a literatura, em 1956 foi proposto um dos primeiros modelos de processo de software indicado para a produção de softwares em larga escala que consistia de 9 etapas (Benington, 1956) *apud* (Madhavji, 1991). Alguns anos depois, em 1970, Royce (1970) *apud* (Madhavji, 1991) propôs o modelo cascata, uma melhoria do modelo proposto por Benington, que é apresentado na Figura 2.1.

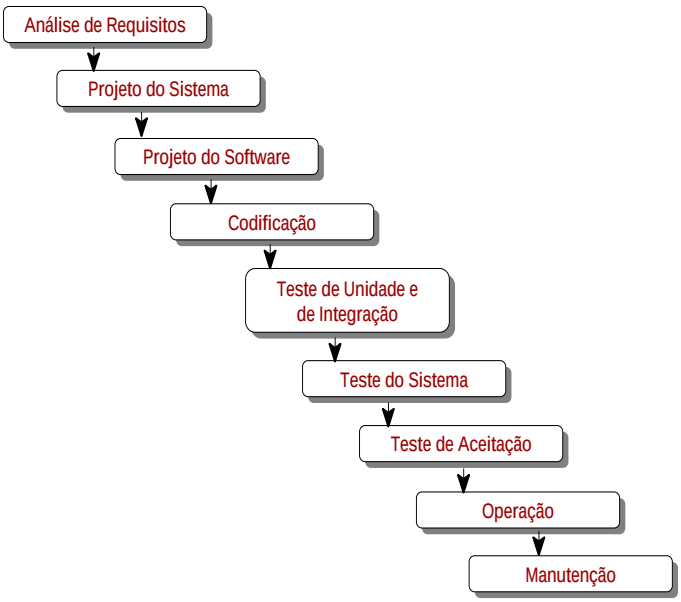

Figura 2.1: Modelo cascata

Posteriormente, em 1986, Boehm (1986) propôs um outro modelo de desenvolvimento de software, o modelo espiral, que sugere várias iterações consecutivas a fim de que ao final do ciclo de iterações o cliente possa avaliar o produto resultante e sugerir correções, adaptações, etc. Na Figura 2.2 é exibida uma representação do modelo espiral.

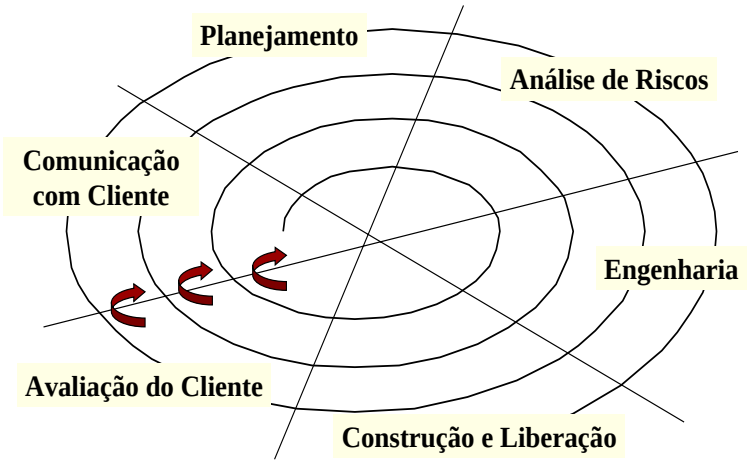

Figura 2.2: O modelo espiral

Outros modelos, mais recentes, surgiram com o intuito de melhorarem os processos de software, de um modo que a maturidade do processo fosse aprimorada gradativamente, como por exemplo o CMM (Paulk et al., 1995), o IDEAL (Gremba e Myers, 1997) e o CMMI (SEI, 2002).

O CMM (*Capability Maturity Model*) é um modelo de processo que possui como estrutura cinco etapas, ou níveis, de capacidade e maturidade, que devem ser alcançados sequencialmente pelas empresas que adotam este modelo de processo. Para atingir o proximo nível a organização deve contemplar todas as atividades obrigatórias que são previstas no nível atual em que a organização se encontra (Paulk et al., 1995). Os cinco níveis do CMM são:

- **Nível 1**: Inicial (*Initial*) - neste nível, a organização da empresa é classificada como "caótica", pois geralmente não há documentação nem controle dos processos, fazendo com que o sucesso dependa do "heroísmo" de alguns funcionários.

- **Nível 2**: Repetível (*Repeatable*) - no nível 2 os processos já são repetíveis e as organizações são caracterizadas como disciplinadas. Há a existência de um processo efetivo com procedimentos e atividades bem fundamentadas, entretanto, o desenvolvimento de novos produtos ainda pode sofrer com estimativas de prazos e custos.

- **Nível 3**: Definido (*Defined*) - neste nível, a corporação já possui um conjunto de processos bem definidos, documentados e padronizados que são utilizados para o desenvolvimento de quaisquer novos produtos. Programas de treinamento asseguram que todos na corporação tenham conhecimento dos processos utilizados.

- **Nível 4**: Gerenciado (*Managed*) - ao atingir o nível 4 as organizações são chamadas de "organizações previsíveis". Tanto o produto como o processo são quantitativamente gerenciados.

- **Nível 5**: Otimizado (*Optimizing*) - no nível 5 as empresas já estão em um processo de melhoria contínua. O processo está constantemente sofrendo melhorias e as lições aprendidas são disseminadas para outros projetos da empresa. Inovações tecnológicas são incorporadas de um modo gerenciável.

Cada um dos níveis do CMM possui um conjunto de áreas chaves de processo, ou KPAs (*Key Process Areas*), que são metas que devem ser cumpridas para que se esteja em conformidade com as atividades de cada nível e, deste modo, estar habilitado a alcançar o próximo nível.

O CMM foi descontinuado[1], devido ao SEI (*Software Engineering Institute*), instituto que o desenvolveu, ter concebido um novo modelo de processo, o CMMI (*Capability Maturity Model Integration*), que é descrito ainda nesta seção.

Um outro modelo para melhoria do processo de software, que foi fundamentado no CMM e desenvolvido pelo mesmo instituto que o desenvolveu o CMMI, é o modelo IDEAL. Este modelo provê uma abordagem utilizável e fácil de entender, pois indica quais são as etapas necessárias para que o programa de melhorias seja efetivo. Seguir as fases, as atividades e os princípios deste modelo tem sido provado como benéfico em muitos esforços de melhoria (Gremba e Myers, 1997).

Utilizando-se de uma estratégia de longo prazo, o modelo provê uma abordagem disciplinada para o programa de melhoria, que consiste em cinco etapas que dão o nome a este modelo:

- **I** - Iniciação (***I**nitiating*): definir um alicerce para um bem sucedido programa de aprimoramento.

- **D** - Diagnóstico (***D**iagnosing*): determinar qual é a posição atual em relação à posição qual deseja se alcançar.

- **E** - Estabelecimento (***E**stablishing*): planejar os detalhes de como os objetivos serão alcançados.

- **A** - Ação (***A**cting*): efetuar o trabalho de acordo como foi planejado.

- **L** - Aprendizado (***L**earning*): aprender com a experiência e aperfeiçoar as habilidades, permitindo a adoção de novas tecnlogias no futuro.

Cada uma das cinco etapas do modelo IDEAL é composta de diversas atividades, assim como os níveis do CMM. Gremba e Myers (1997) reportam que as atividades da fase inicial são cruciais, pois se forem realizadas corretamente e por completo as

[1] http://www.sei.cmu.edu/cmm/

atividades subsequentes poderão fluir sem grandes contratempos. Entretanto, se as atividades forem efetuadas sem o devido comprometimento com as mesmas, os autores alertam que tempo, esforço e outros recursos serão desperdiçados nas etapas seguintes.

Ainda discutindo sobre diferentes modelos de processos de software, um outro modelo amplamente aceito pelas organizações e que também foi fundamentado no CMM é o modelo CMMI (*Capability Maturity Model Integration*). Conforme descrito na literatura (SEI, 2002), o CMMI é uma abordagem para aprimoramento de processo que equipa as organizações com os elementos essenciais de um processo efetivo.

O CMMI pode ser usado para guiar a melhoria de processos para um determinado projeto, para apenas uma divisão, ou ainda, uma organização inteira. Este modelo auxilia na integração de atividades organizacionais que tradicionalmente eram separadas, na definição de objetivos e prioridades para o processo de aprimoramento, bem como fornece uma orientação para processos de qualidade e um ponto de referência para avaliar os atuais processos (SEI, 2002).

Um importante ponto que é destacado pelo CMMI é a *institucionalização*. A institucionalização é conceito relevante na melhoria do processo, pois diz que o processo precisa ser "enraizado" na maneira como o trabalho é feito pela organização, ou seja, que ele faça parte do modo como o trabalho é feito (SEI, 2007).

Assim como o CMM, o CMMI possui seus níveis, porém estes são definidos de acordo com a modalidade adotada pela organização. O CMMI conta com duas modalidades, que são: o modelo contínuo e o modelo em estágios, conforme descrito por Pressman (2006) e SEI (2007).

No modelo contínuo cada área de processo é avaliada com base em metas e práticas específicas, permitindo que a organização tenha diferentes níveis para cada área de processo existente no CMMI, conforme pode ser observado na Figura 2.3 (Pressman, 2006). Ao observar a Figura 2.3 é possível visualizar que uma organização pode possuir diferentes níveis de capacidade para cada área em questão. Por exemplo, conforme pode ser visto na figura em questão, o nível da área PP (Planejamento de projeto) é 3 enquanto o nível da área MA (Medição de análise) é 1.

No modelo contínuo cada área de processo pode ser graduada em até seis níveis e eles são chamados de "níveis de capacitação".

Os seis níveis do modelo contínuo são (SEI, 2007):

- **Nível 0** - Incompleto (*Incomplete*): um processo incompleto é um processo que não é realizado ou é realizado parcialmente. Não existem objetivos gerais para este nível, considerando que não há motivos para institucionalizar um processo parcialmente realizado.

- **Nível 1** - Realizado (*Performed*): o nível de capacitação 1 é caracterizado como um *processo realizado*, ou seja, que satisfaz os objetivos específicos da área de processo, ajudando e habilitando o trabalho a adquirir mais capacitação. Apesar de

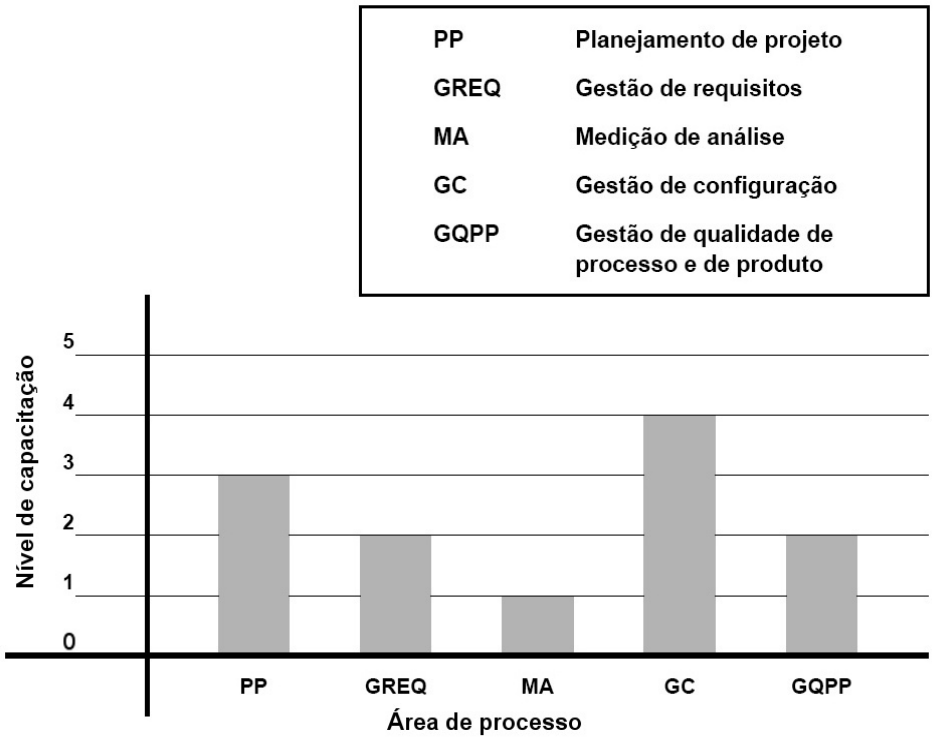

Figura 2.3: Diferentes níveis de capacitação no modelo contínuo para cada área de processo do CMMI. Adaptado de Pressman (2006).

o nível 1 de capacitação resultar em importantes melhorias, elas correm o risco de serem perdidas caso não sejam institucionalizadas. A institucionalização ajuda a garantir que as melhorias sejam mantidas·

- **Nível 2** - Gerido (*Managed*): este nível é tido como resultado de um *processo gerido* (ou *gerenciado*). Um processo gerido é aquele que é realizado (nível 1) e tem a infra-estrutura básica para suportar o processo. O processo é planejado e executado de acordo com as políticas; emprega pessoas capacitadas e que estão equipadas com os recursos adequados para produzir resultados controlados; envolve os desenvolvedores relevantes; é monitorado, controlado e revisto; é avaliado quanto a sua aderência à descrição do processo. A disciplina proveniente deste nível ajuda a garantir que as práticas existentes serão mantidas em tempos de estresse.

- **Nível 3** - Definido (*Defined*): o nível 3 de capacitação é caracterizado como um *processo definido*. Um processo definido é um processo gerenciado (nível 2) que é adaptado a partir do conjunto padrão de processos da organização e de acordo com as recomendações, medidas e outras informações do processo de melhoria da organização. A diferença crucial entre o nível 2 e o 3 é que no nível 3 os padrões, descrições de processos e os procedimentos para um determinado projeto são adaptados para o mesmo a partir de um conjunto padrão de processos da organização, fazendo com que deste modo o processo seja mais consistente.

Outra importante diferença é que no nível 3 os processos são descritos com maior rigor e também são gerenciados mais pró-ativamente.

- **Nível 4** - Quantitativamente gerido (*Quantitatively Managed*): caracterizado como um processo *quantitativamente gerido*, neste nível o processo tem como característica ser definido (nível 3) e controlado por meio de técnicas estatísticas e quantitativas. A qualidade e o desempenho do processo são entendidos em termos estatísticos e são gerenciados ao longo da vida do processo.

- **Nível 5** - Otimizado (*Optimizing*): além de ser um processo quantitativamente gerido (nível 4), no nível 5 o processo já é chamado de *processo otimizado* e é caracterizado por suas melhorias virem do entendimento das causas dos contratempos que são inerentes ao processo. A questão é melhorar o desempenho do processo continuamente por meio de métodos incrementais e inovadores.

Adotando o modelo contínuo, uma organização consegue definir quais áreas ela deseja se aprimorar e qual o nível de capacitação desejado para cada uma das áreas de processo (SEI, 2007).

Já no modelo por estágios é necessário que todas as áreas de processo tenham sido completadas para que se avance para o próximo nível do CMMI. Por exemplo, para que o nível 3 seja alcançado, necessariamente é preciso ter satisfeito todas as atividades previstas no nível 2.

Outra característica do modelo em estágios é que nesse modelo os níveis são chamados de "níveis de maturidade", e são somente 5, iniciando-se do nível 1, onde as organizações são chamadas de "caóticas" e seu sucesso depende do heroísmo dos funcionários, e indo até o nível 5, que por sua vez, possui como principal característica a melhoria contínua do processo. Portanto, os níveis no modo em estágios ficam dispostos da seguinte forma (SEI, 2007):

- **Nível 1** - Inicial (*Initial*).

- **Nível 2** - Gerido (*Managed*).

- **Nível 3** - Definido (*Defined*).

- **Nível 4** - Quantitativamente gerido (*Quantitatively Managed*).

- **Nível 5** - Otimizado (*Optimizing*).

2.3 Processo de software em projetos de software livre

Antes de caracterizar um processo de software livre, faz-se interessante conhecer o conceito de software livre. Segundo a Free Software Foundation (2011), software livre, ou *"free"* (do inglês), é aquele que vem com a permissão de ser utilizado, copiado,

distribuído e modificado por qualquer qualquer pessoa, mesmo que tal distribuição seja feita gratuitamente ou que taxas sejam cobradas para tal tarefa. Permitir que um software seja modificado, em outras palavras, também significa que o código-fonte do mesmo é disponibilizado.

