Processo de Software Livre em Ambiente Acadêmico: Experiências e Lições Aprendidas

Débora M. B. Paiva
Universidade de São Paulo, Dept. Computação,
São Carlos, Brasil, 13560-970
debora@icmc.usp.br

and

Marcelo A. S. Turine
Universidade Federal do Mato Grosso do Sul, Dept. Computação,
Campo Grande, Brasil, 79070-900
turine@dct.ufms.br

and

Renata P. M. Fortes
Universidade de São Paulo, Dept. Computação,
São Carlos, Brasil, 13560-970
renata@icmc.usp.br

Abstract

Development of free software in academic setting has been common in the last years. In this paper we present our experience and lessons learned with the SAFE (*Software Engineering Available for Everyone*) project, from the software process perspective. We discuss the main issues regarding software process standards we have applied and the influences among them, research interests and OSS (*Open Source Systems*) characteristics. SAFE is a research project being developed by university and industrial sectors. We believe that researchers and free software community members can benefit from lessons learned and count on improvements in their own methodologies.

Keywords: Software Process, Open Source Systems, Academic Software Process.

Resumen

O desenvolvimento de software livre em ambiente acadêmico se tornou bastante comum nos últimos anos. Neste artigo apresentamos nossa experiência e lições aprendidas com o projeto SAFE (Engenharia de Software Disponível para Todos), sob a perspectiva de processo de software. Discutimos as principais questões relacionadas com os padrões de processo de software que aplicamos e as influências entre esses, os interesses de pesquisa e as características de Software Livre. O projeto SAFE é um projeto de pesquisa em desenvolvimento por parceiros de duas universidades e uma empresa. Acreditamos que pesquisadores e membros da comunidade de software livre podem se beneficiar das lições aprendidas e refletir sobre possíveis melhorias em suas metodologias.

Palabras claves: Processo de Software, Software Livre, Processo de Software Acadêmico.

1 INTRODUÇÃO

Nos últimos anos, os governos de diversos países têm incentivado o desenvolvimento de projetos de pesquisa envolvendo software livre. Por exemplo, o projeto de pesquisa denominado COSPA [25], desenvolvido por quinze instituições da Europa, com participação de membros da academia, indústria e governo de seis diferentes países, tem como principal objetivo estudar como *Open Source Systems* (OSS) podem ser utilizados na administração pública. No Brasil, de forma similar, o interesse por parte do governo em financiar a pesquisa e estimular o desenvolvimento de software livre tem sido observado em várias ações [1]. Uma das sete diretrizes para implantação e operação do governo eletrônico de toda a administração pública federal é a implementação de softwares livres. Tal diretriz serve como referência geral para estruturar as estratégias de intervenção, adotadas como orientações para todas as ações de governo eletrônico, gestão do conhecimento e gestão da TI no governo federal[2].

No entanto, em ambiente de pesquisa o desenvolvimento de software (seja software livre ou não) possui peculiaridades que devem ser convenientemente tratadas. De acordo com Ambati e Kishore [2], as características especiais do desenvolvimento de software acadêmico, tais como prototipação rápida e alta rotatividade dos membros dos projetos representam diferenças fundamentais que impõem uma revisão das práticas de Engenharia de Software para atender às necessidades do contexto acadêmico. Os autores também sugerem que dificuldades comuns no desenvolvimento de software em ambiente de pesquisa, como a comunicação e a coordenação de projetos, sejam resolvidos utilizando-se práticas e ferramentas advindas da comunidade de software livre. De forma similar, observa-se que atividades comumente cumpridas no desenvolvimento de projetos de pesquisa podem ser adotadas por desenvolvedores de software livre no sentido de colaborar para a solução de diversas dificuldades que têm sido percebidas por eles. Bezroukov [5] também enfatiza que há muitas possibilidades de colaboração entre a comunidade de software livre e a comunidade acadêmica.

O projeto SAFE (*Software Engineering Available for Everyone*) [13] é um projeto de software livre que iniciou em 2004 e está em fase avançada de desenvolvimento. Seu objetivo é a implementação de uma infra-estrutura que permita integrar ferramentas de software livre de apoio às atividades de Engenharia de Software, possibilitando um apoio automatizado ao processo de desenvolvimento de projetos de software livre. Durante o seu desenvolvimento tem sido observado que características, restrições e demandas do contexto do desenvolvimento de pesquisas e do desenvolvimento de software livre se complementaram de forma a compor o processo de desenvolvimento que está sendo utilizado. Os processos de software livre muitas vezes precisaram ser adaptados e um fator determinante para que isso ocorresse foi a existência de gerenciamento e financiamento externo ao projeto. Por outro lado, os processos acadêmicos de software também sofreram modificações de forma a atender as características de um projeto distribuído e colaborativo.

Neste artigo é apresentado o processo utilizado para o desenvolvimento do projeto SAFE, que envolve características de desenvolvimento de software livre e desenvolvimento de projetos de pesquisa em um contexto de desenvolvimento ágil. São indicados casos de sucesso, problemas encontrados e lições aprendidas. Buscou-se também comparar os resultados obtidos no contexto do projeto SAFE com as experiências obtidas com o desenvolvimento de outros projetos, relatadas em documentos científicos. Esta comparação foi realizada tanto em relação a projetos de software livre quanto em relação a projetos de pesquisa. Espera-se que a partir da experiência relatada, outros pesquisadores e comunidade de software livre, que já estejam envolvidos ou que pretendam se envolver com o desenvolvimento de software livre, se beneficiem dessa análise e a considerem como estímulo para continuar aprimorando essa sinergia.

Na Seção 2 são discutidos os processos de software utilizados em ambiente acadêmico e os processos de software utilizados pela comunidade de software livre. Na Seção 3 são apresentadas as principais características do projeto SAFE, na Seção 4 é descrita a experiência e as lições aprendidas no desenvolvimento do projeto. Na Seção 5 são apresentadas experiências relativas ao cumprimento de um processo distribuído. Finalmente, na Seção 6 são discutidas as principais conclusões.

