Definição de um Processo Padrão para Software Livre

Bruno Carreira Coutinho Silva, Ricardo de Almeida Falbo

Mestrado em Informática – Universidade Federal do Espírito Santo (UFES)
Av. Fernando Ferrari s/n, Campus de Goiabeiras – 29.060-900 – Vitória – ES – Brasil
brunocarreira@gmail.com , falbo@inf.ufes.br

Abstract. *Free Software is more and more earning space in software market. Nowadays, there are several projects of this kind in progress around the world. This new software development model brings along a new philosophy, affecting many of the software industry principles. Despite of its importance and growth, in most cases, free software development is not being done according to the best practices of Software Engineering. In this scenario, many times software processes are not formally defined. This paper discusses an effort for defining a standard process for free software projects at LabES/UFES. The initial goal of defining these processes is to apply it in ODE´s Project, a project that aims to develop a software engineering environment as a free software.*

Resumo. *O movimento de Software Livre vem ganhando cada vez mais espaço no mercado, sendo que atualmente já existem milhares de projetos desse tipo em andamento no mundo inteiro. Esse novo modelo de desenvolvimento de software traz consigo uma nova filosofia, afetando muitos dos atuais princípios da indústria de software. Apesar de sua importância e crescimento, na maioria das vezes, seu desenvolvimento não vem sendo feito segundo as melhores práticas da Engenharia de Software, incluindo nesse cenário a não utilização de processos de software bem definidos. Neste trabalho, é discutida a iniciativa de se definir um processo padrão para projetos de software livre desenvolvidos no LabES/UFES. O objetivo inicial da definição desse processo é aplicá-lo no projeto do ambiente de desenvolvimento de software ODE.*

1. Introdução

O modelo de Software Livre (SL) tem despertado interesse e suscitado reflexões nos mais diversos segmentos da comunidade de software (governo, academia, indústria etc) no Brasil e no exterior. O surgimento de uma rede virtual de desenvolvedores e usuários, complexa, auto-organizada, com motivações diversas, e a existência de novas formas de licenciamento de software sinalizam a introdução de novas variáveis no setor de software [Softex, 2005a].

A recente pesquisa realizada pelo Observatório Econômico da Softex e o Departamento de Política Científica e Tecnológica da UNICAMP, com o apoio do Ministério de Ciência e Tecnologia [Softex, 2005a], indica que, apesar de não se tratar de uma ruptura tecnológica, o modelo de SL traz uma nova forma de desenvolver e licenciar software que está quebrando modelos tradicionais de apropriabilidade e de desenvolvimento tecnológico.

O movimento de SL foi formalizado em 1984, quando Richard Stallman iniciou um projeto para criar uma versão do sistema operacional Unix, livre de restrições, publicando o Manifesto GNU e, posteriormente, fundando a *Free Software Foundation* – FSF [Reis, 2003]. Contudo, somente mais recentemente esse movimento ganhou maior divulgação.

Os mais de 20 anos de evolução permitiram ao modelo de SL avançar em diversos aspectos: técnico, político-estratégico, de adequação às necessidades dos usuários, de qualidade, segurança etc. Essa evolução é resultado de um conjunto heterogêneo de fatores e se trata, na verdade, de um processo evolutivo, cujos caminhos ainda estão sendo trilhados, envolvendo o desenvolvimento e a manutenção de software (e de material relacionado, como documentação), difusão, estímulo e apoio ao uso, chegando até a uma visão empresarial, que encontra no modelo de SL uma nova opção de negócios [Softex, 2005a].

Apesar de sua importância e crescimento, muitas vezes, projetos de SL não são conduzidos observando-se as melhores práticas da Engenharia de Software, como advoga a área de estudos da Qualidade de Software, incluindo-se neste cenário a pouca utilização de processos de software bem definidos. Contudo, esse aspecto é apontado sistematicamente como condição *se ne qua non* para a obtenção de produtos de software de qualidade. Assim, esse é um assunto bastante relevante para a consolidação do movimento de SL, sobretudo porque os processos de software utilizados no seu desenvolvimento tendem a ser diferentes daqueles comumente utilizados no desenvolvimento da maioria dos projetos de software não livre [Nakagawa, 2002].

Este trabalho discute os principais aspectos considerados no esforço realizado para se definir um processo padrão para projetos de software livre a serem desenvolvidos sob controle do Laboratório de Engenharia de Software da Universidade Federal do Espírito Santo (LabES/UFES). O objetivo inicial da definição desses processos é aplicá-los no projeto do ambiente de desenvolvimento de software ODE [Falbo et al., 2003], a ser disponibilizado em breve como software livre, denominado Projeto ODE Livre.

Este artigo está estruturado da seguinte forma: a seção 2 conceitua SL e apresenta características importantes de processos de software para o seu desenvolvimento; a seção 3 apresenta as principais características do LabES/UFES, organização para a qual foi definido o processo padrão discutido neste artigo, e do Projeto ODE Livre, motivador para essa iniciativa; a seção 4 discute os principais aspectos considerados na definição do processo padrão para SL; a seção 5 discute trabalhos correlatos; e finalmente, na seção 6, são apresentadas as conclusões e perspectivas futuras de continuidade deste trabalho.