A tarefa de disponibilizar o código-fonte também pode se enquadrar na definição de softwares de código aberto, ou *"open-source"* (do inglês). De acordo com a Open Source Initiative (2006), para que um software seja considerado de código-aberto não basta simplesmente disponibilizar o código-fonte do mesmo, pois ainda é necessário atender a algumas observações.

Deixar o código-fonte acessível, bem publicado e legível (técnicas de *obfuscation*[2] não podem estar aplicadas) são exemplos de tais observações. Estas observações ainda citam o caso onde o autor do código-fonte possa querer preservar os seus direitos autorais sobre o código-fonte original: isto precisa estar explícito na licença que acompanha tal produto ou código-fonte.

Raymond (2001) destacou algumas características inerentes ao processo de produção de software livre:

- **Desenvolvimento descentralizado via Internet:** muitas vezes o desenvolvimento de softwares livre é realizado de forma colaborativa e a Internet tem demonstrado ser um eficiente canal de comunicação para os desenvolvedores, que utilizam sites FTP, repositórios de versões de código e correio eletrônico para viabilizarem suas atividades.

- **Participação dos usuários:** a formação de um grupo de usuários finais que mantêm uma comunicação regular com os desenvolvedores é comum. Estes usuários também comunicam-se entre si e ajudam os desenvolvedores contando-lhes suas experiências e os problemas enfrentados com a utilização do software.

- **Interesse pessoal do autor:** geralmente o próprio autor do software tem uma motivação pessoal em criar e manter o software, pois ele pode ser usuário do mesmo.

- **Desenvolvimento distribuído e ferramentas de comunicação:** diversas ferramentas e serviços são utilizados para suportarem a comunicação entre os desenvolvedores que comumente encontram-se geograficamente distantes entre si.

Scacchi et al. (2006) afirmam que, diferentemente dos modelos tradicionais de desenvolvimento de software, no desenvolvimento de software livre é comum não ter um regime muito formal para o gerenciamente de custos e prazos e, ainda, os artefa-

[2]Técnicas utilizadas com o intuito dificultar o entendimento e a leitura de um código-fonte por seres humanos

tos produzidos costumam ser disponibilizados em repositórios de controle de versões, como o SourceForge[3], o GNU Savannah[4] e o FusionForge[5], por exemplo.

Um importante trabalho reportado na literatura, que foi realizado por Raymond (1999), destacou dois modelos distintos de processos de produção de software livre, que são o modelo Catedral e o modelo Bazaar.

O modelo Catedral é descrito pelo autor como uma forma mais tradicional e conservadora, possuindo longos ciclos de desenvolvimento e um grande tempo entre os lançamentos de novas versões para o público em geral. Um exemplo de projeto que se enquadra neste modelo de desenvolvimento é o GNU Emacs[6] (Raymond, 1999).

O outro modelo abordado por Raymond, o modelo Bazaar, é descrito como um modelo colaborativo que possibilita aos usuários participarem e opinarem livremente sobre o desenvolvimento. Devido ao grande número de pessoas que podem ter acesso aos códigos-fonte, erros podem ser mais rapidamente encontrados, avaliados e corrigidos. Ao contrário do modelo Catedral, neste modelo os lançamentos ocorrem mais frequentemente. Exemplos de projetos mantidos sobre este modelo, que foram citados pelo autor, são o sistema operacional Linux e o Fetchmail[7].

Apesar de na literatura serem reportados diversos projetos de software livre registrados nos repositórios de código-fonte (Scacchi, 2007), na maioria das vezes é somente informado o tipo de licença utilizada no projeto, dando a entender que cada um destes projetos utiliza a sua própria metodologia ou um modelo de desenvolvimento de software livre.

Outros trabalhos que coletaram as características do processo utilizado no desenvolvimento de diversos projetos de software livre (Mockus et al., 2000; Krishnamurthy, 2002; Massey, 2002; Reis, 2003) também levam à conclusão de que na maioria das vezes são utilizadas abordagens similares, porém não muito bem definidas, ou que não estariam seguindo um modelo de desenvolvimento padronizado.

A partir desses fatos citados na literatura, observa-se a relevância de que a idéia do desenvolvimento de um modelo de processo de software específico para software livre tem sido evidenciada diante do atual estado da arte.

2.3.1 Um modelo de maturidade para projetos de código aberto

Devido à crescente popularidade dos produtos desenvolvidos como "de código aberto" e "software livre", referenciados também como **OSS** (*Open Source Softwares*), surgiu a necessidade de serem definidos processos que ajudassem a elevar o nível de confiança em tais produtos (Wittmann e Nambakam, 2009).

[3]http://sourceforge.net/.
[4]http://savannah.gnu.org/
[5]http://fusionforge.org/
[6]http://www.gnu.org/software/emacs/
[7]http://fetchmail.berlios.de/

Como um dos objetivos do projeto QualiPSo, foi proposto a elaboração de um modelo de processo de software para projetos de software livre que tivesse seus princípios fundamentados sobre os modelos CMM e CMMI, levando este novo modelo a ser referenciado como *CMM-Like Model*.

Este modelo auxilia as organizações produtoras de software a definirem um simples, porém efetivo, processo de software que é capaz de agregar melhorias nas atuais práticas de desenvolvimento de tais organizações e também ajuda aos consumidores e usuários finais a avaliarem as mesmas de uma maneira objetiva.

"O modelo de processo, denominado ***QualiPSo OpenSource Maturity Model*** *(que também é abreviado como* ***OMM****), tem como foco a qualidade e o aprimoramento do processo, e indiretamente, a qualidade do produto, que está intrinsicamente ligada a qualidade do processo"* (Wittmann e Nambakam, 2009).

Os autores afirmam que, como uma marca de qualidade os softwares comerciais costumam carregar um "certificado de qualidade", muitas vezes conseguidos por meio da aderência a um processo bem conhecido, como o ISO 9001, por exemplo. Estes modelos de certificação requerem detalhada documentação, uma clara e definida responsabilidade organizacional, entre outros requisitos, que são somente possíveis de serem alcançados por organizações que possuem uma infra-estrutura física.

Estes certificados são difíceis de serem conseguidos para produtos que são desenvolvidos por equipes que estão espalhadas geograficamente ou que apenas possuem contato entre si por meios virtuais. As métricas de garantias básicas de qualidade, como testes exaustivos e testes de campo, não são suficientes para criar confiança nos produtos OSS. Portanto, para que fosse possível incluir a tal "confiança" nos produtos OSS, o projeto propôs o modelo OMM.

O modelo OMM tem como objetivo agregar confiança nos processos de desenvolvimento das organizações que utilizem ou produzam produtos OSS. O intuito é que as empresas passem a utilizar softwares livres na sua cadeia de produção e, ainda, não somente em protótipos, mas também nas suas principais linhas de produtos. *"A confiança no processo de desenvolvimento pode trazer economia nos custos e ajuda a reduzir o tempo de produção"* (Wittmann e Nambakam, 2009).

Para desenvolver o modelo OMM os integrantes e pesquisadores do projeto QualiPSo realizaram uma extensa pesquisa a fim de identificar quais são os principais fatores que podem agregar a confiança em um processo de desenvolvimento de produtos OSS. Como fontes desta pesquisa eles utilizaram:

- A leitura e o estudo da extensa literatura disponível sobre OSS;
- A recuperação e a análise de repositórios públicos de produtos OSS;
- Entrevistas e investigações (*surveys*) com pessoas que trabalham em empresas que utilizam, integram e desenvolvem produtos OSS;

A análise dos dados coletados durante a pesquisa efetuada no contexto do projeto QualiPSo resultou na identificação de "elementos de confiança" ou, como são referenciados no projeto, **"*trustworthy elements*" (TWE)**.

Segundo Petrinja et al. (2008), no contexto do projeto, define-se TWE como um fator ou aspecto específico de um processo de desenvolvimento de software, ou produtos resultantes, que indiretamente influencia a crença e a confiança dos envolvidos na qualidade global deste processo quando é utilizado em um projeto de OSS.

A maioria destes elementos de confiança (ou, elementos confiáveis), identificados a partir das pesquisas realizadas no contexto do projeto, confirmaram evidências apontadas por outras pesquisas sobre OSS publicadas na literatura. Wittmann e Nambakam (2009) afirmam que além dos elementos identificados, a participação de importantes empresas produtoras de software na definição do processo OMM, como HP, Sun e IBM, entre outras, também agrega confiança ao processo de software proposto.

Identificados os TWEs, um das questões que surgiram durante a concepção do novo modelo foi: *"quais destes elementos identificados e quais elementos do CMMI deverão ser incorporados ao modelo OMM ?"*.

Primeiramente, os elementos TWEs foram comparados com as áreas de processo do CMMI. Foi observado que para alguns dos TWEs já existiam áreas do CMMI que cobriam os mesmos aspectos, podendo um único TWE ser coberto por uma ou um conjunto de áreas de processo do CMMI, ou até mesmo, serem cobertos por algumas das menores atividades que estão contidas dentro de uma específica área do CMMI.

Entretanto, alguns dos TWEs que foram identificados por meio das entrevistas e investigações junto à comunidade OSS não puderam ser mapeados para algumas das práticas previstas no CMMI, como por exemplo, o TWE que referencia as licenças de software.

Na Tabela 2.1, são exibidos os mapeamentos entre TWEs e elementos do CMMI, em ordem de relevância, que foram identificados pelos participantes e pesquisadores do projeto Qualipso, visando definir o modelo OMM.

Rel.	**TWE OSS**	**Área de processo do CMMI**
1	Documentação do Produto (PDOC)	Solução técnica (*Technical Solution*) (TS)
2	Popularidade/Reputação do produto (REP)	Não coberto pelo CMMI
3	Uso de padrões bem estabelecidos e divulgados (STD)	Garantia de Qualidade do processo e do produto (*Process and Product QA*) (PPQA)
4	Disponibilidade e uso de um roteiro (*roadmap*) (RDMP)	Requisitos de desenvolvimento e integração do produto (*Requirements Development Product Integration*) (RD)

5	Qualidade do plano de testes (QTP)	Verificação (*Verification*) (VER)
6	Relacionamento entre os envolvidos (*stakeholders*) (STK)	Planejamento de projeto (*Project Planning*) (PP) Gerenciamento de Requisitos (*Requirements Management*) (REQM)
7	Licenses (LCS)	Not covered
8	Ambiente técnico (Ferramentas, Linguagens, etc) (ENV)	Planejamento de Projeto (*Project Planning*) (PP)
9	Número de *commits* e *bug reports* (DFCT)	Não está diretamento associado a um item
10	Manutebilidade e Estabilidade (MST)	Desenvolvimento dos requisitos (*Requirements Development*) (RD) Solução técnica (*Technical Solution*) (TS)
11	Contribuição das organizações para o produto OSS (CONT)	Não é coberto pelo CMMI
12	Resultado da avaliação do produto por empresas terceiras (RASM)	Se for coberto pelo CMMI seria no item Foco do processo organizacional (*Organizational Process Focus*) (OPF)

Tabela 2.1: Mapeamento entre os elementos confiáveis e as áreas de processo do CMMI. Adaptado de Wittmann e Nambakam (2009).

Análogo aos níveis do CMMI e suas áreas de processo, com o propósito de facilitar o desenvolvimento de produtos OSS e também assegurar a qualidade dos mesmos, os TWEs são agrupados em 3 níveis (Wittmann e Nambakam, 2009):

- Os TWEs do **nível básico** são essenciais para o desenvolvimento e o lançamento de produtos OSS confiáveis (de alta qualidade). No nível básico estão inclusos a documentação básica do produto, padrões de desenvolvimento, plano de testes, licença, ambiente técnico, gerenciamento de defeitos (gerenciamento de configuração), manutenibilidade e estabilidade;

- No **nível intermediário** estão incluídos os TWEs referentes à reputação, roteiro (*roadmap*), contribuições de outras organizações, o comprometimento dos envolvidos (*stakeholders*) e as avaliações feitas por terceiros. Estes TWEs possuem como pré-requisito a satisfação dos TWEs do nível básico.

- No último nível do modelo OMM, o **nível avançado**, estão incluídos os TWEs dos níveis básico e intermediário. Em adição a estes TWEs, existem outros que são exclusivos deste nível.

Para completar um nível no modelo OMM é necessário que todos os TWEs inclusos em tal nível e nos níveis inferiores sejam satisfeitos. Na Figura 2.4 é possível observar a disposição dos TWEs em cada um dos níveis do modelo proposto, ficando clara distinção entre os níveis do OMM e a idéia de que as TWEs e práticas do CMMI podem servir de pré-requisito para o próximo nível.

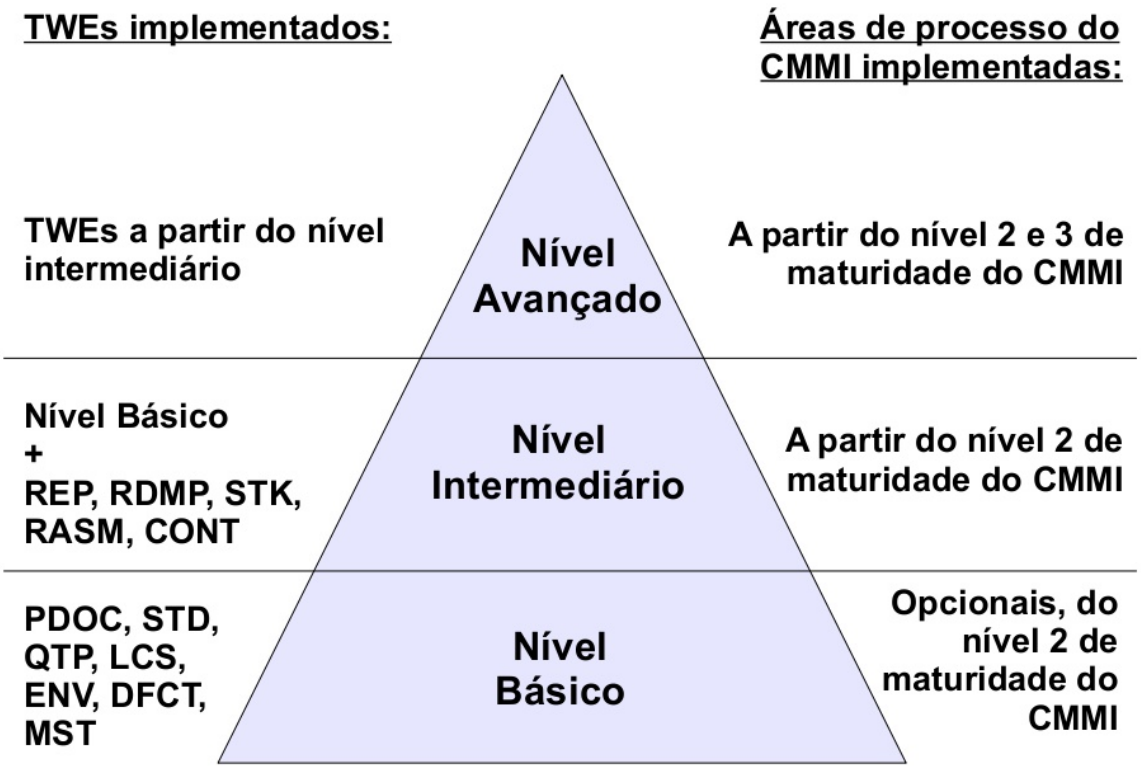

Figura 2.4: Os elementos confiáveis (TWEs) do OMM. Adaptado de Wittmann e Nambakam (2009)

Complementando as informações exibidas na Figura 2.4, os níveis do OMM são definidos do seguinte modo (Wittmann e Nambakam, 2009):

Nível básico: este nível contém os TWEs essenciais que serão necessários para os desenvolvedores construirem produtos OSS confiáveis. Alguns destes elementos estão inclusos nas áreas de processo dos níveis de maturidade 2 e 3 do CMMI (PDOC e STD, por exemplo).