2 PROCESSOS DE SOFTWARE UTILIZADOS EM AMBIENTE ACADÊMICO E PROCESSOS DE SOFTWARE LIVRE

Nos últimos anos, diferentes paradigmas têm sido investigados para atender ao desenvolvimento de pesquisas acadêmicas em relação aos processos de software cumpridos [17, 9]. Apesar dos resultados que têm sido obtidos ainda é comum

[1] http://www.softwarelivre.gov.br/SwLivre/

[2] http://www.governoeletronico.gov.br/SwLivre/

que softwares acadêmicos sejam re-escritos por diferentes pessoas devido às dificuldades de manutenção e reuso, acarretando prejuízos à evolução de pesquisas científicas [6].

Diversos pesquisadores têm indicado as principais práticas de engenharia de software que devem ser cumpridas em ambiente de pesquisa. No geral, espera-se que um processo cumprido em ambiente de pesquisa valorize, principalmente, atividades de prototipação, atividades relacionadas ao gerenciamento de tarefas, cronograma e recursos dos projetos, atividades de qualidade de software como documentação, gerenciamento de configurações, testes e a comunicação entre os membros dos projetos [21, 22, 11].

Uma experiência interessante em relação à aplicação de práticas de Engenharia de Software no desenvolvimento de pesquisas foi relatada por Walker [26]. Os esforços concentrados na melhoria dos processos cumpridos no *Software Engineering Applications Laboratory* resultaram na obtenção de importantes certificações. Muitas lições foram aprendidas com esta experiência e os autores destacam como importantes fatores de sucesso o gerenciamento contínuo dos projetos e a documentação subjacente.

Em relação ao processo de software livre, podem ser observadas variações substanciais entre os projetos, e não devem ser feitas generalizações amplas. De acordo com Reis [19], existe uma preferência clara por ferramentas de controle de versão e listas de correio eletrônico, que são utilizadas pela grande maioria dos projetos. Uma parcela menor dos projetos de software livre, porém significativa utiliza sistemas de acompanhamento de alterações/defeitos.

Para coordenar o trabalho que realizam, os membros da comunidade de um projeto utilizam a Internet através de ferramentas simples amplamente disponíveis: correio eletrônico, páginas Web, listas de discussão, em conjunto com ferramentas de desenvolvimento de software. Em geral, os membros dos projetos tendem a usar estes veículos para comunicar experiências, problemas e solicitações. O uso intensivo dessas ferramentas apóia o desenvolvimento de projetos desenvolvidos tipicamente de forma distribuída, como ocorre no contexto de software livre.

É interessante notar que o fato das ferramentas apoiarem a troca de experiência na comunidade tem como efeito colateral a possibilidade de arquivar-se esta comunicação para análise futura. Normalmente estes arquivos online são indexados por serviços de busca na Web e servem como uma fonte importante de documentação para os usuários, já que freqüentemente apresentam problemas recorrentes, já discutidos no passado por outros [19].

Assim, observando-se as ferramentas comumentes utilizadas pela comunidade de software livre é possível identificar os processos mais valorizados. Nota-se que a comunicação entre os membros é um dos fatores mais importantes para o sucesso dos projetos. Outras atividades fundamentais se referem ao controle de versões e controle de alterações.

Conforme apresentado por Bezroukov [5] os processos de software livre e os processos acadêmicos de fato se sobrepõem. O autor destaca que é importante fazer uma analogia entre a forma de trabalho da comunidade científica e da comunidade de software livre e identificar as experiências bem sucedidas que podem favorecer uma a outra.

3 CARACTERÍSTICAS GERAIS DO PROJETO SAFE

O desenvolvimento de uma infra-estrutura que permita integrar ferramentas de software livre de apoio às atividades de Engenharia de Software mencionada anteriormente é o principal objetivo do projeto SAFE. Dessa forma, pretende-se oferecer um suporte automatizado para o processo de software livre, que seja simples o suficiente para atrair a colaboração e a participação de desenvolvedores (nos diversos níveis de familiaridade com o processo OSS). Está em desenvolvimento atualmente um ambiente integrando ferramentas OSS (*webtools*), tais como Bugzilla[3] e Subversion[4], de suporte às atividades de projeto.

O projeto SAFE foi aprovado em uma chamada pública da FINEP em 2004. Nessa chamada foram submetidas cento e vinte e três propostas das quais apenas dezesseis foram aprovadas. O projeto SAFE foi um dos contemplados, recebendo, portanto, financiamento da FINEP[5].

As principais metas a serem alcançadas com o desenvolvimento do projeto SAFE são apresentadas a seguir:

- *Capacitação de pessoal* - capacitar pessoal por meio do uso intensivo das *webtools* mais utilizadas em processos de software livre (Bonsai e Bugzilla).
- *Implementação do framework* - especificar e implementar o framework (SAFE) para integração das ferramentas de auxílio ao desenvolvimento de software livre.
- *Elaboração de cartilha* - elaborar uma “cartilha” explicando o uso do framework SAFE.

[3] www.bugzilla.org
[4] http://subversion.tigirs.org
[5] Chamada Pública MCT/FINEP/CT-INFO - 01/2003 - Convênio 01.04.0477-00

- *Desenvolvimento Wiki/RE* - desenvolver *Wiki/RE*, como uma ferramenta para suporte a engenharia de requisitos que será integrada ao framework.
- *Validação e teste do framework* - validar e testar do framework por meio da sua utilização durante o desenvolvimento de ferramentas.
- *Transferência de tecnologia* - transferir e multiplicar a tecnologia de framework desenvolvida para utilização real em empresas de desenvolvimento de software.

A equipe original do projeto SAFE, conforme descrita na proposta submetida em 2004, era composta por 32 pessoas, sendo 3 coordenadores (um de cada uma das entidades envolvidas: ICMC-USP, UFMS e Async Open Source); os demais membros sendo pesquisadores colaboradores e professores das instituições de ensino envolvidas (ICMC-USP e UFMS).

Por ser realizado principalmente em ambiente acadêmico, onde existe alta rotatividade dos pesquisadores envolvidos devido às distintas fases de compromisso com seus programas de estudo, a equipe sofreu algumas alterações, principalmente entre alunos de mestrado e de graduação. Assim, foram desligados 5 estudantes e, por outro lado, 7 novos membros passaram a integrar a equipe. Atualmente, 34 pessoas atuam diretamente na pesquisa e desenvolvimento (P&D) relacionados ao projeto SAFE.

