Avaliação de Ferramentas para Controle Automatizado de Versões de Artefatos de Requisitos de Software

Susana Brunoro C. Oliveira[1], Astério K. Tanaka[2], Dalessandro S. Vianna[1]
[1]*susanabrunoro@yahoo.com.br*
dalessandro@ucam-campos.br
Instituto Universitário Candido Mendes – Campos
Rua Anita Peçanha, 100 - Parque São Caetano - Campos dos Goytacazes, RJ

[2]*tanaka@uniriotec.br*
Universidade Federal do Estado do Rio de Janeiro (UNIRIO)
Centro de Ciências Exatas e Tecnologia
Avenida Pasteur 458 – Urca - Rio de Janeiro, RJ

Abstract

The control of all the artifacts generated by a development team tends to be a great challenge in software development process. Such a need starts in the moment that the project begins, that is, in the preliminary requirements elicitation phase. From this moment on, any change not duly registered can reflect in failure at the end of the project. Having tools that allow the control of the artifacts generated during the development process is, thus, very important for the development teams. This presents an evaluation of the usage of tools for the development of requirements artifacts and for control of changes in the project, from the preliminary requirements elicitation phase.

1. Introdução

Segundo Kruchten [13], agora que todos os sistemas simples já foram desenvolvidos, gerenciar a complexidade de grandes sistemas tem se tornado o principal interesse das empresas desenvolvedoras de software. O dinamismo com que essas empresas têm que operar aliado à alta competitividade no setor tem feito com que os desenvolvedores de software busquem maior eficácia no processo de produção de seus produtos.

Com o intuito de aperfeiçoar o desenvolvimento de software e obter produtos com os níveis desejáveis de qualidade, dentro do cronograma e orçamento propostos, a última década assistiu a uma mudança de enfoque com relação à garantia da qualidade. Tem-se, então, uma nova abordagem na qual o foco principal das atenções não está mais nos produtos criados durante o desenvolvimento, mas, sim, no próprio processo produtivo, visto que este tem se mostrado um dos fatores determinantes para o alcance da qualidade do produto final [8].

O objetivo do presente artigo é apresentar o resultado da avaliação de ferramentas, existentes atualmente no mercado, que permitem o desenvolvimento de artefatos de requisitos de software e o controle das mudanças sofridas no projeto, desde a etapa de levantamento preliminar dos requisitos. A abordagem proposta contempla desde a definição dos critérios e a seleção das ferramentas para avaliação até a apresentação e análise dos resultados obtidos. O resultado final deste trabalho será um importante auxílio para tomada de decisão de gerentes de projeto de engenharia de software.

O presente artigo está organizado da seguinte maneira: a segunda seção apresenta uma visão geral sobre a necessidade de gerência de requisitos e de configuração, segundo o modelo CMMi; as seções 3 e 4 apresentam respectivamente uma abordagem sobre gerência de requisitos e sobre gerência de configuração e mudança; a seção 5 apresenta a avaliação comparativa das ferramentas analisadas e a última seção traz a conclusão do trabalho.

2. Qualidade de Software: O Modelo CMMi

O CMMi é uma estrutura, ou conjunto de normas, que descreve os elementos de um efetivo processo de desenvolvimento de software [19]. Seu objetivo é descrever passos evolucionários de um processo imaturo até atingir a maturidade e disciplina no desenvolvimento de sistemas. Por ele, são cobertas práticas de planejamento, engenharia e gerenciamento de desenvolvimento e manutenção de software. São estabelecidos cinco níveis de maturidade (inicial – repetível – definido – gerenciável – otimizado), cada um dos quais é definido por áreas chaves de processo que, por sua vez, identificam um grupo de atividades relacionadas, as quais, quando executadas coletivamente, remetem a equipe ao alcance de metas importantes para o estabelecimento de maturidade no

processo.