Os elementos obrigatórios para o nível básico são:

- **PDOC**: (OSS TWE) - Documentação do Produto (*Product Documentation*)

- **STD**: (OSS TWE) - Uso de padrões bem estabelecidos e divulgados (*Use of Established and Widespread Standards*)
- **QTP**: (OSS TWE) - Qualidade do Plano de Testes (*Quality of Test Plan*)
- **LCS**: (OSS TWE) - Licenças (*Licenses*)
- **ENV**: (OSS TWE) - Ambiente técnico (Ferramentas, Sistema operacional, linguagem de programação, ambiente de desenvolvimento) (*Technical Environment (Tools, OS, Programming Language, Dev Environment*.))
- **DFCT**: (OSS TWE) - Número de *commits* e *bug reports (Number of Commits and Bug Reports)*
- **MST**: (OSS TWE) - Manutenibilidade e Estabilidade (*Maintainability and Stability*)

Elementos opcionais do nível básico:

- **PP**: (CMMI) - Planejamento de Projeto (*Project Planning*)
- **CM**: (CMMI) - Gerenciamento de Configuração (*Configuration Management*)
- **PPQA**: (CMMI) - Garantia de Qualidade do processo e do produto (*Process and Product Quality Assurance*)
- **PMC**: (CMMI) - Monitoria e Controle do Projeto (*Project Monitoring and Control*)
- **REQM**: (CMMI) - Gerenciamento de Requisitos (*Requirements Management*)
- **SAM**: (CMMI) - Gerenciamento de acordo com o fornecedor(*Supplier Agreement Management*)
- **MA**: (CMMI) - Medição e Análise (*Measurement and Analysis*)

Nível intermediário: o nível intermediário é estabelecido sobre as TWEs do nível básico. Desenvolvedores de OSS devem implementar as TWEs do nível intermediário e as áreas de processo do nível 2 de maturidade do CMMI, que agora são obrigatórias para este nível do OMM.

Elementos do nível intermediário:

- **REP**: (OSS TWE) - Popularidade/Reputação do produto (*Popularity of the SW Product*)
- **RDMP**: (OSS TWE) - Disponibilidade e uso de um roteiro (*roadmap*) (*Availability and Use of a (product) Roadmap*)

- **STK**: (OSS TWE) - Relacionamento entre os envolvidos (*stakeholders*) (*Relationship between Stakeholders (Users, Developers etc)*)
- **RASM**: (OSS TWE) - Resultado da avaliação do produto por empresas terceiras (*Results of Assessment of the Product by 3rd Party Companies*)
- **CONT**: (OSS TWE) - Contribuição das organizações para o produto OSS (*Contribution to OSS Product from SW Companies*)
- **PP**: (CMMI) - Planejamento de Projeto (*Project Planning*)
- **CM**: (CMMI) - Gerenciamento de Configuração (*Configuration Management*)
- **PPQA**: (CMMI) - Garantia de Qualidade do processo e do produto (*Process and Product Quality Assurance*)
- **SAM**: (CMMI) - Gerenciamento de acordo com o fornecedor(*Supplier Agreement Management*)
- **PMC**: (CMMI) - Monitoria e Controle do Projeto (*Project Monitoring and Control*)
- **MA**: (CMMI) - Medição e Análise (*Measurement and Analysis*)
- **REQM**: (CMMI) - Gerenciamento de Requisitos (*Requirements Management*)

Nível avançado: o nível avançado é estabelecido sobre os níveis básico e intermediário do OMM. Nesse nível são incluídos a maioria das áreas de processo do nível 3 de maturidade do CMMI.

Elementos do nível avançado:

- **DAR**: (CMMI) - Análise de decisão e resolução (*Decision Analysis and Resolution*)
- **IPM**: (CMMI) - Gerenciamento integrado de projeto (*Integrated Project Management*)
- **OPD**: (CMMI) - Definição do processo organizacional(*Organizational Process Definition*)
- **OPF**: (CMMI) - Foco do processo organizacional (*Organizational Process Focus*)
- **OT**: (CMMI) - Treino organizacional (*Organizational Training*)
- **PI**: (CMMI) - Integração de produto (*Product Integration*)
- **RD**: (CMMI) - Desenvolvimento de requisitos (*Requirements Development*)
- **RSKM**: (CMMI) - Gerenciamento de riscos (*Risk Management*)

- **TS**: (CMMI) - Soluções técnicas (*Technical Solution*)
- **VAL**: (CMMI) - Validação (*Validation*)
- **VER**: (CMMI) - Verificação (*Verification*)

2.4 Considerações finais

Neste capítulo foram apresentados e definidos os principais conceitos relativos aos processos de software, bem como as atividades envolvidas nos mesmos. Também foram apresentados alguns modelos de maturidade e aprimoramento de processos de software, indicando a necessidade atual do estabelecimento de um modelo específico para o desenvolvimento de produtos OSS, devido aos atuais e já existentes não serem adequados às atividades inerentes de um processo de software livre. Por fim, foi apresentado o modelo de processo de software proposto pelo projeto QualiPSo, denominado OMM.

CAPÍTULO

3

Documentação e Métricas de processo de software livre

3.1 Considerações iniciais

Neste capítulo são apresentados conceitos relativos a atividade de documentação de processo de software bem como os conceitos e definições envolvidos na identificação de métricas para processos.

Também são descritas as necessidades do projeto QualiPSo quanto à disponibilização de suporte automatizado que permita o acompanhamento de processos que sigam o modelo OMM.

3.2 Documentação de processo de software

Durante um processo de software diversas atividades são realizadas e um volume grande de informações sobre tais atividades geralmente é produzido. O registro das informações acerca de um software, segundo Pfleeger (2001), Sommerville (2003) e Pressman (2006), deve ser realizado em documentos que representem o seu processo de construção e que reflitam o seu estado atual.

Ainda segundo os mesmos autores, diversos artefatos podem formar a documentação de um processo de software, e estes devem ser gerenciados, mantidos e atualizados para que a documentação seja considerada de qualidade.

Conforme descrito por Visconti e Cook (2002), a documentação precisa estar sempre refletindo o estado atual de um projeto. As complicações por não se manter uma documentação coerente com o estado em que se encontra um software podem ser grandes,

pois decisões podem ser tomadas com base nas informações presentes na mesma. As decisões realizadas com base em documentos desatualizados podem ser desastrosas e trazerem grande prejuízo.

Considerando que a documentação de um software é formada por diversos artefatos e que a qualidade de tais artefatos define a qualidade de uma documentação, infere-se que os artefatos de software presentes em um processo influenciam substancialmente na qualidade deste processo.

Kruchten (2003) define artefato de software como uma informação presente em um processo de software que pode ter sido produzida, modificada ou, então, apenas utilizada por este processo. Como exemplos de artefatos de software, podem ser citados:

- Contrato de software
- Documento de especificação de requisitos
- Diagramas
- Componentes de software
- Modelos de documentos (*template*)
- Código-fonte

Segundo Sommerville (2003) e Pressman (2006), muitas vezes, em um projeto de software, a documentação pode estar desatualizada, e nos casos mais críticos, pode até ser inexistente. Como maneiras de evitar tais problemas e para apoiar a tarefa de documentação, Cruz (2009) descreve as seguintes recomendações:

- A utilização de documentos dinâmicos e suportados por ferramentas.
- Utilização de modelos de documentos, determinando o que deve ser e como deve ser documentado.
- Definição de procedimentos de documentação devem ser incorporados ao processo de software.
- Manter artefatos de software em documentos eletrônicos com o suporte de ferramentas específicas, ao invés da utilização do papel.

3.3 Métricas para documentação de processo de software livre

Para aprimorar um processo de desenvolvimento de software é necessário, primeiramente, definir quais serão as metas e os critérios pelos quais será avaliado o processo

(Mashiko e Basili, 1997). Uma das metas, por exemplo, pode ser a de saber o quão eficiente é a remoção de *bugs* de um determinado software, ou então, saber se o nível de manutenibilidade de tal software é bom. Após serem definidas as metas e os critérios para avaliar as mesmas, é possível investigar quais são os fatores que influenciam tais critérios.

A abordagem GQM (*Goal-Question-Metric*), em português Meta(Objetivo)-Questão-Métrica, é um modelo para quantificar as metas e questões relevantes em projetos de desenvolvimento de software. "*O GQM pode ser aplicado para medir objetivos específicos considerando o produto, o processo de desenvolvimento ou os recursos necessários para progredir no projeto, por meio de uma genérica, estruturada e coerente metodologia para quantificar os relevantes objetivos do projeto como um todo*" (Mashiko e Basili, 1997).

Conforme descrito por Petrinja et al. (2008), basicamente, o modelo propõe a adoção de um sistema estruturado para definir quais as metas a serem alcançadas pelo projeto a fim de que o mesmo seja efetivo. A definição de tais metas pode ser obtida por meio da identificação dos aspectos relevantes para o projeto.

Após serem definidas as metas, é necessário elaborar alguns questionamentos para que seja possível obter uma perspectiva de cada um dos aspectos definidos previamente. Elaborados os questionamentos, é necessário estabelecer um conjunto de regras para identificar e aplicar as métricas disponíveis, que podem ser extraídas a partir de dados do projeto que sejam adequados para medir quantitativamente uma meta em específico.

Na Figura 3.1 é possível observar o esquema do modelo GQM dividido por níveis.

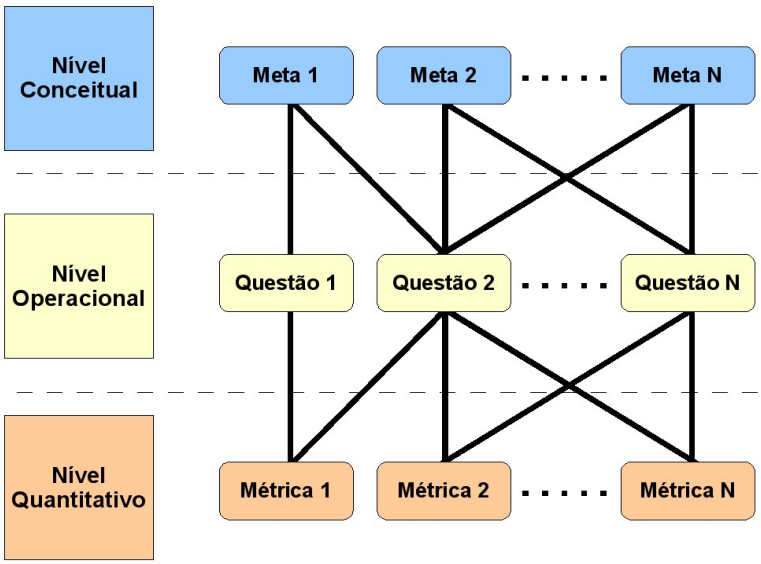

Figura 3.1: Esquema da metodologia GQM. Adaptado de Petrinja et al. (2008)

- **Nível conceitual (Meta):** contém os objetivos para as métricas considerando os produtos, processos ou recursos, especificando as metas das medidas.

- **Nível operacional (Questão):** contém um conjunto de questões usadas para caracterizar como as metas serão alcançadas.

- **Nível quantitativo (Métrica):** contém as métricas que podem ser *subjetivas* (quando dependem da interpretação ou ponto de vista de alguém, como "satisfação" ou "desempenho") ou *objetivas* (quando dependem somente do objeto a ser medido; "número de linhas de código" e "quantidade de pessoas envolvidas no desenvolvimento de um módulo" são exemplos deste tipo de métrica).

Tendo em mente os 3 níveis do GQM descritos, uma abordagem para identificação das métricas é a de, primeiramente, estudar todo o processo com o objetivo de identificar quais são os pontos críticos que podem interferir na sua qualidade. Identificados tais pontos, sendo abordado o nível "Goal", deve-se partir para a elaboração dos questionamentos que podem responder como os níveis de tais pontos críticos podem ser acompanhados. Deste modo, a identificação das métricas pode ser obtida por um plano de medição dos pontos críticos identificados, com o objetivo de posteriormente analisar os dados obtidos da medição e conferir se os mesmos permitem mensurar a qualidade de um fator em questão.

Métricas do OMM

Visando possibilitar que um projeto de software que esteja utilizando o modelo OMM seja acompanhado e avaliado, os pesquisadores e integrantes do projeto QualiPSo utilizaram a abordagem GQM para também identificarem métricas para tal modelo. Com tais métricas identificadas é possível o acompanhamento e avaliação da qualidade de um processo que segue tal modelo (Wittmann e Nambakam, 2009).

As métricas identificadas durante a aplicação da abordagem GQM foram classificadas em 2 tipos: se podem ser automaticamente ou manualmente coletadas.

Wittmann et al. (2010) relacionam no documento que descreve o modelo OMM, intitulado "*OMM: CMM-Like model for OSS*", as métricas do modelo proposto para processo de software livre.

Na Tabela 3.1 são exibidas métricas definidas no escopo do modelo OMM. Devido ao objetivo do presente trabalho ser relacionado com métricas de documentação do processo de software, foram relacionadas algumas das métricas do TWE (PDOC) associado a esta prática. No entanto, outras métricas também disponbilizadas poderiam se adequar ao PDOC.

Métricas
Verificar a presença de documentos que reportem desvios nos planos do projeto, riscos e comprometimentos.
Verificar a presença de relatórios que documentem os resultados das revisões periódicas para o monitoramento do progresso do projeto.

Verificar a presença de documentos que descrevam o ambiente de testes.
Verificar a existência de documentação que defina procedimentos para o planejamento de revisões por pares (*peer review*).
Verificar a existência de documento de especificação de requisitos.
Verificar a existência de documentação que descreva a sequência de integração dos produtos e o rationale associado.
Verificar a existência de documentos que descrevam o ambiente de integrações.
Verificar a existência de documentos ou ferramentas que registrem informações acerca dos riscos envolvidos.
Verificar a existência de documentos ou ferramentas que registrem informações acerca do gerenciamento de riscos.

Tabela 3.1: Algumas das métricas obtidas no processo de aplicação do GQM junto ao modelo OMM (Wittmann et al., 2010).

3.4 Ferramentas para suportar o OMM

Dentre as diversas atividades existentes no contexto do projeto QualiPSo, uma delas é responsável por prover ferramentas que apoiem a implantação do modelo OMM. De acordo com Petrinja et al. (2010a), foi necessário a criação de um "arcabouço de software" (*framework*) que possibilitasse a avaliação automatizada do processo de desenvolvimento de produtos OSS.

Conforme citado, a automação da avaliação do processo é um fator importante dentro dos objetivos do projeto QualiPSo, sendo assim, grandes esforços foram dedicados para suprir essa necessidade. Entretanto, a completa automação não foi possível. Dentre os motivos que não permitiram que a total automação da avaliação do processo fosse alcançada, podem ser citados:

- A natureza subjetiva de algumas métricas, sendo este um dos fatores que mais contribuíram para que a completa automação não fosse possível;
- A complexidade de alguns procedimentos de avaliação e medições, pois algumas métricas incluem vários e diferentes aspectos que são complexos de serem automatizados;
- A variabilidade dos requisitos para diferentes tipos de envolvidos (desenvolvedores, integradores e usuários);
- A não-disponibilidade de algumas ferramentas e o esforço necessário para que fossem implementadas.

Considerando o problema da automação parcial da avaliação do processo, foi elaborada uma lista de possíveis ferramentas que auxiliariam neste processo de avaliação, sendo que, para alguns aspectos, mais de um tipo de ferramenta seria disponibilizado para suportar tal atividade.

Segundo Petrinja et al. (2010a), um dos planos do projeto QualiPSo era disponibilizar um conjunto de funcionalidades automatizadas que dariam a idéia da qualidade do processo de OSS em determinado estágio do projeto, sem que demandasse muito tempo dos usuários no momento de informar os dados para tal sistema de avaliação. Contudo, tal plano não foi possível de ser realizado, pois algumas das funcionalidades requisitadas pelos entrevistados não puderam ser implementadas.

Sendo assim, diante das limitações e dificuldades enfrentadas, foi proposto um conjunto básico de funcionalidades que fossem automatizadas e, para aquelas funcionalidades que não puderam ser automatizadas, foi estabelecido que as mesmas fossem providas por ferramentas que demandassem certo grau de atenção dos usuários quanto à interação com o sistema.