Como resultado do desenvolvimento do projeto foram obtidas 12 publicações em conferências nacionais, uma publicação em revista nacional e cinco publicações em conferências internacionais; foram gerados oito relatórios técnicos e têm sido desenvolvidos 13 projetos de iniciação científica, 5 projetos de mestrado e 3 projetos de doutorado.

4 A EXPERIÊNCIA NO PROJETO SAFE

Para apresentar a experiência na perspectiva do processo de desenvolvimento de software, a seguir são discutidos cinco sub-processos cumpridos e as lições aprendidas relacionadas.

4.1 CONCEPÇÃO DO PROJETO

Desde o final da década de 1990, uma equipe de estudantes e pesquisadores do ICMC-USP está envolvida com projetos de software livre e o processo aplicado nesse tipo de projeto. Diversos resultados na forma de projetos de software livre têm sido obtidos, tais como:

VersionWeb: a primeira versão da ferramenta *VersionWeb* foi lançada em 2000. Possibilita uso da maior parte das operações do CVS (*Concurrent Versions System*) através de uma interface Web [24, 23].

DocRat: a primeira versão da ferramenta *DocRat* foi lançada em 2004 e era denominada *DocRationale*. Consiste de uma ferramenta Web para captura, armazenamento e a recuperação de *design rationale* de artefatos de software, implementa um repositório digital dos artefatos produzidos durante o processo de software, associado ao seu *design rationale*. Apoia ao processo de desenvolvimento de software, pois permite gerenciamento das informações relativas aos projetos como planejamento de fases, equipes e datas. Em 2005 foi reimplementada e uma nova *release* da *DocRat* baseada na abordagem SoC (*Separation of Concerns*) está disponível [14, 1].

NoRiskPlanning: a primeira versão da ferramenta *NoRiskPlanning* foi lançada em 2001. Consiste de uma ferramneta Web, desenvolvida como uma agenda eletrônica para planejamento de atividades pessoais ou em grupo. Tem sido utilizada também na Intranet do ICMC para reserva de salas, por todos docentes e funcionários. Em 2005 foi reimplementada e uma nova *release* da *NoRiskPlanning* baseada na abordagem SoC (*Separation of Concerns*) está disponível [12].

Homework: a primeira versão da ferramenta *Homework* foi lançada no início do ano de 2005. Consiste de uma ferramenta Web para acompanhamento de tarefas em grupos, que podem ser realizadas de maneira assíncrona; ajuda os alunos a se comunicar, bem como notifica todos os participantes: professores e alunos, sobre as diversas intervenções durante a realização das tarefas, ou trabalhos práticos, que podem ser enviados via Web.

Py-VersionWeb: esta ferramenta, em fase de especificação, consiste de evolução da ferramenta *VersionWeb* para trabalhar com repositórios SubVersion e agregar funcionalidades de grupo, na forma de um *Groupware*, oferecendo maior suporte para projetos de software livre segundo as características dos mesmos, identificadas no projeto *VersionWeb*, citado anteriormente.

Os resultados obtidos com o desenvolvimento desses projetos, juntamente com os resultados obtidos no contexto do mestrado de Reis [19] motivaram a realização de novas investigações por uma equipe maior, em que pessoas possuindo diferentes competências pudessem colaborar no intuito de aprofundar os conhecimentos e gerar novas soluções para os problemas identificados a partir da experiência obtida.

Para viabilizar o desenvolvimento do projeto foram preenchidos diversos formulários que foram solicitados pela FINEP, agência que financia o projeto. Foram apresentadas informações como objetivo geral do projeto, objetivos específicos, metas, justificativa, metodologia, resultados esperados, planejamento da gerência do projeto, impacto econômico, científico, tecnológico e social, identificação dos participantes e indicação do grau de instrução, elaboração de cronograma e descrição de recursos financeiros solicitados.

Conforme observado por Reis [19], a maioria dos projetos de software livre são iniciados por motivos pessoais, por não existir um sistema que possua as funcionalidades necessárias, não havendo uma contribuição significativa de uma organização formal. Mesmo projetos de maior sucesso, como Linux e Perl tiveram início como projetos pessoais. Assim, o valor estratégico dos projetos geralmente é facilmente compreendido, de forma tácita, pelos participantes dos mesmos, pois são desenvolvidos sistemas que atendem às necessidades particulares dos desenvolvedores. Por outro lado, como observado por Bellotti et al [4], durante a fase de elaboração de um projeto de pesquisa é fundamental identificar e apresentar, de forma clara, o valor estratégico do projeto e seu impacto na sociedade. Além disso, conforme observado por Boldyref et al [6] em sua experiência com o desenvolvimento do projeto OSCAR (*Open Source Component and Artefact Repository*), os pesquisadores devem ser muito cuidadosos em relação aos objetivos e metas que eles buscam alcançar, isto é, devem evitar objetivos elevados e difíceis de alcançar em um curto período de tempo. Tais fatores estão sendo considerados no desenvolvimento do projeto SAFE e podem ser considerados requisitos básicos para a sustentabilidade da pesquisa e da extensão no ambiente acadêmico.

Lições Aprendidas:

- Para viabilizar o desenvolvimento do projeto SAFE, que se apresenta como um projeto de software livre desenvolvido em ambiente acadêmico, foram requeridas diversas informações pela agência financiadora, dentre elas, o objetivo do projeto, a metodologia para desenvolvimento e os resultados esperados. Essas informações foram apresentadas considerando-se fortemente o embasamento científico conquistado por alguns membros da equipe em trabalhos anteriores envolvendo o tema do projeto e a relevância e impacto na sociedade. Expressar de forma clara estas informações ajudaram a evidenciar aos avaliadores do projeto (na agência de financiamento) a relevância dos problemas e das soluções de tecnologia de informação inovadoras propostas. Outro ponto favorável observado foi que a equipe se sentiu motivada e responsabilizada pelo desenvolvimento do projeto.
- Como parte das atividades de concepção do projeto, os recursos necessários para o desenvolvimento foram identificados e solicitados à agência de financiamento. Observou-se que foi importante evidenciar os recursos financeiros de acordo com as necessidades das organizações solicitados e justificá-los. Além disso, os recursos humanos (a equipe) foi contactada e estimulada a apresentar suas competências em relação ao tema proposto e se familiarizar e conscientizar das metas propostas.