Dentre as áreas chaves de processo do segundo nível estão a gerência de requisitos e a gerência de configuração de software e, para uma equipe mapear a aderência ao modus operandi que o CMMi pressupõe, é necessário inicialmente que essas áreas chaves sejam atendidas. Assim, torna-se fundamental, não somente atender as áreas chaves citadas, mas ainda que as atividades sejam feitas de modo integrado, para garantir a integridade das informações e evitar dubiedades e inconsistências. Portanto, desde o primeiro degrau na escada do modelo CMMi, integrar o gerenciamento dos requisitos à gerência de configuração do software é fundamental para a maturidade do processo.

3. A Gerência de Requisitos

Segundo o Dicionário Aurélio da Língua Portuguesa [10], requisito é uma condição necessária para alcançar certo objetivo. Kruchten [13] define requisito como "uma condição ou uma capacidade que o sistema deve atender". Certamente, o objetivo de todo desenvolvedor é a criação de um produto final que agregue qualidade ao ambiente onde será implantado. Portanto, é condição *sine qua non* que todo software atenda aos requisitos estabelecidos e acordados entre as partes envolvidas com o sistema. No entanto, o amadurecimento da prática de engenharia de software tem mostrado, ao longo do tempo, que os requisitos dos sistemas são elementos dinâmicos que mudam durante o processo de desenvolvimento. Controlar essas mudanças corresponde a um grande desafio para a equipe de gerência de requisitos. As principais preocupações da gerência de requisitos são: gerenciar mudanças nos requisitos acordados; gerenciar os relacionamentos entre os requisitos; gerenciar as dependências entre o documento de requisitos e outros documentos produzidos ao longo do processo [9]. A utilização de ferramentas apropriadas para o gerenciamento das mudanças nos requisitos é de grande importância, pois imprime maior qualidade ao processo de engenharia de software, o que, conseqüentemente, acarreta em qualidade no produto final gerado.

Vários fatores conduzem a mudanças nos requisitos. Segundo Breitman [3], evolução é "o processo de transformação sofrido por uma especificação ao longo da vida útil da aplicação que esta corresponde. Desta forma, uma especificação evolui durante as fases de desenvolvimento e manutenção da aplicação". Mais adiante, completa com a afirmação de que o principal motivo das mudanças evolutivas é "o fato dos próprios usuários não conhecerem a totalidade dos requisitos do sistema no momento de sua elicitação". Mudanças também decorrem da identificação de novas necessidades e do processo evolutivo de conhecimento do mundo real por parte da equipe desenvolvedora.

Este caráter evolutivo dos requisitos dos sistemas é "uma característica aceita pela maior parte dos desenvolvedores de sistemas e obriga a existência de um gerenciamento desta evolução, provendo mecanismos para manter a rastreabilidade, o controle de configuração e a eliminação de inconsistências e desatualizações" [2].

3.1. Ferramentas para Controle de Requisitos

Os Repositórios de requisitos são ferramentas de gerência de requisitos que devem prover um repositório central de informações acerca das necessidades apresentadas pelos usuários e por todas as partes envolvidas com o sistema a ser desenvolvido [5]. Esse repositório deve envolver os requisitos, seus atributos, inter-relacionamentos existentes e qualquer outra informação gerencial pertinente ao ambiente. Tal repositório não é importante apenas para o sucesso do projeto, ele passa a fazer parte do acervo da empresa, uma vez que os requisitos podem ser evoluídos, adaptados e reusados, acarretando em minoração dos custos e prazos de projetos seguintes.

Desse grupo de ferramentas, esperam-se algumas características básicas a fim de resolver problemas específicos, conforme segue:

Predisposição para detalhamento textual dos requisitos através de integração dinâmica com software editor de textos ou através de armazenamento de texto explicativo [1]: Os requisitos de um sistema provêm de fontes diversas de obtenção, tais como documentos ou observações que requerem um relato mais extenso. Quando armazenados em bancos de dados, esses requisitos carecem de contexto e são difíceis para os usuários entenderem. Por outro lado, quando armazenados somente em editores de texto, são difíceis de organizar, priorizar e rastrear.