Durante as entrevistas realizadas com os membros da comunidade OSS foram identificadas diversas funcionalidades que são comuns entre as ferramentas existentes de suporte à avaliação da qualidade do processo de software. Um grande percentual das pessoas que foram entrevistadas informaram um mesmo conjunto de funcionalidades que consideram importantes no processo de avaliação, permitindo, deste modo, identificar a relevância de cada característica e funcionalidade requerida.

Apesar de diversas características terem sido identificadas como importantes na avaliação da qualidade do processo OSS, os pesquisadores e integrantes do projeto QualiPSo deram maior atenção àquelas características relacionadas aos elementos confiáveis, os TWEs, conforme reportado por Petrinja et al. (2010a).

Portanto, fica evidenciado que as ferramentas que são capazes de prover dados de métricas referentes aos elementos TWE possuem relevância diferenciada no contexto do projeto QualiPSo.

3.4.1 Disponibilização de uma ferramenta para acompanhamento de documentação de processo de software

Considerando as necessidades apontadas no contexto do projeto QualiPSo, relacionadas ao provisionamento de ferramentas que possibilitem o acompanhamento do processo OMM, foi proposta no escopo do presente trabalho de mestrado a disponibilização de uma ferramenta.

A ferramenta proposta tem como objetivo o acompanhamento do registro de *Design Rationale* e da comunicação entre membros de uma equipe de desenvolvimento de softwares que utilizem uma wiki para executar tais tarefas. O objetivo foi de que, com tal ferramenta, fosse possível contribuir com as métricas do TWE relacionado à documentação do OMM, o PDOC.

Tal ferramenta, denominada WikiRat, foi disponibilizada na forma de uma extensão para uma wiki. Os conceitos e definições envolvidos para o desenvolvimento de tal ferramenta são abordados e descritos nos Capítulos 4 e 5.

3.5 Considerações finais

Neste capítulo foram apresentados os conceitos e definições sobre documentação e métricas para processo de softwares. Também foram abordadas as necessidades do projeto QualiPSo e a maneira pela qual o presente trabalho visa contribuir com este projeto. Os assuntos discutidos neste capítulo contribuíram com a definição da ferramenta proposta, auxiliando a elaboração de bases para a etapa de implementação.

No Capítulo 4 são relacionadas definições acerca dos softwares categorizados como wikis e também são discutidos os trabalhos que se relacionam com a presente dissertação.

CAPÍTULO

4

Wikis para documentação de processo de software

4.1 Considerações iniciais

Wikis são softwares que a cada dia estão se tornando mais comuns em diferentes áreas e aplicações. Atualmente as wikis podem ser encontradas desempenhando o papel de substituir o site de uma empresa, como um manual do usuário acerca de um produto ou então como uma biblioteca ou enciclopédia.

O capítulo atual ressalta a utilização das wikis no contexto do processo de software procurando reportar casos nos quais a utilização de wikis foi considerada positiva. Neste capítulo também são reportadas experiências pessoais do autor da presente dissertação de mestrado durante a utilização de wikis tanto no ambiente acadêmico como também no corporativo.

4.2 Wikis

A origem da palavra wiki vem de um termo Havaiano, que transmite o significado de "veloz", "rápido", ou ainda, "acelerado" (Leuf e Cunningham, 2001). A palavra também é associada com a expressão "**W**hat **I** **K**now **I**s" (Jang e Green, 2006), transmitindo a idéia de informação e conhecimento.

Na computação, wikis são sistemas que permitem, de uma forma fácil e simplificada, a edição de páginas web por quaisquer pessoas ou grupo de usuários, entretanto, a filosofia com qual os usuários lidam ao editar as páginas também é considerada para categorizar os sistemas deste tipo (Louridas, 2006).

Leuf e Cunningham (2001) definem wiki como aplicações web utilizadas para a gerência de conhecimento de uma forma colaborativa.

Considerando tais descrições presentes na literatura, é possível afirmar que uma wiki é um conjunto de páginas web interligadas por meio de hyperlinks, na qual cada uma destas páginas podem ser editadas de uma maneira fácil e por diversas pessoas. As wikis ainda podem manter um registro das alterações, permitindo que versões anteriores de seus conteúdos possam ser restaurados. Normalmente os usuários de uma wiki são incentivados a melhorarem ou produzirem conteúdo nas mesmas.

A grande utilização destes sistemas mostram o quão aceitos eles são (Tetard et al., 2009), demonstrando-se como uma excelente opção, tanto no setor acadêmico como também nas corporações (Majchrzak et al., 2006), de ferramentas de edição colaborativa. O site da Wikipedia[1] é um dos maiores exemplos do sucesso que as wikis alcançaram perante a comunidade de usuários na web, possuindo atualmente[2] mais de 3.000.000 (três milhões) de artigos publicados na forma de uma enciclopédia livre, sendo que todo este conteúdo foi produzido pela comunidade de usuários na web.

A característica de permitir que diversos usuários participem da geração de conteúdo pode ser considerada como principal fator para que as wikis tenham tanto sucesso. A facilidade e velocidade com que os usuários conseguem realizar suas tarefas também fazem com que estes sistemas recebam grande aceitação (Abeti et al., 2009).

Dentre os diversos sistemas do tipo wiki que são disponibilizados atualmente, a maioria deles possuem as seguintes características (Abeti et al., 2009; Tetard et al., 2009):

- edição colaborativa;
- facilidade e simplicidade na edição e produção de conteúdo;
- controle de versão das páginas;
- controle de usuários e grupos de usuários;
- listas de controle de acesso dos usuários;
- comunicação entre usuários (chat, fórum, etc).

Maxwell (2007) avalia as wikis como uma plataforma de publicação que servem como alternativa aos sistemas gerenciadores de conteúdo mais rígidos, especialmente porque as wikis permitem que os editores e colaboradores definam sua própria estrutura de publicação ao invés de terem que utilizar uma estrutura própria da ferramenta ou uma outra que tenha sido definida por uma equipe técnica. Segundo o autor, as wikis praticamente não impõem restrições aos usuários.

[1] http://www.wikipedia.org
[2] Site acessado em Fevereiro de 2011, considerando a língua inglesa

4.3 Utilização de wikis em processo de software

São reportados na literatura diversas ocorrências da utilização de wikis para suportar as atividades inerentes de um processo de software, como a documentação, por exemplo.

Devido a característica de edição colaborativa, a wiki permite aos integrantes de uma equipe de desenvolvimento de software trabalharem juntos sobre um mesmo assunto, promovendo, deste modo, o trabalho em grupo, no qual diversas pessoas se esforçam para atingir um resultado em comum e que beneficie a todos.

Comumente, dentro de uma organização as equipes de trabalho podem enfrentar grandes dificuldades para recuperarem informações úteis acerca de um determinado projeto quando as mesmas possuem o intuito de que tais informações sejam compartilhadas com outras equipes.

Considerando que as organizações podem mais facilmente resolverem os problemas atuais quando elas conseguem recuperar as soluções do passado, e também identificarem quem são os experts no assunto em questão, a empresa alvo do estudo conduzido por Munson (2008) procurou um repositório que pudesse ser explorado pelos seus funcionários.

Para suprir tal necessidade o grupo chegou a conclusão de que utilizar uma wiki seria uma boa alternativa.

Um dos argumentos em pró da adoção de uma wiki para tal tarefa foi que os membros da equipe já estavam acostumados a utilizar uma wiki, pois já conheciam a Wikipedia e sabiam como recuperar conteúdo na mesma. Deste modo, decidiram por utilizar a MediaWiki[3], sistema sob o qual foi desenvolvida a Wikipedia, como ferramenta de publicação de conhecimento.

Durante tal estudo de caso, foi notado que houve maior motivação por parte dos membros das equipes de trabalho nas tarefas de documentarem as informações sobre projetos anteriores.

Também reportando experiências do uso de wikis no processo de software, Dekel (2007) relata que em situações de atividades colaborativas os desenvolvedores tendem a se apoiar mais sobre o suporte que as ferramentas web proveêm, como os sistemas de registro de bugs, por exemplo. Segundo Dekel (2007), as wikis ganharam um espaço nos projetos como uma plataforma para a discussão de idéias, comunicação, documentação e as vezes até substituindo as tradicionais ferramentas de suporte para projetos.

Na pesquisa descrita por Munson (2008), foi observado que usuários da wiki gostavam quando podiam identificar os autores envolvidos na edição de um tópico, pois isto permitia que posteriormente tais autores pudessem ser consultados para maior aprofundamento sobre o assunto, ou então, para a solução de dúvidas acerca de determinado conteúdo. Ainda explorando tal característica, alguns usuários gostavam da

[3]http://www.mediawiki.org/

possibilidade de melhorar o conteúdo produzido por outro autor, ou então, adicionar um link para uma página de autoria própria mas que também abordasse o assunto em questão.

Dekel (2007) observou que a facilidade de utilização das wikis diminui as barreiras que impedem os usuários de participarem na produção de conteúdo. Isto é devido as mesmas possuírem um processo de autenticação e controle de modificações que são facilitados, bem como as notações simples para a formatação. O autor atribui tal facilidade de uso como uma das causas do sucesso das wikis.

Um ponto interessante observado é relativo ao estilo de documentação dos membros da equipe. Alguns membros da equipe preferiam explicitamente descrever todo o conteúdo, temendo que as entradas editadas pelos outros membros poderiam estar incompletas e com conteúdo potencial possivelmente sendo perdido (Munson, 2008). Já outros membros possuíam um estilo diferente de documentação, apenas dando entrada a conteúdos que julgassem ser reutilizáveis, pois alegavam que um conteúdo demasiadamente detalhado poderia intimidar os usuários que simplesmente faziam uma busca na ferramenta.

Posteriormente, foi observado que os usuários naturalmente convergiam para um único estilo conforme o grupo adquiria mais experiência com a wiki.

Ainda considerando as conclusões do trabalho reportado por Munson (2008), quando membros de um projeto encontram informações úteis de um projeto anterior e conseguem aplicar tal conhecimento no trabalho atual, estas pessoas tem a possibilidade de aperfeiçoarem as soluções já existentes, dando maior destaque aos trabalhos atuais.

A maior facilidade ao recuperar conteúdo em uma wiki, quando comparada a um servidor de arquivos, também foi observada. Usuários reclamaram que em um servidor de arquivos a informação é organizada hierarquicamente, não permitindo grandes relações entre diferentes conteúdos, o que facilmente pode ser contornado com a utilização de links entre diversos conteúdos da wiki.

Schugerl et al. (2009) afirma que as wikis podem ser importantes ferramentas nas tarefas de documentação de software atualmente. Os principais motivos apontados pelo autor são a facilidade de uso, a flexibilidade e a natureza colaborativa.

Outros exemplos do uso de wikis durante o processo de software também são reportados na literatura, como o trabalho de Abeti et al. (2009), no qual uma wiki foi utilizada para comunicar o registro dos requisitos de um software durante a sua fase de planejamento.

Também explorando a utilização de wikis no processo de software, Lara (2005) reportou em seu trabalho a utilização de uma wiki para a publicação do conteúdo gerado pelo sistema desenvolvido durante o decorrer de sua pesquisa. O sistema desenvolvido, denominado DocRat, é um sistema web para o registro de *Design Rationale*, que utiliza todo o mecanismo de uma wiki para a publicação do conteúdo gerado (discussões, artefatos, etc.). Diferente do sistema desenvolvido no decorrer da pesquisa

reportada na presente dissertação (descrito no Capítulo 5), o sistema desenvolvido por Lara (2005) possui seu próprio controle de usuários e cadastro de projetos, tendo como foco o registro de *rationale*. Já o sistema descrito no Capítulo 5, tem como propósito quantificar e coletar as categorizações do *rationale* registrado em uma wiki.

Jing e Fan (2008) relatam a utilização de wikis no meio corporativo, no qual foi utilizado uma wiki para o registro de informações que constituiam uma base de conhecimento. Durante o trabalho os autores concluíram que as wikis servem como um modelo de software para a constituição de uma base de conhecimento em corporações modernas.

Conforme pode ser observado, são diversos os trabalhos na literatura reportando a utilização de wikis durante as tarefas mais comuns de um processo de software.

Seja para o registro da comunicação entre diferentes equipes de desenvolvimento, para a documentação de requisitos de software, ou ainda, para a a constituição de repositórios ou bases de conhecimento, são inúmeros os relatos de experiências positivas com tais tipos de software.

Com base na diversidade de trabalhos presentes na literatura e também considerando os aqui mencionados, conclui-se que a utilização de wikis para as tarefas inerentes de um processo de software tem se demonstrado como uma boa opção. Também é possível afirmar que a comunidade científica tem despendido grande atenção na investigação das possíveis utilizações e aplicação destes softwares, caracterizando a relevância do assunto.

4.3.1 Wikis e Design Rationale

As discussões que comumente ocorrem durante o desenvolvimento de um projeto, quando devidamente documentadas, futuramente ajudam a entender as razões por quais determinadas ações foram ou não tomadas. Denomina-se *Design Rationale* as informações que explicam ou descrevem as razões, os motivos, por quais determinadas decisões foram tomadas em um projeto (Burge e Brown, 2000).

Conforme descrito na literatura, a atividade de *Design Rationale* provê mecanismos de comunicação entre os membros de uma equipe de desenvolvimento, permitindo que durante o processo de software seja possível entender quais foram as decisões críticas tomadas, quais alternativas foram investigadas e etc., evitando que posteriormente enganos sejam cometidos (Dix et al., 2003).

Considerando as definições dadas pelos autores supracitados, é válido afirmar que informações decorrentes de um processo de tomadas de decisões de um determinado projeto servem como fonte para a síntese de uma base conhecimento. Tal base de conhecimento pode servir como fonte de consultas para desenvolvedores antes deles iniciarem novas atividades em um projeto, ou então, quando enfrentarem dificuldades perante uma situação em particular.

Os registros de rationale de projetos passados podem proporcionar aos desenvolvedores a facilidade de saberem o que funcionou em determinada situação no passado e poderia funcionar no momento atual, ou então, quais caminhos não devem ser tomados por não terem levado ao sucesso anteriormente.

O desenvolvimento de software pode ser caracterizado por uma série de decisões e implementações que são manifestadas por meio de artefatos de software ou outros artefatos do projeto (Ko et al., 2007).

Segundo LaToza et al. (2007), um dos maiores problemas durante a manutenção de um software é reproduzir as decisões e o *rationale* por trás dos artefatos. Geralmente os desenvolvedores, ou até mesmo toda a equipe, falham durante o registro do rationale.

O entendimento dos artefatos por completo implica em compreender quais foram os julgamentos realizados pelo desenvolvedor durante a sua tomada de decisões e, caso este rationale não tenha sido registrado ou capturado, tal compreensão dos artefatos pode ser prejudicada.

Associando a facilidade de utilização e a popularidade junto aos usuários de ferramentas web e desenvolvedores de software, combinados com a maneira flexível de edição de textos de forma colaborativa, a utilização de wikis para o registro de Design Rationale pode diminuir as lacunas (LaToza et al., 2007; Ko et al., 2007) atualmente existentes no que se diz respeito à documentação de artefatos de software bem como o processo de decisão por trás de tais artefatos.

A facilidade quanto ao uso das wikis pode ser considerada como uma barreira a menos para os integrantes de uma equipe de desenvolvimento de software durante as documentações e registros que normalmente são requeridas em um processo de software. A complexidade de se utilizar um software pode ser determinante para que o mesmo seja utilizado ou não na tarefa de documentação, conforme descrito por Schugerl et al. (2009), no qual os programadores tendem a não utilizar as ferramentas apropriadas para tal tarefa.

A prerrogativa de que sistemas muito complexos podem desmotivar a atividade de documentação de um software, bem como, o registro da comunicação entre vários membros de uma equipe e as decisões realizadas pela mesma, pode ser contornada pelas wikis, devido a baixa complexidade na sua utilização. Esta idéia é reforçada por Maxwell (2007), cujo trabalho cita uma situação na qual usuários preferiram utilizar wikis para registrarem seus textos ao invés dos famosos Sistemas Gerenciadores de Conteúdos (CMS - Content Mananagement System), devido a esses não possuirem um modo mais flexível de edição, impondo aos usuários respeitarem, na maioria das vezes, uma estrutura pré-definida.