4.2 CICLO DE VIDA

No início do projeto foram realizadas diversas reuniões entre os membros envolvidos para a compreensão do objetivo do projeto e identificação dos requisitos iniciais. Os participantes foram motivados a expressar suas opiniões a respeito da proposta aprovada, identificar casos de uso do software e indicar possíveis direcionamentos para o desenvolvimento do projeto (uso de metodologias, ferramentas, plataformas e arquiteturas). A maioria das reuniões foi realizada em formato de *brainstorming*, cuidando-se do planejamento e do registro (por meio de atas) dos ítens apresentados e discutidos em cada reunião. Como resultado dessas reuniões foram gerados diagramas de casos de uso, descrição dos casos de uso e modelo de dados por duas das instituições participantes. Toda documentação do projeto tem sido publicada em uma wiki específica do projeto[6] [3]. A comunicação por emails continua sendo a forma mais intensamente utilizada, e por meio de uma lista criada para o projeto, contém informações relativas ao planejamento e ao registro das informações das reuniões, além de questionamentos técnicos e discussões diversas para fundamentar as decisões.

[6] http://safe.icmc.usp.br/coteia

A atividade evoluiu e na seqüência foi elaborado um projeto rápido do sistema e um protótipo foi desenvolvido pelos membros da terceira instituição participante, conforme ilustrado na Figura 1, que consiste na página Web de acesso à ferramenta Bugzilla, após o usuário ter realizado *login* na tela inicial do framework SAFE. Esses resultados do desenvolvimento inicial do projeto foram discutidos no I Workshop SAFE[7], em abril de 2005, em que estiveram presentes representantes de todas as instituições envolvidas. Foram tomadas diversas decisões relacionadas ao planejamento das atividades do projeto, que também foram registradas em formato de ata, enviadas por email aos participantes e armazenadas na wiki do projeto.

Figura 1: Exemplo de uso da ferramenta Bugzilla no SAFE

A partir do desenvolvimento do protótipo, toda a equipe manifestou que foi possível entender melhor e refinar os requisitos do sistema. Como conseqüência, atualizações foram realizadas nos diagramas inicialmente gerados. Com esta evolução, decidiu-se por dividir o trabalho de implementação entre os membros de cada instituição envolvida, reforçando a abordagem colaborativa, tanto na tomada de decisões como na realização das tarefas de desenvolvimento propriamente. Os resultados obtidos dessa etapa de desenvolvimento foram então apresentados no II Workshop SAFE[8], em novembro de 2005. Novamente estiveram presentes membros das três instituições envolvidas. Com as discussões ocorridas, os casos de uso do sistema foram novamente refinados e a implementação continuou a evoluir. É importante observar que durante a realização dos workshops, em que as atividades executadas e os resultados obtidos foram apresentados, sentiu-se a necessidade da elaboração de outros diagramas (principalmente diagrama de classes e de arquitetura do sistema), de forma a facilitar o entendimento e a evolução do projeto. Considerando-se que o projeto está sendo desenvolvido de forma distruibuída e colaborativa, comumente os membros das diferentes instituições participantes precisavam entender e dar continuidade a partes do sistema desenvolvido por outros membros. Um problema identificado neste sentido foi a falta de documentação de testes realizados, que gerava incertezas a respeito da avaliação da implementação.

Para o desenvolvimento de projetos de pesquisa, a adoção de um ciclo de vida iterativo tem sido indicada como fundamental [22, 9], principalmente devido ao fato de que em ambiente acadêmico percebe-se que os requisitos emergem em sucessivas iterações. Para o desenvolvimento do projeto SAFE não foi estabelecido inicialmente um modelo de ciclo de vida a ser utilizado e, na prática, a abordagem iterativa sugerida na literatura foi adotada e evidenciada como sendo naturalmente necessária.

Em relação ao ciclo de vida também podem ser observadas diferenças em relação ao desenvolvimento de projetos de pesquisa e de projetos de software livre. As fases de análise de requisitos e testes merecem ser destacadas como diferenciais. Em projetos de pesquisa os requisitos são identificados na medida em que o projeto é desenvolvido e testes comumente não são realizados [9]. Em projetos de software livre é muito comum que funcionalidades de outros pacotes ou sistemas sejam replicadas e, assim, pelo menos parte dos requisitos de um novo projeto já foram descritos ou implementados anteriormente. Em relação a testes, há grande confiança em teste funcional realizado pelo usuário final, que é conhecido como teste beta [19].

[7] http://safe.icmc.usp.br/safe/events-old/apresentacoes-do-i-workshop-safe
[8] http://safe.icmc.usp.br/safe/events-old/apresentacoes-ii-workshop-safe

Lições Aprendidas: a experiência em relação ao ciclo de vida para o desenvolvimento do projeto foi interessante principalmente em relação à "descoberta" dos requisitos do sistema. A abordagem iterativa e incremental foi adotada na prática e os efeitos positivos foram observados. A atividade de testes é essencial no processo e podem ser utilizadas práticas cumpridas pela comunidade de software livre, por exemplo, realização de testes beta e disponibilização dos sistemas em repositórios públicos. O uso de ferramentas de controle de alteração, como a bugzilla, são essenciais nesses casos. O registro e a documentação do processo de comunicação por meio de atas na wiki e emails é primordial para a recuperação das decisões tomadas (design rationale) [20]. Outra possibilidade se refere à aplicação de metodologias ágeis no desenvolvimento de projetos de pesquisa. Por exemplo, XP é um método ágil que possui como uma de suas práticas a realização de testes. Este método tem sido explorado para o desenvolvimento de projetos de pesquisa [4, 9].