Integração com outras ferramentas de desenvolvimento de sistema [3]: Os membros das equipes precisam ter acesso ao estado mais atual dos requisitos sem que se tenha de duplicá-los, portanto é necessário que seus artefatos estejam sob controle de versão.

Acesso distribuído dos requisitos [14]: Equipes geograficamente distribuídas precisam estar providas com as últimas informações dos requisitos.

Rastreabilidade dos requisitos [11]: Todas as funcionalidades esperadas devem ser implementadas, caso contrário, o motivo pelo qual não o foi deve estar devidamente documentado, como no caso dos requisitos conflitantes.

Impacto da mudança de um requisito [5]: Requisitos mudam e é preciso entender o seu impacto no sistema e em outros requisitos.

Permitir utilizar atributos e queries em requisitos [5]: Sem uma documentação de risco, prioridade e dificuldade associada a cada requisito, pode-se não conseguir definir objetivamente quais requisitos devem ser implementados primeiro.

Auditar as mudanças dos requisitos [21]: Muitas vezes é necessário saber quem criou ou modificou um requisito a fim de se entender seus porquês.

Repositório central para todos os elementos dos requisitos [21]: Requisitos são atualizados sem nenhum histórico de revisão, documentos de requisitos são compartilhados entre os membros da equipe e estes ficam sem ter certeza de qual é a mais atual, ou ainda, membros diferentes da equipe podem estar trabalhando com diferentes versões dos requisitos.

Acesso controlado às informações dos requisitos [14]: Requisitos são modificados por membros da equipe não autorizados para fazê-lo, resultando em requisitos que não espelham efetivamente as necessidades dos usuários.

Estrutura customizavel de requisitos [13]: Requisitos normalmente são numerosos e não organizados logicamente. É necessário, portanto que se possa organizar de acordo com uma taxonomia pré-definida.

Oferta de templates de projeto [13]: Templates de projeto evitam retrabalho de definição de projetos semelhantes e orientam as atividades de equipes pouco experientes.

Exportação dos conteúdos para um formato não proprietário [3]: Os requisitos de um sistema precisam ser divulgados a todos os membros da equipe de desenvolvimento e precisam ser independentes da ferramenta que os gerou.

4. Gerência de Configuração e Mudança

O software durante o seu ciclo de vida passa por diversos estágios e representações [16]. Seu desenvolvimento é um processo dinâmico que sofre mudanças de curso até a sua conclusão. Para que não haja inconsistência nos itens de informação importantes para o projeto, a criação e as alterações desses itens devem ser acompanhadas e controladas pelo gerente de projeto. O processo de gerência de configuração é um processo de ciclo de vida do software que permite que esse controle seja realizado [18].

4.1. Ferramentas para Gerência de Configuração

Existe, atualmente, no mercado, uma grande variedade de ferramentas para gerência de configuração, principalmente, no que diz respeito a controle de versão de software. A avaliação, no presente trabalho, é feita sob o ponto de vista de gerência de requisitos e como essas ferramentas podem agregar qualidade a essa atividade no processo de desenvolvimento. Desse grupo de ferramentas, esperam-se algumas características básicas a fim de suprir necessidades específicas, conforme apresentado a seguir:

Suporte a configurações [6]: Identificação de distintas versões de arquivos que compõem uma distribuição e recuperação da mesma. Devem ser analisados os aspectos de rastreabilidade que tornam possível o estabelecimento de relações entre distintos itens de configuração e seus históricos de modificações.

Controle de permissões [15]: Atribuir permissões individuais de leitura e gravação aos membros da equipe aos diversos itens de configuração.

Controle de arquivos em diversos formatos [5]: As ferramentas não podem se restringir ao controle de arquivos em formato ASCII, mas devem prover controle também em formatos Unicode e binário.

Suporte a desenvolvimento remoto [14]: Deve suportar projetos com diversos sites de acesso remoto. Necessário para equipes que se distribuem em diferentes áreas geográficas, em que cada membro realiza uma atividade, e a integra às demais.