Ainda citando a complexidade, um dos pontos negativos apontado por usuários envolvidos no estudo conduzido por Munson (2008) foi relativo ao aprendizado da sintaxe específica das wikis, o que dificultava a incorporação de conteúdos mais complexos. O problema foi solucionado com a utilização de um editor visual para a wiki

(editor WYSIWIG[4]). A maioria das wikis possuem atualmente extensões (*plugins*) que permitem a edição em modo gráfico, não requerindo o aprendizado de uma linguagem ou sintaxe específica.

A diversidade de sistemas wikis atualmente disponíveis de forma gratuita para a utilização atenua o problema da aquisição de softwares por meio de uma equipe de desenvolvedores ou corporação. O site WikiMatrix[5] atualmente lista mais de 120 sistemas deste tipo, sendo a grande maioria gratuita para o download e utilização.

A disponibilização destes sistemas na forma de software de código aberto também possibilita a personalização para que atendam a requisitos específicos de uma determinada compania ou projeto com características peculiares. Estes fatores podem ser considerados como facilitadores da adoção de wikis no processo de software e, em particular, no processo de documentação e registro do rationale.

Devido a possibilidade de edição de textos de forma colaborativa, as wikis também diminuem a complexidade de serem mantidas diversas instâncias de um documento de uma forma sincronizada. Tal característica pode também ser considerada como um motivo adicional para a adoção destes sistemas no apoio da comunicação entre os diversos membros de uma equipe de desenvolvimento de software. A autoria de páginas nas wikis pode ser considerada uma tarefa fácil, o que encoraja os integrantes de uma equipe a documentarem suas soluções.

Ainda assim, em situações nas quais não são possíveis a utilização de páginas web para documentar as características de um software, ou seu processo, as wikis permitem que anexos sejam incluídos em suas páginas, como uma planilha de cálculos, um arquivo de som, ou uma imagem, por exemplo.

Analisando os registros presentes na literatura e nesta seção discutidos, é possível afirmar que existem diversas evidências de que o emprego de wikis para as atividades relacionadas ao registro da documentação de softwares, bem como os seus processos de desenvolvimento, é uma opção viável e que merece a devida atenção, pois pode contribuir com algumas lacunas existentes nestas atividades quando referidas e aplicadas conforme a maneira tradicional.

Nas seções 4.3.2 e 4.3.3 são reportadas experiências do autor da presente dissertação de mestrado quanto a utilização de wikis para o registro de conhecimento/rationale.

4.3.2 Participação em um experimento utilizando wikis para o registro de Design Rationale

No decorrer da elaboração desta dissertação de mestrado, o aluno autor deste trabalho teve a oportunidade de participar de um experimento conduzido por Neto et al. (2009) no Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo (ICMC-USP, São Carlos).

[4]Do inglês, What You See Is What You Get

[5]http://www.wikimatrix.org/ - acessado em Fevereiro de 2011

O aluno de mestrado em questão participou do experimento citado como integrante de um grupo de usuários que utilizaram uma wiki para registrarem o Design Rationale durante o desenvolvimento de uma interface para um software web.

O experimento conduzido por Neto et al. (2009) teve como objetivo a captura de Design Rationale durante o desenvolvimento de interfaces gráficas para um software web que respondesse a comandos de voz (interfaces multimodais). O intuito dos autores do trabalho foi a identificação de padrões de projeto por meio da análise dos registros de rationale realizados pelos desenvolvedores das interfaces em questão.

Os autores do experimento propuseram que, por meio da observação do Design Rationale registrado pelos desenvolvedores, seria possível identificar padrões de desenvolvimento (*design patterns*) que serviriam como um guia para o desenvolvimento de novas interfaces.

O experimento foi conduzido de um modo que, tendo em mãos um conjunto de diretrizes e dicas de desenvolvimento de interfaces multimodais, os programadores deveriam desenvolver um sistema web considerando como modos de interação a interface por voz e a interface gráfica. Todas as comunicações entre membros de equipe, bem como discussões acerca de quais tipos de componentes deveriam ser utilizados para concluirem o desenvolvimento de uma funcionalidade em particular também deveriam ser registrados na wiki utilizada pelo experimento. Para o experimento foi utilizada a MediaWiki como sistema wiki para registro de rationale.

A idéia foi a de que, por meio de observações nos registros de rationale, seria possível identificar um "modelo comportamental" para o desenvolvimento de interfaces multimodais. As restrições do sistema, os possíveis cenários, e as condições sob quais as tarefas seriam executadas seriam o alvo de tais observações.

A atividade de identificar as possíveis tarefas a serem executadas no sistema bem como quais seriam os melhores componentes para a execução de tais tarefas, também eram um dos registros que os autores queriam encontrar por meio da análise do rationale capturado.

Complementando, além do registro durante a fase de desenvolvimento, foi realizado o registro da fase de testes, na qual diferentes equipes de desenvolvedores testavam as interfaces desenvolvidas por outras equipes e registravam suas impressões e avaliações acerca do conteúdo analisado. Tais avaliações também serviriam como base para futura análise, com o intuito de identificarem as melhores abordagens de desenvolvimento.

Neto et al. (2009) concluiram que o experimento obteve resultados expressivos e que a análise do Design Rationale registrado pelos usuários servirá como base para a elaboração de padrões de desenvolvimento de aplicações web com interfaces multimodais.

Ainda assim, os autores identificaram com o experimento sugestões de como melhorar uma interface web multimodal considerando diversos tipos de tarefas a serem executadas.

Considerando as impressões obtidas pelo autor do presente trabalho durante a participação do experimento descrito, os seguintes pontos merecem atenção quanto a utilização de wikis para o registro da comunicação entre membros de uma equipe de desenvolvedores e do rationale durante um processo de desenvolvimento de software:

- **Complexidade de uso**:

  ▷ Praticamente consenso entre todos os membros das equipes de desenvolvedores, utilizar a wiki para a produção de conteúdo foi considerada uma atividade de baixa complexidade.

  ▷ A concorrência durante a edição das páginas web da wiki por membros de uma mesma equipe ocorreu diversas vezes durante o decorrer do experimento. Entretanto, o sistema foi capaz de automaticamente detectar tais situações e bloquear a edição da página em questão até que o editor terminasse a tarefa, evitando assim que conteúdo fosse perdido.

  ▷ A facilidade durante a utilização fazia com que os desenvolvedores documentassem até mesmo pequenas coisas que normalmente não seriam documentadas caso houvesse maior burocracia para tal atividade.

  ▷ O usuários não precisavam se preocupar se membros de outras equipes editariam ou cometessem vandalismo com o conteúdo da própria equipe, pois o sistema provia um mecanismo de restrições e permissões para cada grupo de usuários e recursos.

- **Flexibilidade na edição**:

  ▷ A flexibilidade quanto a edição foi um característica que tanto produziu efeitos positivos como negativos. Devido a ser possível editar utilizando pouquíssimas, ou nenhuma, estruturas, as equipes sentiam-se a vontade para produzirem as páginas da maneira como lhe conviessem. Isto permitiu com que algumas páginas tivessem um layout demasiadamente poluído ou desorganizado.

  ▷ Confirmando as idéias reportadas por Munson (2008), foi possível observar que algumas páginas não possuiam um estilo de organização durante as primeiras edições devido a cada membro redigir o conteúdo como bem quisesse, contudo, nas edições seguintes, naturalmente os usuários identificavam a melhor maneira de organizarem seus textos.

- **Base de conhecimento**:

  ▷ Diversas vezes foram observadas situações em que as equipes consultavam as páginas de outras equipes com o objetivo de encontrarem soluções para prob-

lemas enfrentados em situações específicas. Na maioria das vezes o problema era recorrente e poupou das equipes grande tempo e esforço na busca pela solução.

- **Simplicidade na instalação**:

  ▷ Em poucas horas foi possível fazer o download do sistema, ler o manual de instruções, criar o banco de dados e fazer com o que o sistema funcionasse e já estivesse disponível para a utilização. A instalação da wiki (realizada pelo autor desta dissertação) ocorreu de forma simples e rápida.

  ▷A administração (criação de usuários, espaços para as equipes e atribuição de permissões) da wiki também foi aprendida após poucas utilizações.

Concluindo, a impressão obtida com a utilização de wikis para o registro de Design Rationale e da comunicação entre membros de equipes de um projeto foi bastante positiva. Foi possível identificar diversos pontos benéficos ao se utilizar um ambiente de edição colaborativa para tais atividades.

Os pequenos contratempos podem ser minimizados com a definição de guias e dicas de edição, bem como de comportamento dos integrantes das equipes. Uma política do que deve ser documentado ou, então, qual o nível de detalhamento da informação a ser introduzida na wiki também pode ser bastante positivo. A capacitação e treinamento das equipes também é um ponto interessante, pois com poucas horas de estudo do sistema é possível conhecê-lo o suficiente para evitar futuros contratempos.

4.3.3 Experiência com wikis no ambiente corporativo

Na empresa qual o aluno de mestrado autor desta dissertação atua como Analista de Sistemas, a utilização de wikis é vista como um fato comum no dia-a-dia dos funcionários.

Com diversos funcionários enfrentando diariamente inúmeros desafios, a empresa concluiu que seria comum a situação na qual seus analistas se deparassem com problemas já enfrentados anteriormente por outras pessoas. Portanto, seria interessante a empresa possuir um local onde fosse publicado o conhecimento adquirido pelos seus funcionários, para que diante de um novo projeto, ou uma situação em particular, tal local pudesse ser consultado com o intuito de ser encontrada a solução do problema atual. Esta abordagem permitiria a empresa economizar tempo e, consequentemente, recursos financeiros para a implementação de suas soluções.

Considerando o cenário descrito, a empresa em questão decidiu-se por utilizar uma wiki para a sua base de conhecimento. Como sistema wiki utilizado a empresa adotou a TWiki[6].

Atualmente a wiki já faz parte da cultura da empresa, sendo utilizada para todos os tipos de documentações internas, desde a processos administrativos e procedimen-

[6]http://twiki.org/

tos para solicitação de recursos técnicos bem como a documentação de seus produtos internos e soluções desenvolvidas.

Com base nas observações realizadas durante o cotidiano na empresa e também considerando as políticas internas da mesma, os seguintes pontos merecem destaque:

- **Cultura interna**:

  ▷ A utilização da wiki já faz parte da cultura interna da empresa. Todos os funcionários utilizam a wiki no seu dia-a-dia para realizarem suas tarefas, seja para consultar o número do telefone de um outro funcionário como para descobrir quais são os documentos necessários para realizar um determinado procedimento.

  ▷ Quando os funcionários precisam realizar uma determinada tarefa mas não sabem como fazê-la, um dos primeiros lugares que eles procuram informações é na wiki. É um consenso geral de que a wiki possui a informação procurada.

- **Institucionalização**:

  ▷ Desde os diretores da empresa, como também os estagiários, todos utilizam a wiki. A gerência e diretoria da empresa encoraja o uso da wiki e também o compartilhamento de informações por dela.

  ▷ Novos funcionários e estagiários possuem como uma das suas atividades de integração, logo nos primeiros dias de trabalho, o estudo da wiki e a sua utilização. Todos são instruídos a procurarem as informações na wiki bem como comunicarem o setor responsável quando não concordarem com o conteúdo publicado na mesma.

  ▷ Adicionalmente, todos funcionários da empresa, não importando o setor em que trabalham (Recursos humanos, Departamento pessoal, Administrativo, Financeiro, Marketing e etc.) possuem uma página pessoal na wiki onde diversas informações são compartilhadas. A empresa encoraja a atualização, sempre que pertinente, do conteúdo de tais páginas.

- **Resolução de problemas**:

  ▷ Existe um local pré-definido na wiki que é destinado a documentação de problemas ocorridos e as maneiras como os mesmos foram corrigidos. É comum os funcionários documentarem os problemas logo após terem encontrado a sua solução.

  ▷ As pessoas se sentem bem ao saberem que ao documentarem um problema, e logo em seguida a solução, elas podem estar ajudando os outros companheiros de trabalho. Além disso, a reputação e visibilidade dentro da corporação é maior percebida quando tais atividades são frequentes, pois a corporação, como um todo, rapidamente identifica o espírito colaborativo dessas pessoas.

- **Base de conhecimento**:

  ▷ Considerando o item anterior aliado ao desenvolvimento de produtos e soluções que podem ser reutilizadas em outros projetos, existe um projeto interno de utilização da wiki como base de conhecimento. Produtos e soluções que foram aplicados em um determinado projeto, mas possuem potencial de serem empregados também em outras situações, são minuciosamente documentados e disponibilizados na wiki.

Analisando cada um dos itens descritos nesta seção, pode se afirmar que o sucesso na utilização de wikis em atividades ligadas ao desenvolvimento de softwares pode ser influenciado positivamente por iniciativas que possuem o suporte e incentivos bem definidos. O emprego da wiki na empresa em questão atualmente não tem recebido críticas negativas e a percepção e o consenso geral é de que as informações devem continuar sendo centralizadas e compartilhadas sob a mesma, pois todos se sentem beneficiados com os conteúdos produzidos pelos próprios funcionários.

A facilidade de utilização de wikis também pode ser percebida no ambiente em questão. Diversas pessoas com diferentes formações acadêmicas e também não ligadas a área de computação utilizam a wiki com naturalidade. Devido a cultura presente no ambiente, as pessoas já possuem a wiki como ponto de referência para a busca de informações. Associando a necessidade das pessoas em obterem informações com a possibilidade de encontrá-las na wiki de uma maneira fácil, é comum o pedido de um funcionário ao outro para que compartilhe determinada informação no sistema em questão.

O apoio da diretoria da empresa, reservando recursos e encorajando os funcionários a compartilharem suas informações na wiki tem impulsionado a grande utilização do sistema.

Contudo, o ponto que mais se destaca entre os outros é a utilização da wiki para a base de conhecimento e para a resolução de problemas. A documentação dos motivos pelos quais um problema ocorreu e a maneira como tal foi resolvido tem ajudado os analistas na resolução dos problemas atuais e tem poupado grandes esforços.

Existem conteúdos que são da autoria de pessoas que já não trabalham mais na empresa, entretanto, tais conteúdos ainda sofrem atualizações regularmente. Muitas vezes pode ocorrer de uma pessoa deter o conhecimento sobre determinado artefato porém não possuir tempo ou recursos disponíveis para compartilhar tal conhecimento.

Iniciativas que partem dos mais altos cargos na hierarquia de uma empresa quando somados com a facilidade de utilização de uma ferramenta, tendem a evitar que o conhecimento seja detido somente pelo funcionário, fazendo com que o mesmo passe a ser da empresa, tornando-se um dos maiores ativos e diferenciais da corporação.

4.4 Considerações finais

As wikis são sistemas amplamente difundidos atualmente, tendo sido utilizadas em diferentes áreas e aplicações, sendo que, dentre os motivos do sucesso das mesmas, os que se destacam são a facilidade com que os usuários conseguem produzir conteúdo e a maneira rápida com que fazem isto.

Tanto no setor acadêmico quanto no setor corporativo, as wikis têm se demonstrado como excelente opção para o registro de conhecimento adquirido durante um processo de desenvolvimento de software ou, então, para o registro das comunicações realizadas entre membros de uma equipe.

Quando as principais características de uma wiki são levadas em consideração durante uma comparação com diversos outros sistemas de publicação de conteúdo, elas podem ter a vantagem de serem menos complexas, terem maior flexibilidade quanto a forma de inserção de conteúdo e já serem utilizadas por um grande número de pessoas, sendo a Wikipedia a maior responsável pela difusão destes sistemas dentre os usuários de sistemas web. Aliando todas estas características das wikis com a possibilidade de serem executadas tais tarefas descritas de uma forma colaborativa, o sucesso das mesmas dentre os demais sistemas web para produção e publicação de conteúdo é muito bem justificado.

Contudo, os diversos relatos descritos na literatura bem como o grande número de entidades utilizando tais sistemas, comprovam o quanto relevantes as wikis são dentro da categoria de softwares web para edição de conteúdo de forma colaborativa.