4.3 TRANSFERÊNCIA DE CONHECIMENTO ENTRE OS PARTICIPANTES

Uma atividade bastante interessante no contexto do projeto foi a transferência de conhecimento entre as pessoas envolvidas, que ocorreu, principalmente, devido a dois fatores:

(1) Em ambiente acadêmico é comum a alta rotatividade dos membros de um projeto e isso de fato ocorreu durante o desenvolvimento do projeto SAFE;

(2) Após a aprovação do projeto e assinatura de contrato das instituições envolvidas, iniciou-se o processo de aquisição de equipamentos de infra-estrutura, o qual se prolongou por quatro meses, atrasando o início do projeto. Neste período, membros de uma das instituições começaram a pesquisar a potencialidade de algumas ferramentas que poderiam ser utilizadas. Quando o projeto começou a ser desenvolvido de fato, estes membros já possuíam familiaridade com diversos conceitos, metodologias e ferramentas relacionadas ao tema do projeto e optou-se pela transferência do conhecimento adquirido para os demais membros.

As atividades cumpridas com o objetivo de viabilizar a transferência de conhecimento entre os participantes estiveram relacionadas basicamente à elaboração de documentação sobre os tópicos estudados no formato de relatório técnico e a realização de cursos para os membros. Por exemplo, iniciou-se a elaboração de material didático sobre a ferramenta Bugzilla. Gerou-se um relatório técnico contendo os objetivos e os recursos da ferramenta, suas principais características e visão geral de suas funcionalidades. O relatório foi disponibilizado aos membros e um minicurso foi realizado no I Workshop do projeto SAFE. Observa-se ainda que o material gerado foi utilizado em diversas outras ocasiões, por exemplo, em disciplinas do curso de Ciência da Computação das duas instituições [18], em outros projetos de pesquisa [9] e em cursos de extensão ministrados para a comunidade, o que possibilitou também a transferência do conhecimento adquirido para outras entidades externas ao projeto.

Lições Aprendidas: a distribuição geográfica dos membros da equipe não prejudica o desenvolvimento de projetos acadêmicos de software livre. A metodologia baseada em encontros presenciais (realização de workshops) e registro dos resultados das discussões foi produtiva. A elaboração de material didático e cursos de extensão sobre o tema do projeto foi fundamental para a formação técnica dos membros e para a transfeência de conhecimento entre eles. Uma forma de aprimorar a abordagem utilizada seria colocar em prática a sugestão de Paula Filho [11] em relação à adoção de uma base de conhecimento que possa ser compartilhada entre os membros de projetos, contendo diferentes tipos de material didático.

4.4 PLANEJAMENTO E GERENCIAMENTO DO PROJETO

No início do desenvolvimento do projeto planilhas foram geradas, utilizando a ferramenta OpenOffice.org Calc, como base para o planejamento inicial e o gerenciamento das atividades do projeto (relacionando as metas do projeto e as pessoas responsáveis; as metas e prazos para cumprimento e as metas, prazos para cumprimento e artefatos que deveriam ser elaborados).

As planilhas foram úteis na atribuição inicial das metas aos participantes e na definição de prazos e de artefatos a serem gerados (de forma mais detalhada que na fase de concepção do projeto). Observando-se que a atualização das

[9] Por exemplo, o projeto Tidia-AE utilizam os recursos disponibilizados pelo projeto SAFE, que inclui documentação, material para apresentação de cursos e repositórios

planilhas estava se tornando uma tarefa pouco eficiente a medida que o volume de informações aumentava, optou-se por utilizar um sistema de gerenciamento online, no caso, o NetOffice[10]. As pessoas envolvidas foram motivadas a acrescentar informações referentes ao desenvolvimento de suas atividades. Apesar de a equipe ter sido treinada para usar a ferramenta e a importância em utilizá-la ter sido enfatizada, as informações registradas foram poucas. Grande parte das informações era registrada por um dos membros do projeto, de acordo com as atividades e os resultados apresentados nas reuniões. De forma similar ao que acontece em ambiente industrial, notou-se que não houve entre os participantes a cultura de documentar, passo a passo, as atividades que cumprem. No geral, apenas os resultados finais de cada atividade foram apresentados na forma de um artefato, o que prejudicou o entendimento do projeto quando novos membros integraram-se à equipe.

No desenvolvimento de software livre, em geral, a principal preocupação consiste em gerenciar o código fonte, utilizando-se ferramentas de controle de versões como CVS [8] e Subversion [10]. Os desenvolvedores são voluntários e não há o papel do cliente que encomenda e paga pelo software. Apesar de ser comum a existência de um líder do projeto, delegando atribuições formalmente, não é primordial o gerenciamento de atividades em relação a prazos. Este cenário muda se o software é desenvolvido em ambiente acadêmico, em que o cumprimento de prazos e custos é fundamental e há o comprometimento por parte dos pesquisadores em entregar os artefatos resultantes do desenvolvimento do projeto. Ainda, o fato de haver um financiador impõe algumas características ao projeto que são distintas das do desenvolvimento de software livre.

> *Lições Aprendidas:* no desenvolvimento de projeto em ambiente acadêmico, em que relatórios técnicos precisam ser constantemente gerados contendo dados relacionados aos diversos recursos do projeto, o uso de um sistema de planejamento e gerenciamento trouxe muitos benefícios e, portanto, foi fundamental no contexto do projeto. A possibilidade de visualizar informações relacionadas ao cumprimento de prazos, o status e o responsável por cada atividade proposta foram as principais contribuições (um exemplo de gráfico gerado pela ferramenta utilizada é apresentado na Figura 2). Além disso, apesar de o registro das informações no sistema de gerenciamento não ter ocorrido de forma ideal, o cumprimento de atividades alocadas a cada participante pôde ser acompanhado, bem como informações como prioridade da atividade no contexto do projeto e taxa de finalização da atividade. Observa-se também que as informações resultantes de reuniões presenciais foram as mais consistentemente utilizadas.