Controle das mudanças [6]: Prover informações sobre quais itens estão em alteração por algum membro da equipe. Para cada alteração devem ser associadas informações sobre data, hora e responsável pela alteração, bem como permitir que se descreva textualmente um resumo sobre a alteração realizada.

Portabilidade [6]: Deve suportar a integração de itens oriundos de clientes com ambientes operacionais distintos.

5. Avaliação Comparativa das Ferramentas

Para a avaliação integrada das ferramentas, foram utilizados dois conjuntos distintos de ferramentas: o primeiro conjunto é composto pelas ferramentas Rational RequisitePro [17] para controle de requisitos e ClearCase [17] para controle de versões. Além dessas duas ferramentas, foi ainda utilizado o RationalRose [17] para geração do modelo de Caso de Uso. O segundo conjunto avaliado é composto pelas ferramentas REM [21] para controle de requisitos, CVSNT [4] para servidor e TortoiseCVS [20] para cliente de controle de versões. Foi ainda utilizada a ferramenta Jude-Take [12] para geração do modelo de Casos de Uso.

A escolha do primeiro grupo tem por objetivo trabalhar sobre um conjunto de ferramentas com participação expressiva no mercado mundial de ferramentas CASE[1]. No segundo grupo, a escolha recaiu sobre ferramentas livres, que têm recebido uma atenção especial do mercado corporativo e principalmente do meio acadêmico. Acredita-se que, com a escolha desses dois conjuntos, o presente trabalho possa contribuir com o maior número possível de usuários.

O estudo de caso desenvolvido para a avaliação, denominado sistema Paragon, é uma aplicação simples, porém representativa do domínio de comércio de varejo com transferência eletrônica de fundos. Esse domínio de aplicação é particularmente afetado por mudanças ambientais e legislativas.

[1] Acrônimo para Computer-Aided Software Engineering

Esperava-se, com a presente avaliação, sobretudo, observar o comportamento das ferramentas de gerência de configuração quanto ao tratamento e controle dos arquivos gerados pela equipe de controle de requisitos e sua disponibilização para os demais desenvolvedores de forma padronizada junto ao restante dos artefatos de desenvolvimento. Inicialmente, foi feita uma análise de cada uma das ferramentas individualmente, da qual se extraiu os resultados que serão apresentados na próximas subseções.

5.1. Ferramentas de Controle de Requisitos

5.1.1. Rational RequisitePro. A correlação entre os diversos requisitos levantados é feita através de planilhas, onde são estabelecidas as relações *Trace To* e *Trace From* de dependência entre os requisitos. É possível rastrear os casos de uso (UC), as especificações suplementares (SUPL) e as solicitações das partes envolvidas (STRQ) com as funcionalidades (FEAT). Na análise de impacto, é possível estabelecer o impacto causado nas características do sistema pelas requisições dos usuários, as especificações suplementares e os casos de uso impactados pelas mudanças nas características do sistema. Na pasta "Análise de cobertura", o software permite o gerenciamento das características não rastreadas por solicitações das partes envolvidas, requisitos suplementares e casos de uso não rastreados por características.

5.1.2. REM – Requirements Engineering Methodology. Os três elementos chaves tratados pela ferramenta são as especificações dos requisitos dos clientes (denominados requisitos-C ou CRS), especificações dos requisitos dos desenvolvedores (denominados requisitos-D ou DRS) e especificação dos conflitos (CFS). De acordo com o autor, essas especificações não têm necessariamente que coincidir com os documentos físicos, elas são simplesmente um instrumento para a organização abstrata das informações.

5.1.3. Avaliação Individual das Ferramentas de Controle de Requisitos. Muito embora não seja objetivo desse trabalho a avaliação individual desse grupo de ferramentas, faz-se necessário avaliar suas características específicas já que essas influenciarão diretamente na qualidade do processo de controle das mudanças dos requisitos. As ferramentas analisadas foram avaliadas com base nas características esperadas de ferramentas de controle de requisitos, descritas no item 3.1.

A avaliação das ferramentas é detalhada abaixo e em seguida apresentada resumidamente no quadro 1.