CAPÍTULO

5

WikiRat

5.1 Considerações iniciais

Este capítulo tem como objetivo a apresentação de uma extensão desenvolvida para o registro de Design Rationale em uma wiki. Procurando contribuir com o OMM, provendo meios de acompanhar o quanto está sendo documentado e discutido acerca de um determinado projeto, o software desenvolvido registra as atividades realizadas pelos usuários procurando prover tais dados destas atividades em uma interface web. Neste capítulo são detalhadas as fases do desenvolvimento, a abordagem para a captura das informações bem como as tecnologias estudadas e aplicadas durante o desenvolvimento.

5.2 WikiRat: Um módulo para mensurar o registro de Design Rationale em uma wiki

Procurando uma forma de mensurar o registro de Design Rationale realizado em uma wiki, foi desenvolvido no contexto deste trabalho uma extensão (*plugin*) que permite quantificar o rationale registrado neste referido sistema.

Como sistema wiki adotado para o desenvolvimento de tal extensão foi utilizada a DokuWiki[1]. Conforme a descrição no site do próprio sistema, a DokuWiki é um sistema que segue os padrões web estabelecidos e tem como principal foco a documentação. Possui como usuários alvo os seguintes perfis: times de desenvolvedores, grupos de trabalho e pequenas empresas. Todas as informações são armazenadas em arquivos de texto puro, permitindo que os dados possam ser lidos por outros sistemas

[1] http://www.dokuwiki.org/

e também não requerendo a instalação de um sistema gerenciador de bancos de dados (SGBD).

Dentre os motivos considerados para adoção deste sistema durante o trabalho aqui reportado, os seguintes se destacam:

- A DokuWiki é um sistema disponibilizado como software de código-aberto.
- É pequena e fácil de ser instalada.
- Não requer um banco de dados, permitindo que usuários sem permissões de administrador em um sistema também consigam instalá-la.
- Possui uma comunidade de desenvolvedores ativa.
- O desenvolvimento de plugins é suportado por uma vasta documentação.

Devido a DokuWiki ser um software de código-aberto, a sua alteração para que atenda novas necessidades pode ocorrer de uma forma facilitada. Devido a ser considerada uma wiki simples e de pequeno tamanho, a compreensão de seu funcionamento poderia se dar de uma forma mais rápida, podendo então utilizar tal tempo economizado para o desenvolvimento da extensão proposta. Um outro motivo, se não for o mais importante deles, o desenvolvimento de plugins para a mesma é suportado por uma documentação bem completa, facilitando o trabalho de desenvolvimento por parte de pessoas que nunca interagiram antes com o código do sistema.

Considerando o desenvolvimento de extensões para a DokuWiki, estas podem ser de 5 tipos:

- **Syntax Plugins**: estendem a sintaxe padrão da DokuWiki.
- **Action Plugins**: podem ser usados para estenderem ou substituirem muitos dos comportamentos e operações padrão da DokuWiki, como salvar as páginas, por exemplo.
- **Admin Plugins**: voltados para desenvolvimento de extensões que são voltadas a administradores e gerentes da wiki.
- **Helper Plugins**: podem ser utilizados para prover funcionalidades a outros plugins, evitando que funções sejam re-implementadas diversas vezes.
- **Renderer Plugins**: permitem que sejam criados novos modos de renderização de uma página.

Para o desenvolvimento da extensão em questão foi utilizado o tipo **Action** de plugins para a DokuWiki. A extensão desenvolvida foi projetada para suportar a realização das seguintes atividades (requisitos):

- Após efetuar uma edição na wiki, o usuário deverá ter uma interface para poder categorizar o tipo de rationale adicionado.
- No momento em que for fazer um upload de arquivos para a wiki, o usuário deverá ter mecanismos de categorizar o upload como um artefato de software.
- A extensão deve disponibilizar uma interface para o acompanhamento da atividade dos usuários, na qual devem ser providos dados que transmitam a idéia da quantidade de rationale registrado.

  ▷ Devem sem providos meios de acompanhar as atividades dos usuários durante um determinado período ou intervaldo de datas.

Tendo como base as especificações definidas durante a fase de projeto da ferramenta, cada um dos requisitos identificados foram traduzidos na forma de funcionalidades na ferramenta desenvolvida, procurando sempre manter a coerência dos requisitos com os detalhes de implementação exigidos durante o desenvolvimento de extensões para a DokuWiki.

5.2.1 Categorização do Design Rationale

Conforme descrito na literatura existem diferentes formas de representar o Design Rationale capturado. Shum (1991) descreve que um modelo, dentre outros existentes, para representar o Design Rationale é o PHI (Procedural Hierarchy of Issues).

Nesta forma de representação, um discussão de rationale possui as seguintes características:

- Cada assunto ou tema de uma discussão é chamado de "**Questão**".
- Podem haver um ou mais "**Posicionamentos**" dos membros da equipe em relação a uma Questão em específico.
- Um Posicionamento pode receber um ou mais "**Argumentos**" contra ou a seu favor.

Na Figura 5.1 é exibida uma representação do modelo PHI. Devido as experiências e impressões reportadas na Seção 4.3.2, relativas a falta de estruturação da wiki, que poderia produzir conteúdos desorganizados, foi proposto que para cada Questão de rationale, deveria haver uma página correspondente na wiki. Os argumentos e os posicionamentos deveriam ficar na mesma página que a questão relacionada.

Como em uma wiki nem todas as páginas podem ser relativas ao Design Rationale, foi desenvolvida uma interface que permite aos usuários indicar se a edição atual é relativa a uma questão, posicionamento ou argumentação.

A interface que permite a categorização de uma edição na wiki é inserida pela WikiRat na mesma interface qual o usuário realiza as edições de conteúdo. Deste

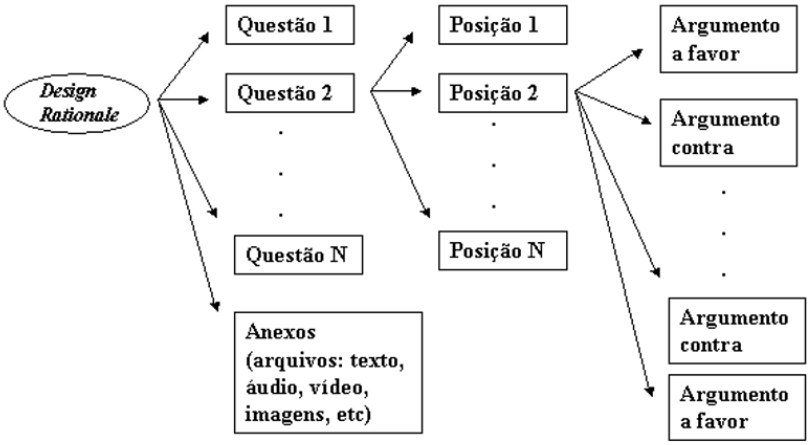

Figura 5.1: Representacão de Design Rationale segundo o modelo PHI

modo, logo após realizar a edição e antes de salvar o conteúdo, o usuário deve especificar na interface em questão o tipo de rationale registrado. A interface inserida pela WikiRat é exibida na Figura 5.2.

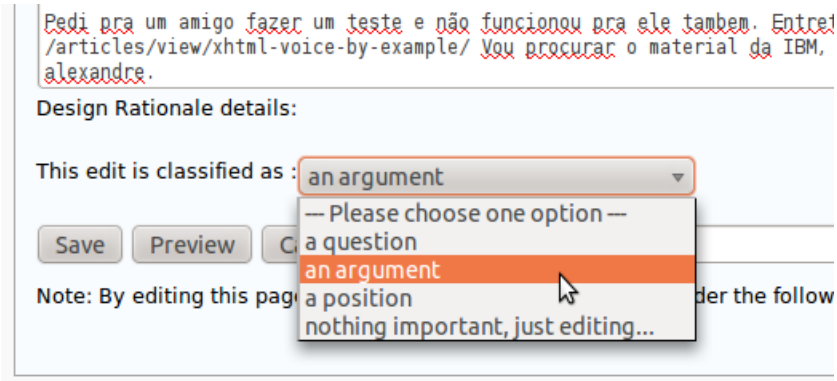

Figura 5.2: Interface inserida na wiki pela WikiRat para classificar o rationale registrado

A inserção da interface em questão é realizada pela extensão WikiRat, que implementa métodos da interface de programação (API) de plugins disponibilizada pela DokuWiki.

Basicamente, a WikiRat espera que um determinado evento ocorra e, durante tal evento, a interpretação do código (linguagem PHP) é interceptada pela WikiRat, que efetua a inserção de um código JavaScript nas páginas da DokuWiki. O evento em questão é relativo ao pré-processamento existente na wiki antes de uma página ser exibida.

O código JavaScript inserido pela WikiRat nas páginas da DokuWiki procura por elementos específicos da interface de edição da wiki e insere novos elementos nesta página. O conteúdo de tais elementos são novamente interceptados pela WikiRat quando o usuário salva a página. A ação de salvar uma página também possui seu evento correspondente, e é desta forma que a WikiRat identifica quando deve inserir o

código JavaScript e quando deve salvar as informações provindas da interface com o usuário.

Procurando também mensurar os dados acerca da produção de artefatos de software produzidos pelos usuários da wiki, foi incluída na WikiRat a funcionalidade que permite ao usuário categorizar um arquivo no momento de seu upload. A funcionalidade em questão foi desenvolvida nos mesmos modelos da funcionalidade representada na Figura 5.2. Na Figura 5.3 é exibida a interface que permite ao usuário indicar qual o tipo de artefato que ele está fazendo o upload.

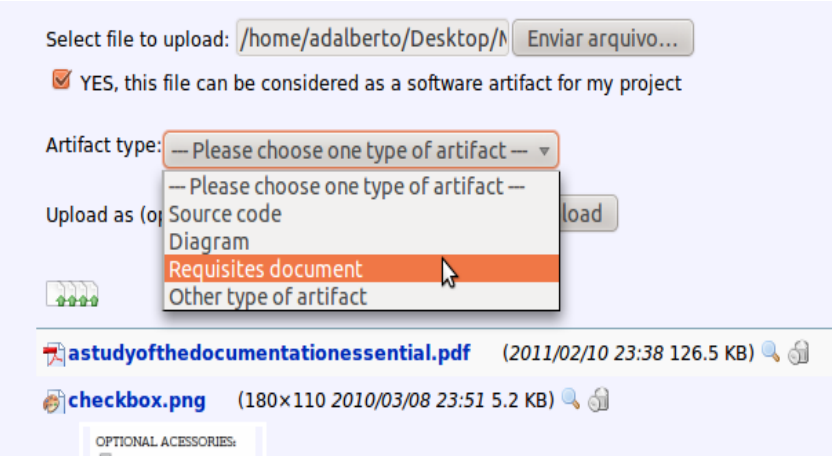

Figura 5.3: Interface para classificar artefatos de software durante um upload de arquivo na wiki

5.3 O armazenamento dos dados na WikiRat

Após terem sido desenvolvidas as funcionalidades que inserem as interfaces para categorização das edições e dos artefatos de software incluídos na wiki, bem como o mecanismo que faz a leitura de tais dados de categorização, foi desenvolvido também o mecanismo que armazena tais dados. Devido a DokuWiki ser um sistema que não requer a instalação de um banco de dados, foi escolhido também para a WikiRat um sistema de armazenamento de dados que não demandasse a instalação de um sistema gerenciador de banco de dados.

Para o armazenamento dos dados da WikiRat foi escolhido o SQLite[2], um banco de dados que armazena os dados em arquivos. Esta abordagem permite que a instalação da ferramenta WikiRat seja menos complexa.

Com o objetivo de manter um número mínimo de informações necessárias para o registro das informações acerca do rationale a ser registrado, foi definido um modelo de ações a serem registradas. As informações basicamente definem:

[2]http://www.sqlite.org/

- **Quem** realizou uma ação no sistema
- **Qual** ação foi realizada
- **Onde** foi realizada uma ação
- **Quando** foi realizada a ação

As seguinte situação podem ser citada como exemplo de evento que provê tais dados:

▷ usuário **leticia1987** registrou um **argumento** na questão da página **logistics_solutions** na data **18/05/2002**.

Deste modo, o diagrama ilustrado na Figura 5.4 demonstra-se ser adequado para o registro de tais informações.

rationale_details	
column_name	data_type
event_id	INT
username	CHAR
action	CHAR
where	CHAR
timestamp	INT

Figura 5.4: Diagrama da tabela do SQLite para armazenar informações de rationale na ferramenta WikiRat

5.4 Visualização das informações coletadas pela WikiRat

Quando usuários da DokuWiki realizam edições e, por meio da interface provida pela WikiRat, informam que tal edição é relacionada a um rationale, podendo este ser uma questão, artefato ou qualquer um dos outros previamente descritos, os dados relativos a este rationale são armazenados na base de dados SQLite, conforme relatado anteriormente.

Procurando um modo de disponibilizar as informações coletadas pela WikiRat, para que pudessem ser consultadas pelos usuários, foi desenvolvida uma interface específica para tal atividade. A interface em questão foi desenvolvida na forma de uma interface, ou página, web, podendo ser acessada pelo mesmo navegador utilizado para acessar a DokuWiki.

Ficando aderente ao modo como foi projetado inicialmente (descrito na Seção 5.2), a WikiRat provê interface para os dados coletados durante a fase de edição. Nesta seção são descritas algumas das funcionalidades de visualização dos dados que são providas pela WikiRat.

Na Figura 5.5 é ilustrada a interface da WikiRat, na qual são exibidos dados acerca da atividade dos membros mais ativos na wiki. Com esta interface é possível identificar os membros de uma equipe que mais estão levantando questões ou gerando conteúdo de uma documentação, produzindo artefatos, etc.

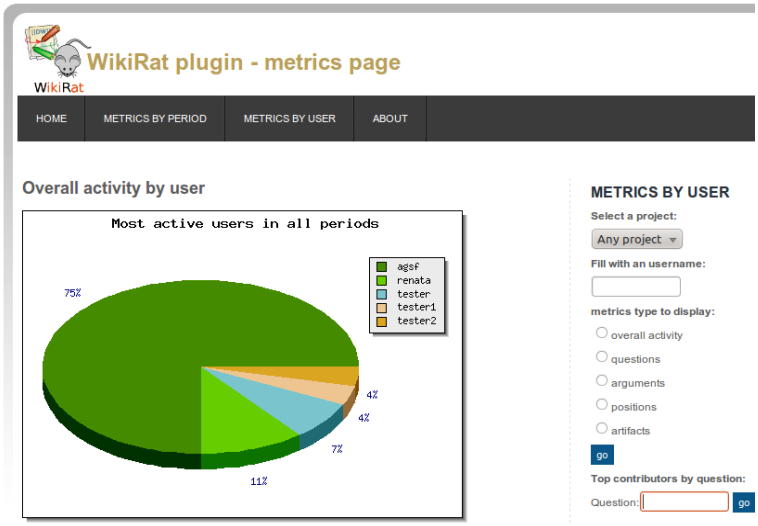

Figura 5.5: Tela na qual são exibidas métricas da atividade dos membros de um projeto na ferramenta WikiRat

Ainda seguindo idéia de identificar os membros mais ativos de uma equipe ou projeto, foi desenvolvida a funcionalidade de identificação dos membros que mais contribuiram para uma questão. São consideradas todas as atividades de rationale para esta métrica, como argumentações, posicionamentos e produção de artefatos. A funcionalidade descrita pode ser visualizada na Figura 5.6.

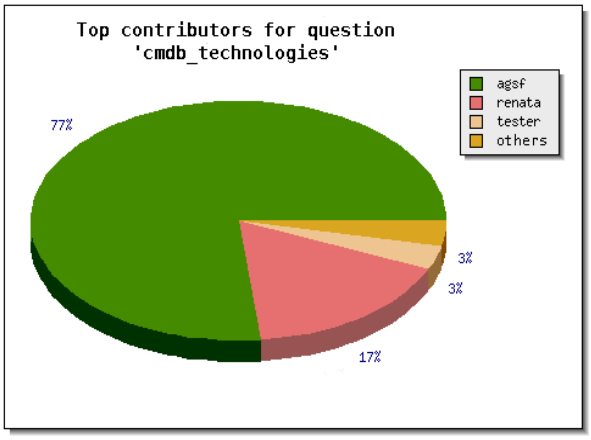

Figura 5.6: Gráfico exibido pela WikiRat: principais contribuidores de uma questão

Fundamentando-se sobre as métricas descritas nas Figuras 5.5 e 5.6, por quais é possível identificar os membros mais ativos e que mais contribuem para a documentação de um projeto, foi desenvolvida também uma funcionalidade que tem o objetivo de identificar o perfil de um usuário.