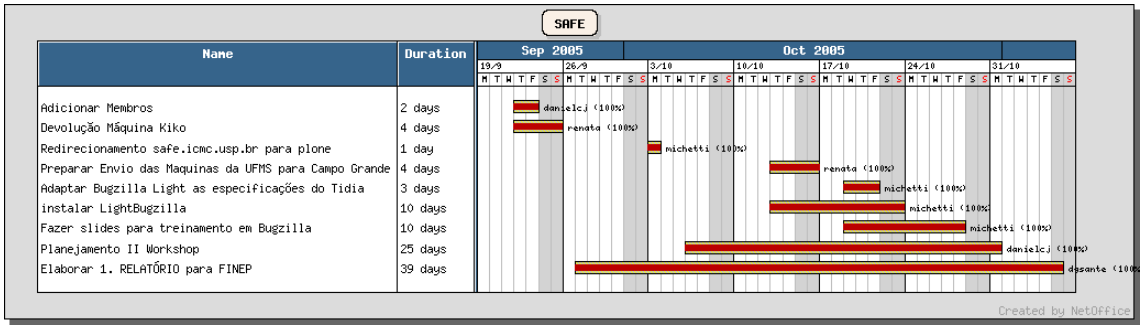

Figura 2: Um exemplo de gráfico gerado durante uso da ferramenta NetOffice

4.5 DOCUMENTAÇÃO DO PROJETO E USO DE FERRAMENTAS

Além dos artefatos gerados como resultado do ciclo de vida do projeto (Seção 4.2) optou-se pela elaboração de uma cartilha, contendo a descrição da pesquisa realizada no contexto do projeto. Na cartilha não foi enfatizado o sistema desenvolvido e sim as etapas cumpridas para a realização das pesquisas subjacentes ao projeto. Considerou-se a elaboração deste documento importante por tornar explícito o embasamento científico do projeto e outras questões relacionadas ao tema, por exemplo, como as fases de um projeto de software livre são geralmente cumpridas (estas informações são fundamentais para que os usuários possam entender o projeto e utilizar o sistema desenvolvido).

É importante mencionar também que ferramentas frequentemente utilizadas em projetos de software livre para apoiar o desenvolvimento, a comunicação entre os membros e a documentação do projeto estão sendo utilizadas com

[10]http://netoffice.sourceforge.net/ e http://safe.icmc.usp.br/netoffice/general/login.php

sucesso no contexto do projeto SAFE. Alguns exemplos são PHPDoc[11], CVS[12], Subversion[13], Plone[14], lista de discussão[15], IRC e Bugzilla[16]. O uso dessas ferramentas tem contribuído significativamente no contexto do projeto SAFE, principalmente por se tratar de um projeto desenvolvido de forma distribuída e colaborativa. Dois relatórios técnicos também foram gerados, visando apresentar os resultados parciais à agência financiadora e servir como documentação do projeto, pois descrevem as atividades cumpridas e os resultados obtidos, sob diferentes perspectivas.

> *Lições Aprendidas:* a documentação é uma necessidade percebida por desenvolvedores de projeto de software livre [19] e projetos de pesquisa [2]. Em desenvolvimento de software livre, há um esforço em documentar o software do ponto de vista funcional [19]. Em particular, tem sido notada a importância em gerar documentação sobre a arquitetura do sistema [15, 7], o que também ocorreu no contexto do projeto SAFE. A documentação por meio de relatórios técnicos e publicação da comunicação entre os membros por meio de uma wiki foram base para o desenvolvimento de um projeto colaborativo. O uso de ferramentas de software livre é completamente viável, além de agregar muito valor ao projeto e promover o desenvolvimento distribuído e colaborativo.

5 A EXPERIÊNCIA COM O CUMPRIMENTO DE UM PROCESSO DISTRIBUÍDO

Uma das principais características de projetos de software livre e de projetos de pesquisa é que o desenvolvimento ocorre de forma distribuída. A partir das atividades descritas na Seção 4 e dos elementos identificados por Maidantchik [16], são apresentados a seguir resultados da experiência obtida no contexto do projeto SAFE em relação à característica de distribuição. Estas experiências podem nortear o desenvolvimento de outros projetos que também explorem esta perspectiva.

1. *Determinação da capacidade das equipes:* durante a fase de elaboração do projeto foram coletadas e registradas informações sobre formação e área de atuação de cada membro, incluindo titulação, competências e experiências de cada um dos membros e disponibilidade para participação no projeto (número de horas por semana, atividades que poderiam colaborar).

2. *Apoio à coordenação distribuída:* cada equipe foi supervisionada localmente por um coordenador que era um aluno de pós graduação. A coordenação local não foi atribuída a apenas uma pessoa, ou seja, diferentes membros atuaram como coordenador durante o desenvolvimento do projeto. A definição das responsabilidades de cada participante foi realizada em conjunto com a coordenadora do projeto. Para acompanhar o cumprimento das tarefas foram realizadas reuniões periodicamente e as mudanças no projeto, decisões e problemas discutidos foram registrados em atas.

3. *Apoio à gerência distribuída do projeto:* O uso da ferramenta NetOffice apoiou a gerência distribuída do projeto. Apesar do uso restrito pelos participantes, as informações eram atualizadas a medida que os resultados eram apresentados em reuniões, quando as atividades eram explicitamente atribuídas ou quando os artefatos eram finalizados. A supervisão das atividades cumpridas foi realizada principalmente por meio da revisão dos artefatos gerados durante o desenvolvimento do projeto. Pontos de controle do projeto não foram explicitamente definidos, porém, a realização dos workshops e os *deadlines* para submissão de relatórios técnicos serviram como importantes marcos do projeto.

4. *Apoio ao controle de artefatos:* a qualidade dos artefatos era avaliada por meio de revisões, que ocorriam quando resultados intermediários eram obtidos ou quando os artefatos eram finalizados. O controle de versões dos artefatos produzidos pelas equipes foi realizado utilizando-se uma das ferramentas em estudo no projeto, a ferramenta Subversion. Não foi definido um processo detalhado para controle dos artefatos envolvendo planejamento, monitoração e notificação de resultados, mudaças, inconsistências e dependências. Foi adotada apenas uma política de permissão de acesso aos servidores e às bases de dados das ferramentas.