Predisposição para detalhamento textual dos requisitos através de integração dinâmica com software editor de textos ou através de armazenamento de texto explicativo: as duas ferramentas avaliadas atendem essa condição de diferentes formas. Enquanto o RequisitePro permite integração com textos de documentos criados no MS-Word, o REM permite que se criem textos explicativos associados a cada elemento de requisito.

Integração com outras ferramentas de desenvolvimento de sistema: apenas o RequisitePro prevê integração com outras ferramentas. O REM não atende de nenhuma forma esse requisito.

Acesso distribuído dos requisitos: também nesse quesito apenas o RequisitePro provê uma solução.

Rastreabilidade dos requisitos: ambas as ferramentas prevêem esse tipo de controle. Nos dois casos é possível associar um requisito a um ou mais casos de uso e verificar quais requisitos não estão sendo tratados em nenhum caso de uso.

Impacto da mudança de um requisito: também é prevista por ambas as ferramentas a análise de impacto que pode acarretar da mudança de algum requisito.

Permitir utilizar atributos e queries em requisitos: as duas ferramentas permitem a atribuição de propriedades aos requisitos.

Auditar as mudanças dos requisitos: ambas as ferramentas atendem esse quesito. No REM, é possível criar um histórico individual para cada mudança.

Repositório central para todos os elementos dos requisitos: ambas as ferramentas possuem repositório para os requisitos através de banco de dados.

Acesso controlado às informações dos requisitos: apenas o RequisitePro provê uma solução.

Estrutura customizavel de requisitos: nesse caso, ambas as ferramentas atendem de forma satisfatória.

Oferta de templates de projeto: apenas o RequisitePro provê uma solução.

Quadro 1: Avaliação das ferramentas de requisitos.

Característica	Requisite Pro	REM
Detalhamento Textual dos Requisitos	***sim***	***sim***
Integração com Outras Ferramentas	***sim***	***não***
Acesso Distribuído dos Requisitos	***sim***	***não***
Rastreabilidade dos Requisitos	***sim***	***sim***
Análise de Impacto das Mudanças	***sim***	***sim***
Suporta Atributos e Queries	***sim***	***sim***
Auditoria das Mudanças	***sim***	***sim***
Repositório Central dos Requisitos	***sim***	***sim***
Controle de Acesso aos Requisitos	***sim***	***não***
Estrutura Customizavel dos Requisitos	***sim***	***sim***
Oferta de Templates de projeto	***sim***	***não***
Exportação para formato não proprietário	***sim***	***sim***

Exportação dos conteúdos para um formato não proprietário: um dos pontos fortes do REM é a geração automática de toda a documentação em HTML. Dessa forma, fica mais fácil disponibilizar para os demais membros da equipe a visualização dos

requisitos do sistema. O RequisitePro permite a exportação individual das diversas visões (*views*) do projeto para formato MS-Word ou CSV. Outra solução provida pela ferramenta é o *RequisiteWeb*, que permite que usuários acessem o conteúdo do RequisitePro via internet.

5.2. Ferramentas de Gerência de Configuração

5.2.1. Rational ClearCase. No ClearCase, é possível a criação de uma VOB pública. Nesse caso, seus elementos podem ser acessados por toda a comunidade desenvolvedora; VOBs privadas somente podem ser acessadas por seus criadores.

5.2.2. CVSNT e TortoiseCVS. O CVSNT é uma ferramenta de código aberto, livre e licenciado sob a Licença Pública Geral GNU. Possui enfoque para o lado servidor da aplicação, sendo necessária a instalação de uma ferramenta cliente no ambiente do usuário, que interaja com o servidor. Dentre as ferramentas cliente para CVSNT, destaca-se o TortoiseCVS, um cliente front-end cujo objetivo é tornar a utilização do CVSNT mais fácil. TortoiseCVS trabalha diretamente através do Windows Explorer, o que torna intuitivo o trabalho daqueles que são familiarizados com esse padrão de interface gráfica.