Alguns usuários podem ter o perfil de levantar questionamentos, outros podem ser mais críticos e somente argumentarem ou, ainda, pode haver o perfil de usuários que preferem se envolver mais com questões de implementação, produzindo artefatos do tipo código-fonte, por exemplo. Na Figura 5.7 são demonstradas as atividades de um usuário e qual a proporção de cada uma destas atividades específicas no total de suas ações na wiki.

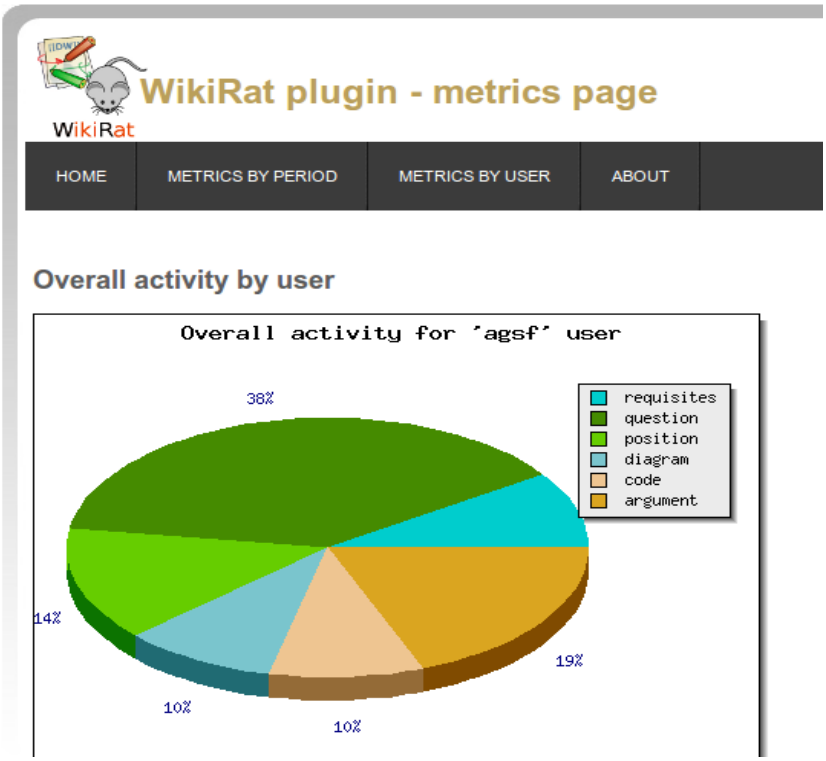

Figura 5.7: Métricas da atividade de um usuário na ferramenta WikiRat

Abordando uma perspectiva relacionada à área de gerência de projetos, também foi incluída na WikiRat uma funcionalidade ligada ao cronograma de atividades. Com a visualização das atividades realizadas em um determinado período, é possível ter a idéia do quanto foi produzido de conteúdo durante determinado tempo, podendo ser identificados picos de atividades e quais foram os membros responsáveis por tais eventos. Na Figura 5.8 é possível visualizar a funcionalidade relativa aos eventos acontecidos em um determinado período.

Considerando as funcionalidades descritas nesta seção e as características do desenvolvimento de extensões para a DokuWiki, bem como o emprego das tecnologias utilizadas, novas funcionalidades podem ser incorporadas a WikiRat, estendendo sua aplicação e a possibilidade de outras medidas também serem capturadas pela ferramenta.

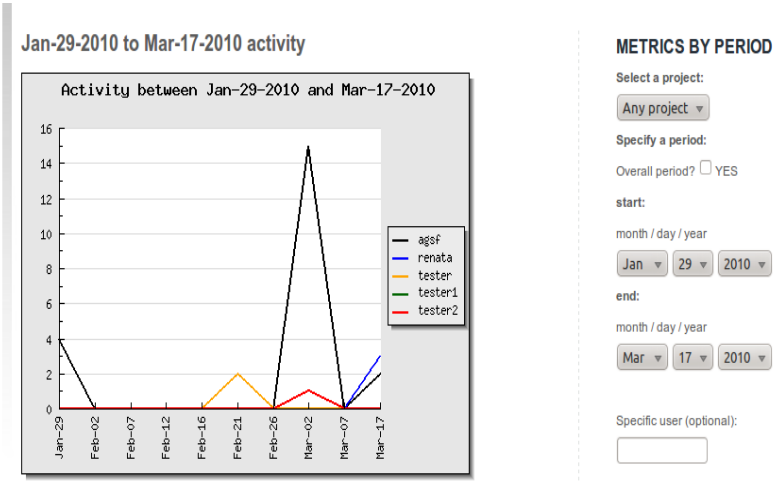

Figura 5.8: Métricas da atividade dos membros de um projeto dentro de um determinado período

A geração dos gráficos na WikiRat

A geração dos gráficos na WikiRat é realizada de forma dinâmica, ou seja, os gráficos são gerados com base nos critérios informados pelos usuários. Considerando tais critérios, é realizada uma consulta na base de dados da ferramenta, a qual retorna um conjunto de dados acerca do rationale registrado. Tendo tais dados acessíveis pelo programa, são utilizados componentes de software que geram os gráficos na forma de uma imagem estática.

Para a geração dos gráficos na WikiRat foi utilizada a biblioteca JpGraph[3]. A biblioteca JpGraph, conforme descrita no seu próprio site, é uma biblioteca escrita em PHP que tem como paradigma de programação orientada a objetos, podendo ser utilizada em qualquer script da linguagem PHP.

Apesar de ser um software proprietário, a JpGraph possui uma versão gratuita, que, ainda assim, oferece diversas opções de geração de gráficos para os programadores. As seguintes características foram consideradas para a adoção desta biblioteca durante o desenvolvimento da WikiRat:

- É compatível com a linguagem PHP, também utilizada na DokuWiki e WikiRat, minimizando, desta forma, questões de compatibilidade.
- Não requer conexão com a Internet para a geração de gráficos, como a API do GoogleChart[4], por exemplo.
- Possui uma documentação extensa e muito bem detalhada, rica em exemplos.
- Pode ser utilizada em softwares com propósitos não-comerciais.
- É gratuita.

[3]http://jpgraph.net/
[4]http://code.google.com/apis/chart/

Desta forma, a utilização de tal biblioteca no contexto do projeto WikiRat deu-se de forma facilitada e de baixa complexidade. A biblioteca teve o funcionamento de acordo com o esperado, demonstrando-se ser uma boa opção para a geração de gráficos em aplicações escritas na linguagem PHP.

5.5 Contribuição com o OMM

Considerando os elementos de confiabilidade (TWEs) descritos no Capítulo 2 e as métricas propostas no modelo OMM, apresentadas no Capítulo 3, a ferramenta WikiRat tem o objetivo de prover métricas que apoiem a avaliação do elemento de confiabilidade denominado PDOC, relativo à documentação.

Conforme pode ser percebido nas descrições das métricas de documentação, as mesmas podem ser muito subjetivas ou não especificarem exatamente qual o tipo de informação deve ser medida.

Deste modo, procurou-se maneiras de identificar tais métricas e o modo como podem ser acompanhadas na WikiRat.

No documento que define o modelo OMM (Wittmann et al., 2010) algumas das métricas relativas à documentação se referem a verificação da existência de determinados tipos de documentos. Devido à WikiRat não ter funcionalidades suficientes para a extração automática de tais informações requisitadas pelo modelo OMM, a extração de parte destas informações pode se dar da seguinte maneira:

- atribuir um nome para determinado documento na wiki:
  Por exemplo, "**test_plan**".
- caso tal documento não exista na wiki, tal métrica é respondida como "não presente ou inexistente".
- caso o documento exista na wiki, a métrica é respondida como "existente".

Ainda assim, no caso da existência de documentos, ou então, refererindo-se a artefatos, ou outros questionamento existentes no OMM, as seguintes questões proveem dados quantitativos acerca da documentação ou rationale:

- Quantas pessoas estão envolvidas na questão X ou no documento Y ?
- Qual o nível de atividade que a questão X ou o documento Y tem demandado?
- Foi feito o upload do documento Y no rationale documentado no item "test_plan"?
- O item "test_plan" demandou quanto de atividade no período X ?

Observando os questionamentos possíveis e os dados providos pela WikiRat, é possível afirmar que tal ferramenta possibilita a extração de métricas que podem ser utilizadas para avaliar algumas das atividades relacionadas à documentação.

Os questionamentos e exemplos dados nesta seção podem ser considerados como sugestões de métricas ou abordagens para obtê-las. Entretanto, caberá ao usuário da ferramenta julgar a forma mais conveniente de serem obtidas informações na WikiRat que o auxiliem a avaliar a atividade de documentação.

5.6 Disponibilização da WikiRat como um software de código-aberto

Visando contribuir com o desenvolvimento de softwares livre e, em específico, o modelo OMM, a WikiRat foi disponibilizada como um software de código-aberto.

As vantanges de se oferecer a WikiRat como um software de código-aberto são relativas ao número de funcionalidades que ela poderia vir a oferecer no futuro e também a manutenção corretiva e evolutiva da mesma.

Por mais que um software seja testado, a possibilidade de serem encontrados defeitos (*bugs*) no mesmo é muito grande, aumentando de forma substancial quando diversos usuários entram em contato com o mesmo.

Partindo deste princípio, a ferramenta foi disponibilizada em um repositório público para que outros desenvolvedores interessados possam contribuir com o desenvolvimento da mesma.

Como repositório para o projeto WikiRat foi escolhido o SourceForge, que além de oferecer um local para o armazenamento do código-fonte, ainda oferece ferramentas para a publicação de documentação, controle de versões e bugs, canal de comunicação com os usuários do projeto hospedado, etc. Segundo dados do próprio site, atualmente[5] são disponibilizados mais de 130 mil projetos de software livre no mesmo.

O código-fonte da WikiRat, conforme descrito, foi disponibilizado no repositório SourceForge. O endereço da página do projeto, na qual também é possível fazer o download do sistema, é o seguinte:

- http://sourceforge.net/projects/wikirat/

Por meio do endereço disponibilizado, além de ser possível fazer o download do sistema para a instalação, também pode ser feito o checkout do repositório SVN do projeto com todo o código-fonte do mesmo. Deste modo, colaboradores que façam melhorias na WikiRat podem reenviar o código atualizado de volta para o repositório, disponibilizando-o novamente para a comunidade. Seguindo esta dinâmica, por meio de diversas iterações, o software tende a ganhar maior maturidade.

Sendo assim, softwares de código-aberto que ainda não estejam terminados ou maduros o suficiente para serem utilizados em "ambiente de produção", podem tornar-se, com o decorrer do tempo, ferramentas de boa qualidade, com a vantagem de poderem ser disponibilizados de maneira gratuita aos seus usuários.

[5]Dados de Fevereiro de 2011

5.7 Considerações finais

Neste capítulo foi apresentada uma extensão para um software do tipo wiki, desenvolvida no contexto do presente trabalho de mestrado. Além da apresentação da ferramenta desenvolvida, denominada WikiRat, foram também descritas as tecnologias empregadas no desenvolvimento da mesma, bem como os motivos de tais tecnologias terem sido escolhidas para o projeto. Foram também apresentadas as principais funcionalidades providas pelo sistema, detalhando-se em cada uma delas, possibilidades de aplicações e a relação com a atividade de desenvolvimento de softwares. Foram também descritas algumas das possibilidades de serem obtidas métricas relativas a documentação por meio da WikiRat.

O desenvolvimento da ferramenta ocorreu de forma aderente à maneira como foi idealizada, entretanto, não foi descartada a possibilidade de a mesma ser melhorada. Como forma de permitir que outros usuários e desenvolvedores da ferramenta contribuam para a evolução da mesma, esta foi disponibilizada na forma de um software de código-aberto.

CAPÍTULO

6

Conclusões e trabalhos futuros

Durante o desenvolvimento de um software, diversas atividades são realizadas até que o mesmo seja finalizado. O planejamento do software, a especificação de seus requisitos, sua implementação, os testes e a manutenção, são exemplos de tais atividades. Quando seguidas sob políticas bem definidas e procedimentos bem estabelecidos, esse conjunto de atividades é denominado processo de software.

Considerando que a qualidade de um produto está relacionada com a qualidade de seu processo de desenvolvimento, conclui-se que a qualidade de um software é dependente da qualidade do processo utilizado para desenvolvê-lo.

Tendo em mente que durante a execução das diversas atividades de um processo de software são produzidos inúmeros documentos e outros artefatos de software, e que os artefatos produzidos em uma fase do desenvolvimento são considerados para a definição das atividades da próxima fase, é possível afirmar que a qualidade dos artefatos produzidos em um processo de software influem diretamente na qualidade do software em questão.

Seguindo a mesma idéia, foi proposto no contexto do projeto QualiPSo um modelo de processo para o desenvolvimento de projetos de software livre, intitulado OMM. O modelo OMM foi proposto devido as características e atividades inerentes de um processo de software livre não serem consideradas pelos já existentes modelos, como o CMM, por exemplo.

Considerando as propostas discutidas no contexto do projeto QualiPSo, foram definidos os elementos de confiabilidade, referenciados no projeto como TWEs (*Trustworthy Elements*). Os TWEs são áreas do modelo OMM consideradas como chave para o sucesso do processo de software, sendo que, para a sua avaliação, foram propostas métricas que permitem mensurar o quanto aderente ao modelo tal processo está.

Ainda no contexto do referido projeto, foi evidenciada a necessidade do provimento de ferramentas que permitam o acompanhamento do processo, por meio da extração de métricas. Tais métricas possibilitariam avaliar a aderência ao modelo.

Dentre os diversos TWEs pertencentes ao modelo OMM, um deles é o elemento PDOC, relativo à documentação de produtos.

Procurando contribuir com o elemento de confiabilidade PDOC do OMM e atendendo a necessidade do projeto quanto ao provimento de ferramentas, o presente trabalho teve como objetivo a análise do emprego de wikis na documentação de processo de software e também a disponibilização de uma extensão para wiki que permitisse mensurar o quanto de *Design Rationale* foi registrado na mesma.

A seguir, nas Seções 6.1, 6.2 e 6.3 são reportadas, respectivamente, as contribuições deste trabalho, as limitações da aplicação desenvolvida e os futuros trabalhos que podem ser derivados da pesquisa aqui reportada.

6.1 Contribuições

Este trabalho de mestrado apresentou as seguintes contribuições para com o meio acadêmico e a comunidade de software livre:

- **Identificação de fatores que suportam o emprego de wikis no processo de software**: foram identificados e reportados neste trabalho fatores que contribuem para o sucesso do emprego de wikis nos processo de desenvolvimento de software. Em especial, foram elencados fatores que contribuem para a utilização de wikis nas atividades relacionadas à comunicação entre membros de uma equipe de desenvolvimento e também para as atividades de documentação, como o registro de *rationale*, por exemplo. Também foram apresentados relatos de experiências do aluno autor deste trabalho de mestrado durante a utilização de wikis, tanto no meio acadêmico como em ambientes corporativos.

- **Auxílio à decisão da aplicação de determinadas tecnologias em projetos de desenvolvimento de softwares**: durante o processo de desenvolvimento da ferramenta disponibilizada por este trabalho de mestrado, foram utilizadas diferentes tecnologias para implementar as funcionalidades providas por tal ferramenta (Seções 5.3 e 5.4). Os motivos pelos quais tais tecnologias foram utilizadas durante o projeto também foram descritos e compartilhados, de um modo breve, servindo de auxílio, ou rationale, para pesquisadores e desenvolvedores de softwares durante um processo de tomada de decisões.

- **Apoio ao modelo OMM de processo de software livre**: por meio das funcionalidades oferecidas pela ferramenta desenvolvida durante estre trabalho, a WikiRat, podem ser obtidos dados acerca da documentação de um processo de software.