[11] http://notsafe.icmc.usp.br/docrat/ta_ma/documentacao/
[12] http://notsafe.icmc.usp.br/versionweb/
[13] http://notsafe.icmc.usp.br/websvn/
[14] http://safe.icmc.usp.br/safe
[15] safe-devel at async.com.br
[16] http://notsafe.icmc.usp.br/bugzilla/ e http://notsafe.icmc.usp.br/bugzilla_light/

5. *Apoio à comunicação entre as equipes:* os principais recursos utilizados para promover a comunicação entre os membros foram emails, lista de discussão e mensagens instantâneas. Durante todo o desenvolvimento do projeto, principalmente durante o cumprimento das atividades do ciclo de vida, houve a possibilidade de utilizar os recursos de comunicação da ferramenta Bugzilla.

6. *Apoio à publicação e ao compartilhamento de informações:* a publicação de informações para a comunidade externa aconteceu por meio da realização de mini-cursos, cursos de extensão e apresentação do projeto em eventos científicos, na forma de apresentação de artigos e de exposição do projeto em outros tipos de reuniões científicas, por exemplo, em mesa redonda em simpósio. Entre os membros da equipe foram utilizados também os recursos "Events", "News" e "Meetings" do ambiente Plone para divulgar reuniões, atas, mudanças no projeto e resultados alcançados. No contexto do projeto SAFE cada participante possui sua área de trabalho no plone e, portanto, as tarefas cumpridas e os resultados obtidos são constantemente registrados e disponibilizados aos participantes do projeto. Além disso, em determinadas ocasiões, o próprio sistema solicita e torna explícita as informações relacionadas a "o que", "quando", "onde" e histório, conforme ilustrado na Figura 3. A apresentação destas informações contribuem para a identificação do contexto no qual os eventos ocorrem.

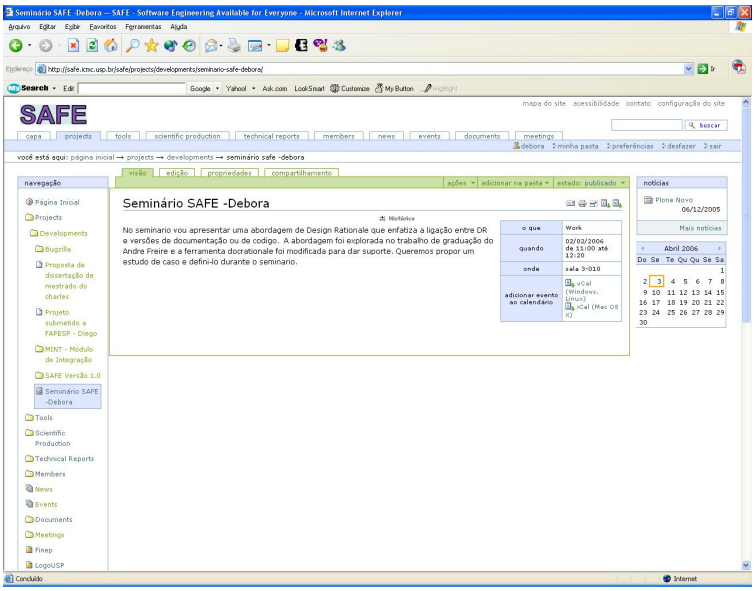

Figura 3: Exemplo de uso do plone no projeto SAFE

7. *Apoio à resolução de problemas:* não foram incorporadas ao processo atividades para notificar a existência de um problema, divulgar a sua resolução e identificar as possíveis implicações no trabalho das equipes. As notificações eram realizadas informalmente, durante a realização de reuniões, ou a partir de resultados das discussões realizadas nos workshops.

8. *Incorporação de um repositório de terminologias:* não foi documentado um glossário sobre o domínio da aplicação do projeto, de forma a possibilitar a compreensão única dos termos pelas equipes envolvidas. Ao invés disso, no início do projeto foram realizadas diversas reuniões entre os membros com o objetivo de explicar os principais termos da área para a equipe. Essas reuniões foram muito importantes em termos técnicos pois favoreceu a familiarização da equipe com o tema de software livre. É importante mencionar também que, apesar de não haver um glossário que fosse atualizado constantemente, o documento gerado por Reis [19] foi utilizado como uma fonte bibliográfica importante em que a definição de muitos termos foi apresentada.

9. *Apoio à redefinição de atividades de software:* conforme apresentado na Seção 4.2, as atividades do processo foram cumpridas de forma iterativa, no entanto, não foi documentado formalmente um processo que devesse ser seguido por todos os membros (ou adaptado de acordo com as características e restrições das equipes). Quanto à infra-estrutura necessária para a realização de tarefas de forma distribuída foram disponibilizados sistema de gerência de versões (Subversion), controle de alterações (Bugzilla) e gerenciamento (NetOffice).

6 CONCLUSÕES

Durante o desenvolvimento do projeto SAFE foi observado que características, restrições e demandas do contexto do desenvolvimento de pesquisas e do desenvolvimento de software livre se complementaram de forma a compor o processo de desenvolvimento que tem sido utilizado. As lições aprendidas foram descritas neste artigo, sob a perspectiva do processo de software que está sendo cumprido.

As principais lições aprendidas são resumidas a seguir:

- definir e expressar claramente os objetivos, a metodologia e os resultados esperados dos projetos ressaltando sua relevância e impacto na sociedade;
- utilizar uma abordagem iterativa e incremental, que permita que os requisitos do sistema sejam "descobertos" na medida em que o projeto evolui;
- realizar encontros presenciais, registrar resultados das discussões, elaborar material didático e realizar cursos de extensão;
- utilizar um sistema para planejamento e gerenciamento do projeto; e
- documentar os projetos de acordo com a necessidade de cada projeto.

A experiência obtida com o desenvolvimento do projeto SAFE foi bastante enriquecedora e certamente abordagens semelhantes serão utilizadas pelo grupo no desenvolvimento de novos projetos. Ressalta-se, em especial, a motivação da equipe em melhorar o processo apresentado, tendo em vista a perspectiva da disciplina de melhoria de processos de software.

Agradecimentos

Agradecemos à FINEP que finacia o Projeto SAFE e à FAPESP, à CAPES e à FUNDECT, pelo apoio ao desenvolvimento das pesquisas no desenvolvimento das ferramentas.