5.2.3. Avaliação Individual das Ferramentas de Controle de Configuração. Muito embora não seja objetivo desse trabalho a avaliação individual desse grupo de ferramentas, faz-se necessário avaliar suas características específicas já que essas influenciarão diretamente na qualidade do processo de controle das mudanças dos requisitos. As ferramentas analisadas foram avaliadas com base nos critérios de características esperadas de ferramentas de controle de configuração, definidos anteriormente no item 4.1.

A avaliação das ferramentas é apresentada resumidamente no quadro 2 e detalhada em seguida.

Quadro 2: Avaliação individual das ferramentas de configuração.

Característica	ClearCase	CVSNT
Suporte a configurações	***sim***	***sim***
Controle de Permissões	***sim***	***sim***
Arquivos Binários	***sim***	***sim***
Ambiente Remoto	***sim***	***sim***
Controle de Mudanças	***sim***	***sim***
Portabilidade	***sim***	***sim***

Suporte a configurações: ambas as ferramentas analisadas implementam essa funcionalidade. Enquanto o ClearCase utiliza o conceito de View, o CVSNT utiliza o conceito de Tag/Label.

Controle de permissões: ambas as ferramentas possuem controle de acesso e permissões através de usuários e grupos, aos quais são atribuídas permissões de gravação, leitura ou acesso.

Controle de arquivos em diversos formatos: ambas as ferramentas suportam o controle de versões de arquivos binários sem comprometer a integridade de seus conteúdos. Nas duas análises, foram armazenados arquivos de ferramentas CASE para geração de diagramas UML e banco de dados com os requisitos do projeto.

Suporte a desenvolvimento remoto: nos testes realizados, foram avaliados apenas os ambientes de rede local. Ambas as ferramentas trabalham com o conceito de cliente/servidor e comportam-se adequadamente nesse ambiente. Apesar de descrita nas duas ferramentas, não foi testada a utilização em ambientes remotos.

Controle das mudanças: as descrições sobre as atualizações efetuadas são implementadas de formas similares nas duas ferramentas: sempre que se faz a atualização do repositório central com os arquivos alterados. Da mesma forma, ao se retirar um arquivo para atualização (operação de check-out), ficam registrados no servidor os dados de identificação dessa operação.

Portabilidade: todos os testes foram realizados em ambiente Windows XP Professional. Entretanto, na documentação de ambas as ferramentas é informada a sua compatibilidade com diversos ambientes operacionais, além da possibilidade de se trabalhar com ambientes de clientes diferentes do servidor e diferentes entre si.

5.3. Integração entre Rational RequisitePro e Rational ClearCase

A integração entre as duas ferramentas se dá de forma quase automática, o próprio RequisitePro já prevê que seus dados sejam armazenados sob controle do ClearCase.

Inicialmente, deve ser informado que os requisitos serão controlados pelo ClearCase e a localização da VOB na qual serão armazenados os elementos do projeto. Feito isso, a cada vez que se deseja criar uma nova versão dos elementos, deve-se solicitar dentro do próprio RequisitePro que se arquivem os elementos com o ClearCase. Nesse caso, é solicitado que o desenvolvedor informe um rótulo e uma descrição para a versão criada. Esses dados farão parte do histórico de revisões dos elementos dentro do ClearCase.

5.4. Integração entre CVSNT e REM

Esse caso representa a maioria dos casos atuais do mercado de ferramentas CASE, já que, apesar de existirem ferramentas poderosíssimas para diversos fins, a maioria trabalha isoladamente, sem se preocupar com a integração umas com as outras. O fato de não haver integração direta entre as ferramentas não comprometeu a integridade dos dados por elas

manipulados.

Inicialmente, é necessário configurar o CVSNT com a informação de quais arquivos devem ser tratados como binários, apesar da ferramenta possuir uma relação pré-definida desses tipos e a capacidade de converter os arquivos binários de seus formatos nativos em um formato no qual possa armazená-los em seus repositórios. Para que o CVSNT identificasse os arquivos gerados pelas ferramentas REM e Jude-Take como binários, foram inseridas novas extensões no arquivo *cvswrappers* (*.rem -kb e *.jude -kb)

Em seguida, deve ser criado um novo repositório ou módulo dentro de um repositório existente para o armazenamento das versões dos arquivos. Para o presente trabalho foi criado um novo repositório denominado /paragon e nele, um módulo denominado *par_teste*. Feito isso, os arquivos já criados podem ser copiados para a pasta que contém o repositório e quando se deseja extrair algum arquivo, basta selecionar o destino e escolher a opção "CVS obter módulo" do TortoiseCVS.