Considerando o modelo OMM, e de uma maneira especial, o elemento de confiabilidade relativo a documentação, o PDOC, podem ser extraídas por meio da WikiRat algumas das métricas propostas no contexto do projeto QualiPSo. Deste modo, é possível verificar a aderência ao referido modelo por parte de um processo de software livre.

- **Disponibilização da WikiRat como software de código aberto**: a disponibilização da ferramenta WikiRat como um software de código aberto contribui não somente com o projeto QualiPSo e a comunidade de software livre, mas também com todos os usuários que tenham interesse na utilização da ferramenta, seja para propósitos acadêmicos ou não. Considerando que existem bugs na ferramenta e pontos a serem melhorados, a disponibilização da mesma na forma de software livre permite aos interessados que a mesma seja obtida, a partir do repositório público, e instalada em ambiente local. Após terem sidos identificados os pontos a serem melhorados e tais melhorias terem sido incorporadas à ferramenta, os desenvolvedores podem submeter a ferramenta novamente ao repositório, permitindo que toda a comunidade usufrua de tais melhorias. Após diversas iterações seguindo esta dinâmica, a tendência é que o software amadureça e possa um dia ser utilizado em outros contextos, não somente experimentais. A disponibilização do código-fonte não só beneficia usuários, mas também serve como base de consulta a desenvolvedores e pesquisadores que queiram investigar o desenvolvimento de extensões ou softwares relacionados.

6.2 Limitações da aplicação desenvolvida

Após o desenvolvimento da ferramenta WikiRat foram idenficados os seguintes pontos a serem melhorados na mesma:

- **Identificação manual do tipo de rationale registrado**: durante o processo de edição de um conteúdo, porém logo antes de salvá-lo, para que a ferramenta possa armazenar os dados acerca do rationale registrado pelo usuário, é necessário que o mesmo indique, por meio da interface gráfica, qual o tipo de rationale que foi inserido na wiki. Caso houvesse na interface dispositivos que, antes da edição, fossem acionados de acordo com o rationale a ser registrado, o usuário não precisaria indicar o tipo de rationale. Como exemplo, pode ser vislumbrada a situação na qual o usuário queira adicionar um argumento para uma determinada questão. Ao invés de escolher editar toda a página e em seguida informar o tipo de rationale registrado, poderiam ser disponibilizados na interface botões próprios para a argumentação e o posicionamento. Deste modo, caso o usuário quisesse inserir um argumento, seria aberta uma caixa de texto específica para a argumentação.

- **A WikiRat não funciona corretamente com as versão mais atual da DokuWiki**: durante a implementação da WikiRat foi adotada uma versão da DokuWiki que foi utilizada durante todo o período de desenvolvimento. Naturalmente, a DokuWiki foi atualizada pelo seu grupo de desenvolvedores, porém, por mudanças nas interfaces de programação (API), a WikiRat não manteve completa compatibilidade com a nova versão. Na nova versão[1] da DokuWiki a interface para identificação de artefatos de software não é exibida corretamente em alguns momentos.

- **Em algumas situações não são registradas as páginas que tiveram artefatos de software adicionados**: Por motivos desconhecidos até o momento da redação desta dissertação, os artefatos, em situações também não reproduzidas sistematicamente, não possuem sua localização registrada pela WikiRat após o upload. O tipo de artefato e o usuário que o "produziu" são registrados normalmente.

- **A WikiRat não provê mecanismos para que outras ferramentas consultem seus dados**: a não ser pela consulta direta do arquivo do banco de dados SQLite da ferramenta, ou somente pela passagem de parâmetros e obtenção dos gráficos em forma de imagem, a WikiRat não provê mecanismos para que outras ferramentas obtenham os dados de rationale na forma textual.

6.3 Trabalhos futuros

Como possibilidades de trabalhos futuros derivados desta pesquisa, fora identificados os seguintes itens:

- **Incorporação de semântica no rationale registrado**: da maneira como foi implementada a WikiRat, não é possível a identificação no texto, de uma maneira automatizada, o que é um argumento ou um posicionamento relacionados com uma questão. Os argumentos e posicionamentos quando inseridos no texto devem ser identificados ou explicitados pelo próprio usuário. Caso houvesse no momento do registro do rationale funcionalidades que permitissem a inserção do texto em uma interface separada, para que, automaticamente, nele fossem aplicados identificadores, seria possível separar de uma maneira automatizada um texto apenas descritivo de um outro que possuísse um significado específico dentro da dinâmica do design rationale.

- **Visualização gráfica das questões relacionadas e seus argumentos, posicionamentos e artefatos**: por meio da identificação semântica do texto que representa um rationale, poderia ser construída uma interface que permitisse a visualização

[1] Teste realizado em Fevereiro de 2011

das questões e os seus relacionamentos com usuários envolvidos, artefatos, outras questões, etc. Desta forma, seria possível avaliar o impacto causado nos itens de um projeto por causa do comprometimento de outro.

- **Disponibilização dos dados coletados pela WikiRat por meio de serviços web**: considerando que em um processo de software normalmente as instituições ou grupos de desenvolvimento já possuem um conjunto de ferramentas utilizadas, faz-se interessante a integração das informações coletadas pela WikiRat com outras ferramentas.

- **Realização de um estudo de caso utilizando a WikiRat**: a elaboração de um estudo, envolvendo diversos usuários utilizando a WikiRat, durante o desenvolvimento de um software, poderia fornecer informações que ajudariam a aprimorar a ferramenta e validar a sua eficácia no acompanhamento das atividades de *Design Rationale*.

- **Elaboração de especificação de requisitos para *plugins* semelhantes**: a elaboração da especificação formal dos requisitos necessários para o desenvolvimento de uma ferramenta semelhante é importante para a implementação de uma extensão similar à WikiRat para outras wikis.

- **Integração de wikis com ferramentas de gerenciamento de bugs (*bug tracking*) e repositório de controle de versões**: em um cenário no qual existam itens de configuração de software versionados por um sistema, e os defeitos de tais itens sejam relacionados em uma ferramenta de bug tracking, poderia ser realizada a integração destes 3 sistemas, procurando prover a documentação acerca de tais itens versionados e rastreados pelo sistema de bugs. Os bugs poderiam ser relacionados tanto com os itens de configuração como também com os documentos registrados na wiki.

- **Desenvolvimento de ferramentas para cobertura de outros TWEs do OMM**: com o objetivo de contribuir com o modelo OMM, provendo ferramentas que permitam o acompanhamento de processos de software livre, poderiam ser disponibilizadas outras ferramentas para acompanhamento dos TWEs. Petrinja et al. (2010a) relacionam no documento intitulado "*Specification of the tools to support the CMM-like model for OSS*" uma série de ferramentas que contribuem com o modelo, entretanto, existem áreas ainda não cobertas por ferramentas que automatizem o processo de identificação de métricas.

Referências Bibliográficas

Abeti, L., Ciancarini, P., e Moretti, R. (2009). Wiki-based requirements management for business process reengineering. In *WIKIS4SE '09. ICSE Workshop*, páginas 14–24. 32, 34

Boehm, B. (1986). A spiral model of software development and enhancement. In *SIGSOFT Softw. Eng. Notes*, volume 11, páginas 14–24, New York, NY, USA. ACM. 7

Burge, J. e Brown, D. C. (2000). Reasoning with design rationale. In *Artificial Intelligence in Design '00*, páginas 611–629. Kluwer Academic Publishers. 3, 35

Cruz, J. L. d. (2009). Uma ferramenta para suporte à documentação e rastreabilidade da informação de um processo de teste de software. Dissertação de Mestrado, Universidade Estadual de Campinas. Departamento de Engenharia de Computação e Automação Industrial. 24

Dekel, U. (2007). A framework for studying the use of wikis in knowledge work using client-side access data. In *Proceedings of the 2007 international symposium on Wikis*, WikiSym '07, páginas 25–30, New York, NY, USA. ACM. 33, 34

Dix, A., Finlay, J. E., Abowd, G. D., e Beale, R. (2003). *Human-Computer Interaction (3rd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA. 35

Free Software Foundation (2011). Categories of free and non-free software. http://www.gnu.org/philosophy/categories.html; Visitado em Fevereiro de 2011. 12

Fuggetta, A. (2000). Software process: a roadmap. In *ICSE '00: Proceedings of the Conference on The Future of Software Engineering*, páginas 25–34, New York, NY, USA. ACM. 5, 6

Gremba, J. e Myers, C. (1997). The IDEAL Model: A Practical Guide for Improvement. *Software Engineering Institute (SEI) publication*, (3). http://www.sei.cmu.edu/ideal/ideal.bridge.html. 8, 9

Jang, S. e Green, T. M. (2006). Best practices on delivering a wiki collaborative solution for enterprise applications. In *Collaborative Computing: Networking, Applications and Worksharing, 2006. CollaborateCom 2006. International Conference on*, páginas 1–9. 31

Jing, H. e Fan, Y. (2008). Usability of wiki for knowledge management of knowledge-based enterprises. In *Knowledge Acquisition and Modeling, International Symposium on*, volume 0, páginas 201–205, Los Alamitos, CA, USA. IEEE Computer Society. 35

Kitchenham, B. e Pfleeger, S. L. (1996). Software quality: the elusive target. In *Software, IEEE*, volume 13, páginas 12–21. 6

Ko, A. J., DeLine, R., e Venolia, G. (2007). Information needs in collocated software development teams. In *Proceedings of the 29th international conference on Software Engineering*, ICSE '07, páginas 344–353, Washington, DC, USA. IEEE Computer Society. 36

Krishnamurthy, S. (2002). Cave or community? an empirical examination of 100 mature open source projects. *First Monday*, 7(6). 14

Kruchten, P. (2003). *The Rational Unified Process: An Introduction*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 3 edição. 24

Lara, S. M. A. (2005). Um suporte à captura informal de design rationale. Dissertação de Mestrado, Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo. 34, 35

LaToza, T. D., Garlan, D., Herbsleb, J. D., e Myers, B. A. (2007). Program comprehension as fact finding. In *Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, ESEC-FSE '07, páginas 361–370, New York, NY, USA. ACM. 36

Leuf, B. e Cunningham, W. (2001). *The Wiki way: quick collaboration on the Web*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA. 31

Louridas, P. (2006). Using wikis in software development. *Software, IEEE*, 23(2):88–91. 31

Madhavji, N. H. (1991). The process cycle. *Softw. Eng. J.*, 6(5):234–242. 7

Majchrzak, A., Wagner, C., e Yates, D. (2006). Corporate wiki users: results of a survey. In *WikiSym '06: Proceedings of the 2006 international symposium on Wikis*, páginas 99–104, New York, NY, USA. ACM. 32

Mashiko, Y. e Basili, V. R. (1997). Using the gqm paradigm to investigate influential factors for software process improvement. *J. Syst. Softw.*, 36(1):17–32. 25

Massey, B. (2002). Where do open source requirements come from (and what should we do about it). In *In 2nd Workshop on Open Source Software Engineering. ICSE*. 14

Maxwell, J. W. (2007). Using wiki as a multi-mode publishing platform. In *Proceedings of the 25th annual ACM international conference on Design of communication*, SIGDOC '07, páginas 196–200, New York, NY, USA. ACM. 32, 36

Mockus, A., Fielding, R. T., e Herbsleb, J. (2000). A case study of open source software development: the Apache server. *Software Engineering, 2000. Proceedings of the 2000 International Conference on*, páginas 263–272. 14

Munson, S. A. (2008). Motivating and enabling organizational memory with a workgroup wiki. In *Proceedings of the 4th International Symposium on Wikis*, WikiSym '08, páginas 18:1–18:5, New York, NY, USA. ACM. 33, 34, 36, 39

Neto, A. T., Bittar, T. J., Fortes, R. P. M., e Felizardo, K. (2009). Developing and evaluating web multimodal interfaces - a case study with usability principles. In *Proceedings of the 2009 ACM symposium on Applied Computing*, SAC '09, páginas 116–120, New York, NY, USA. ACM. 37, 38

Open Source Initiative (2006). The open source definition. http://www.opensource.org/docs/osd; Visitado em Fevereiro de 2009. 13

Paulk, M. C., Weber, C. V., Curtis, B., e Chrissis, M. B. (1995). The capability maturity model: Guidelines for improving the software process/cmu/sei. 8

Petrinja, E., Silliti, A., Nambakam, R., Wittmann, M., Oltolina, S., Ruffati, G., Robles, G., e Ortega, F. (2008). Trustworthy elements identified in os processes. working document wd 6.2.1. Technical report, QualiPSo project - http://www.qualipso.org/. 2, 16, 25

Petrinja, E., Sillitti, A., Oltolina, S., Ruffati, G., Ortega, F., e Fortes, R. P. M. (2010a). Specification of the tools to support the cmm-like model for oss. working document wd6.4.1. Technical report, QualiPSo project - http://www.qualipso.org/. 27, 28, 61

Petrinja, E., Sillitti, A., Oltolina, S., Ruffati, G., Ortega, F., e Fortes, R. P. M. (2010b). Tools to support cmm-like model for oss, v4. working document wd6.4.2. Technical report, QualiPSo project - http://www.qualipso.org/. 2

Pfleeger, S. L. (2001). *Software Engineering: Theory and Practice*. 6, 23

Pressman, R. S. (2006). *Engenharia de Software*. McGraw-Hill, 6a. edição. 5, 10, 11, 23, 24

Raymond, E. S. (1999). The cathedral and the bazaar. In *The Cathedral and The Bazaar*, páginas 27–78. O'Reilly and Associates, Sebastopol, 1st edição. 14

Raymond, E. S. (2001). *The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary*. O' Reilly. 2, 13

Reis, C. R. (2003). Caracterização do modelo de processo para projetos de software livre. Dissertação de Mestrado, Instituto de Ciências Matemáticas e de Computação. Universidade de São Paulo. 14

Scacchi, W. (2007). Free/open source software development. In *ESEC-FSE '07: Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering*, páginas 459–468, New York, NY, USA. ACM. 14

Scacchi, W., Feller, J., Fitzgerald, B., Hissam, S., e Lakhani, K. (2006). Understanding Free/Open Source Software Development Processes. *Software Process: Improvement and Practice*, 11(2):95–105. 13

Schugerl, P., Rilling, J., e Charland, P. (2009). Beyond generated software documentation. *Software Maintenance, IEEE International Conference on*, 0:547–550. 34, 36

SEI (2002). Capability maturity model integration, cmu/sei-2002-tr-011 esc-tr-2002-011. Technical report, Software Engineering Institute (SEI). Carnegie Mellon. 8, 10

SEI (2007). Cmmi for acquisition. version 1.2, cmu/sei-2007-tr-017. Technical report, Software Engineering Institute (SEI). Carnegie Mellon. 10, 12

Shum, S. J. (1991). *A Cognitive Analysis of Design Rationale Representation*. Tese de Doutorado, University of York. 47

Sommerville, I. (2003). *Software Engineering*. Addison Wesley. 5, 23, 24

Tetard, F., Patokorpi, E., e Packalen, K. (2009). Using wikis to support constructivist learning: A case study in university education settings. In *System Sciences, 2009. HICSS '09. 42nd Hawaii International Conference on*, páginas 1–10. 32

Tyrrell, S. (2000). The many dimensions of the software process. *Crossroads*, 6(4):22–26. 6, 7

Visconti, M. e Cook, C. R. (2002). An overview of industrial software documentation practice. In *Proceedings of the XII International Conference of the Chilean Computer Science Society*, SCCC '02, páginas 179–, Washington, DC, USA. IEEE Computer Society. 23

Wittmann, M. e Nambakam, R. (2009). Cmm-like model for oss. working document wd6.3.1. Technical report, QualiPSo project - http://www.qualipso.org/. 14, 15, 16, 17, 18, 26

Wittmann, M., Nambakam, R., Maldonado, J. C., Malheiros, V., Oltolina, S., Ortega, F., Petrinja, E., Ruffati, G., e Sillitti, A. (2010). Omm: Cmm-like model for oss (ver.1.1). Technical report, QualiPSo project - http://www.qualipso.org/. 2, 26, 27, 54