Referências

[1] Lara S. M. A. and Fortes R. P. M. An Ubiquitous Mechanism to Capture Design Rationale of Software Artifacts. In *4th International Information and Telecommunication Technologies Symposium*, 2005. In portuguese.

[2] V. Ambati and S. P. Kishore. How can Academic Software Research and Open Source Software Development Help Each Other? In *4th Workshop on Open Source Software Engineering, 26th International Conference on Software Engineering*, pages 5–8, 2004.

[3] C. R. E. Arruda Jr., C. A. Izeki, and M. G. C. Pimentel. CoTeia: Uma Ferramenta Colaborativa de Edição Baseada na Web. *Workshop de Ferramentas e Aplicações do VIII SBMIDIA*, 2002.

[4] V. Bellotti, R. Burton, N. Ducheneaut, M. Howard, C. Neuwirth, and I. Smith. Xp in a research lab: The hunt for strategic value. In *Proceedings of the Third International Conference on eXtreme Programming (XP2002)*, pages 56–61, 2002.

[5] N. Bezroukov. Open Source Software Development as a Special Type of Academic Research. *First Monday – peer-reviewed journal on the internet*, 4(10), 1999.

[6] C. Boldyreff, D. Nutter, and S. Rank. Communication and Conflict Issues in Collaborative Software Research Projects. In *4th Workshop on Open Source Software Engineering, 26th International Conference on Software Engineering*, pages 14–17, 2004.

[7] I. T. Bowman, R. C. Holt, and N. V. Brewster. Linux as a Case Study: Its Extracted Software Architecture. In *Proceedings of the 21st International Conference on Software Engineering*, 1999.

[8] Cederqvist et al. *Version Managment with CVS*, 2006. Disponível online em: `http://ximbiot.com/cvs/manual/`.

[9] O. Chirouze, D. Cleary, and G. G. Mitchell. A Software Methodology for Applied Research: eXtreme Researching. *Software – Practice and Experience*, 35(15):1441–1454, 2005.

[10] Ben Collins-Sussman, Brian W. Fitzpatrick, and C. Michael Pilato. *Version Control with Subversion*. O'Reilly, draft 9837 edition, 2004. Livro online, disponível em `http://svnbook.red-bean.com/svnbook/book.html`.

[11] W. P. Paula Filho. Requirements for an Educational Software Development Process. In *Proceedings of The 6th Annual Conference on Innovation and Technology in Computer Science Education*, 2001.

[12] R. P. M. Fortes, A. P. Freire, V. H. Vieira, and D. M. B. Paiva. An Academic Web-Based Agenda and its Engineering Process. In *VII Workshop Ibero Americano de Ingeniería de Requisitos Y Desarrollo de Ambientes de Software*, pages 151–156, 2004.

[13] R. P. M. Fortes, M. A. G. Silva, A. P. Freire, and D. C. Junqueira. SAFE – Software Engineering Available For Everyone. In *V Workshop sobre Software Livre*, pages 203–206, 2004.

[14] S. D. Francisco, C. A. Izeki, D. M. B. Paiva, and R. P. M. Fortes. A System for Supporting Design Rationale of Software Artifacts. In *XXIX Latin-American Conference on Informatics*, 2003. In portuguese.

[15] M. W. Godfrey and E. H. S. Lee. Secrets from the Monster: Extracting Mozilla's Software Architecture. In *Proc. of the Second International Symposium on Constructing Software Engineering Tools*, 2000.

[16] C. L. L. Maidantchik. *Gerência de Processos de Software para Equipes Geograficamente Dispersas*. PhD thesis, Universidade Federal do Rio de Janeiro, 1999.

[17] Paula Filho W. P. An Educational Software Development Process. In *Proceedings of the ACIS International Conference on Computer Science, Software Engineering, Information Technology, E-Business and Applications (CSITeA'02)*, 2002.

[18] D. M. B. Paiva, A. M. Manduca, M. A. G. Silva, L. J. Quemello, R. T. V. Braga, and R. P. M. Fortes. Reforçando a Comunicação com Uso de uma Ferramenta de Software Livre em Ensino de Engenharia de Software. In *XIII Workshop sobre Educação em Computação, XXV Congresso da Sociedade Brasileira de Computação*, 2005.

[19] Reis C. R. Caracterização de um Processo de Software para Projetos de Software Livre. Master's thesis, Instituto de Ciências Matemáticas e de Computação - Universidade de São Paulo, 2003.

[20] W. C. Regli, X. Hu, M. Atwood, and W. Sun. A Survey of Design Rationale Systems: Approaches, Representation, Capture and Retrieval. *Engineering with Computers: An International Journal for Simulation-Based Engineering, special issue on Computer Aided Engineering in Honor of Professor Steven J. Fenves*, 16:209–235, 2000.

[21] P. N. Robillard and M. Robillard. Improving Academic Software Engineering Projects: A Comparative Study of Academic and Industry Projects. *Annals of Software Engineering*, 6:343–363, 1998.

[22] J. Segal. When Software Engineers Met Research Scientists: A Case Study. *Empirical Software Engineering*, 10(4):517–536, 2005.

[23] M. D. Soares and R. P. M. Fortes. Um Suporte ao Controle de Versões na Web. In *Anais do Caderno de Ferramentas do Simpósio Brasileiro de Engenharia de Software (SBES'2001)*, Rio de Janeiro, 2001.

[24] M. D. Soares, R. P. M. Fortes, and D. A. Moreira. VersionWeb: A Tool for Helping Web Pages Version Control. In *Proceedings of the International Conference on Internet Multimedia Systems and Applications (IMSA 2000)*, pages 275–280, Las Vegas USA, November 2000.

[25] G. Succi and P. Zuliani. Exploiting the Collaboration between Open Source Developers and Research. In *4th Workshop on Open Source Software Engineering, 26th International Conference on Software Engineering*, pages 97–99, 2004.

[26] AJ Walker. Level 5 Process Capability Achievement: a Case Study from Software Engineering Research Management. *Software Process: Improvement and Practice*, 8(1), 2003.