A cada operação de *checkout* e *commit*, o CVSNT registra as operações e, através do TortoiseCVS, é possível observar seu histórico.

5.5. Avaliação da Integração entre as ferramentas

Nos dois casos, os conjuntos de ferramentas apresentaram desempenho satisfatório e atenderam às atividades de gerência de configuração de requisitos. As operações de *checkout* e *checkin/commit* foram realizadas com sucesso, sem comprometer a integridade dos dados binários dos artefatos gerados pelas ferramentas. Dentre os pontos fortes dos dois conjuntos, é importante destacar a facilidade operacional das ferramentas integradas e a possibilidade de integração das especificações dos requisitos às baselines do software. Certamente, o ponto de maior destaque das ferramentas RequisitePro e ClearCase foi a integração prevista pela primeira; entretanto, a inexistência dessa característica não comprometeu em momento algum o funcionamento do conjunto REM e CVSNT, apenas abreviou a intuição da atividade.

6. Conclusão

A definição e a utilização de um processo consistente de desenvolvimento é uma condição *sine qua non* para o alcance de índices cada vez mais altos de qualidade dos produtos desenvolvidos. No entanto, não basta somente definir e utilizar um processo, também é preciso que este esteja amparado por ferramentas que verdadeiramente dêem suporte ao trabalho dos desenvolvedores, respeitando a particularidade de cada ambiente e perfil de cada equipe. É de mister importância, também, que essas ferramentas estejam interligadas e, dessa forma, amparem-se mutuamente no objetivo de majorar a qualidade do processo, refletindo fidedignamente, através das suas funcionalidades, as atividades dos desenvolvedores.

Para que uma ferramenta apóie devidamente a prática humana, é necessário que esteja amparada para atender àquela que seja talvez a mais certa de todas as características de um desenvolvimento: a mudança nos requisitos ao longo do projeto.

Baseado nessa hipótese, o presente trabalho apresentou uma avaliação do uso de ferramentas para o desenvolvimento e controle das alterações dos artefatos de requisitos. Para apoiar o estudo comparativo, foi desenvolvido um sistema em que os procedimentos de verificação e controle das mudanças foram devidamente testados.

Durante o decorrer deste trabalho, foram observados alguns pontos que devem ser mencionados. A mudança ocorrida nos requisitos no sistema é prevista por todos os processos atuais de engenharia de software e refletem diretamente no produto final do processo, entretanto, poucas são as equipes que possuem um controle efetivo sobre seus efeitos. Quanto mais interligadas estiverem as ferramentas CASE, mais interligados estarão os artefatos por elas produzidos. Por isso a sugestão feita pelo presente trabalho é que o mesmo ambiente que controla as mudanças nos códigos fontes sirvam para o controle das alterações nos requisitos. Isso aumenta a integração e a familiaridade dos desenvolvedores com o ambiente.

A crescente preocupação das empresas em produzir software de qualidade reconhecida tem elevado a demanda de se estabelecer padrões de eficiência nos seus processos de desenvolvimento. Nesse contexto, o objetivo deste trabalho foi propor um processo sistemático, baseado em ferramentas de controle das mudanças nos artefatos de requisitos de software durante o seu processo de desenvolvimento.

7. Referências Bibliográficas

[1] BATISTA, Edinelson Aparecido; Carvalho, Ariadne M.B.R. Uma Taxonomia Facetada para Técnicas de Levantamento de Requisitos. Anais do WER03 - Workshop em Engenharia de Requisitos, Piracicaba-SP, 2003.

[2] BERGMANN, Ulf. Evolução de Cenários Através de um Mecanismo de Rastreamento Baseado em Transformações. Tese de Doutorado. Disponível em http://www-di.inf.puc-rio.br/~julio/bnncap3.pdf. Acessado em 19/05/05, Rio de janeiro: PUC-RJ, 2003.

[3] BREITMAN, Karin K.; Leite J.C.S.P. Suporte Automatizado à Gerência da Evolução de Cenários. Anais do WER98 – Workshop em Engenharia de Requisitos, Maringá-PR, 1998.

[4] CVSNT. Disponível em http://www.march-hare.com/cvsnt. Acessado em 09/10/2005.

[5] DAVIS, Alan M, Dean A. Leffingwell. Using Requirements Management to Speed Delivery of Higher Quality Application. Disponível em http://www-306.ibm.com/software/rational/info/literature/reqanalysis.jsp. Acessado em 20/08/2004, 1996.

[6] DIAS, André Felipe. GC do Ponto de Vista das Ferramentas de Apoio. Disponível em http://www.pronus.eng.br/artigos_tutoriais/gerencia_configuracao/gerencia_configuracao.php?pagNum=3. Acessado em 20/04/2006.

[7] GNU. The GNU General Public License. Disponível em http://www.gnu.org/licenses/ licenses.html#GPL. Acessado em 24/10/2005.

[8] GOMES JUNIOR, Augusto Gonçalves, Avaliação de Processos de Software baseado em Medições. Tese de Doutorado. Rio de Janeiro: COPPE/UFRJ, 2000.

[9] HAZAN, Claudia e LEITE, J. C. S. P. Indicadores para a Gerência de Requisitos. Anais do WER03 - Workshop em Engenharia de Requisitos, Piracicaba-SP, 2003.

[10] HOLANDA, Aurélio Buarque de Holanda. Dicionário da língua portuguesa. Rio de Janeiro,: Nova Fronteira, 1988.

[11] JARKE, M. Requirements tracing. Communications of the ACM, 41(12):32--36, 1998

[12] JUDE. Disponível em http://www.componentsource.com/. Acessado em 09/12/2005.

[13] KRUCHTEN, Phillipe. The Rational Unified Process: an Introduction. Boston, EUA, 2000.

[14] LOPES, Leandro T.; Majdenbaum, Azriel; Audy ,Jorge Luis N. Uma Proposta para Processo de Requisitos em Ambientes de Desenvolvimento Distribuído de Software.. Anais do WER03 - Workshop em Engenharia de Requisitos, Piracicaba-SP, Brasil, 2003.

[15] MILLIGAN, Tom. Better Software Configuration Management Means Better Business: The Seven Keys to Improving Business Value. Disponível em http://www-306.ibm.com/software/rational/info/ literature/. Acessado em 20/08/2004, 2003

[16] OLIVEIRA, Angelina A.A.C.P.. Gerência de Configuração de Software. Instituto Nacional de Tecnologia da Informação – ITI, órgão do Ministério da Ciência e Tecnologia, Campinas, SP: 2001.

[17] RATIONAL. Use Case management with Rational Rose and Rational RequisitePro. Disponível em http://www.306.ibm.com/software/rational/info/literature/reqanalysis.jsp. Acessado em 20/08/2004.

[18] ROCHA, Ana Regina Cavalcanti da, Maldonado, José Carlos, Weber, Kival Chaves. Qualidade de Software – Teoria e Prática. São Paulo: Prentice Hall, 2001.

[19] SEI. Software Engineering Institute Technical Report CMU/SEI-93-TR-024, 1993. Disponível em http://www.sei.cmu.edu. Acessado em 20/08/2005.

[20] TORTOISECVS. Disponível em http://www.tortoisecvs.org/. Acessado em 24/10/2005.

[21] TORO, Amador Duran. Un Entorno Metodológico de Ingeniería de Requisitos para Sistemas de Información. Tese de Doutorado. Disponível em http://www.lsi.us.es/~amador/. Acessado em 16/09/2005. Sevilha, Espanha, 2000.