2. Software Livre e Processo de Software

Software Livre - SL (*Free Software*) é o software disponível com a permissão para qualquer um usá-lo, copiá-lo e distribuí-lo, seja na sua forma original ou com modificações, seja gratuitamente ou com custo. Em especial, a possibilidade de modificações implica que o código fonte esteja disponível [Hexsel, 2002].

SL é uma questão de liberdade, não de preço. Mais precisamente, se refere a quatro liberdades para seus usuários, a saber [Free Software Foundation, 2005]:

0. Liberdade de executar o programa, para qualquer propósito;
1. Liberdade de estudar como o mesmo funciona e adaptá-lo para as suas necessidades;
2. Liberdade de redistribuir cópias de modo a ajudar ao próximo;
3. Liberdade de aperfeiçoar o programa e liberar os seus aperfeiçoamentos, de modo que toda a comunidade se beneficie.

As liberdades 1 e 3 implicam na necessidade de se ter, pelo menos, acesso ao código-fonte, apontado como a principal característica dos projetos de software livre.

Ainda que a idéia de software livre remonte às décadas de 60 e 70, quando grande parte do código fonte era distribuída aberta e gratuitamente, somente na década de 80 a idéia de SL se popularizou, com a criação da *Free Software Foundation – FSF*, por Richard Stallman. O movimento de SL nasceu de uma contestação aos mercados proprietários mais poderosos da indústria de software, revelando todo um apelo político, institucional e emocional [Softex, 2005a], e é hoje um movimento sólido.

Produtos de Software livres apresentam algumas características vantajosas peculiares, muitas vezes, resultantes do próprio processo de desenvolvimento dessa classe de produtos de software [Nakagawa, 2002], incluindo estabilidade, portabilidade para uma variedade de plataformas, acesso ao código-fonte, suporte aos usuários por parte dos desenvolvedores, baixo custo, e evolução contínua, com versões mais novas do software sendo lançadas com maior freqüência e defeitos sendo corrigidos em um tempo menor do que em projetos de software proprietário [Davis et al., 2000]. Pelo seu caráter cooperativo, propiciado principalmente pela distribuição do código fonte, o desenvolvimento de SL é feito por colaboradores dispersos geograficamente, muitas vezes, ao redor do mundo. Por todas essas características, SL exige uma abordagem de desenvolvimento diferente da tradicionalmente empregada.

Raymond (1999) propôs a adoção de um modelo denominado Modelo Bazar para o desenvolvimento de SL, incluindo características peculiares, tais como: disponibilização de código fonte; envolvimento de diversos desenvolvedores, muitas vezes, voluntários; dispersão geográfica dos mesmos em um processo colaborativo; tempo curto de desenvolvimento; e liberdade para os desenvolvedores escolherem o trabalho que desejam realizar. Vale apontar que esse modelo requer uma pessoa (ou grupo de pessoas) centralizando o processo.

O Modelo Bazar tem sido aplicado com sucesso em diversos projetos, tais como os projetos do Linux e do Apache. Observa-se, contudo, que para se obter sucesso na sua utilização, é necessário, ainda, ter [Nakagawa, 2002]: um objetivo bem definido do que será desenvolvido; uma motivação fácil de ser compreendida; um bom líder que mantenha os desenvolvedores motivados e o objetivo em mente; uma comunidade de participantes que trabalhe com entusiasmo e de forma descentralizada; e uma tecnologia que possibilite a comunicação de forma eficiente. Assim, o sucesso do Modelo Bazar não depende somente de disponibilização do código fonte, sendo necessário organizar e coordenar todo o processo de forma apropriada.

Apesar de interessante e muito útil, o Modelo Bazar é apenas uma referência inicial para um processo de software livre, principalmente quando confrontado com normas e modelos de qualidade, tal como a ISO/IEC 12207. Sob a ótica dessas normas e modelos, um processo de software pode ser definido como um conjunto de atividades, métodos, práticas, estruturas organizacionais, tecnologias e artefatos que são necessários para conceber, desenvolver, entregar e manter um produto de software [Fuggeta, 2000]. Assim, um processo eficaz deve, claramente, considerar as relações entre as atividades, os artefatos produzidos no desenvolvimento, as ferramentas e os procedimentos necessários e a habilidade, o treinamento e a motivação do pessoal envolvido.

Claramente, o Modelo Bazar não chega a esse nível de definição. Na verdade, Raymond (1999) estava focado nas estruturas organizacionais e características de um projeto de SL, não estando diretamente preocupado com atividades, artefatos e procedimentos que um processo deveria seguir. Até porque, processos têm de ser definidos caso a caso, levando em consideração características específicas do projeto em questão, incluindo as tecnologias envolvidas, o tipo de software a ser desenvolvido, o domínio da aplicação e a equipe de desenvolvimento [Rocha et al., 2001].

Entretanto, ainda que diferentes projetos demandem processos com características específicas para atender às suas particularidades, é senso comum na área de processo de software que é possível estabelecer um conjunto de ativos de processo (atividades, artefatos, recursos e procedimentos) que devem fazer parte de todos os processos de uma organização, dito processo padrão da organização [Rocha et al., 2001]. Assim, entendemos que pode ser útil adotar essa filosofia para um projeto de SL, em que cada sub-projeto pode definir um processo de manutenção ou desenvolvimento, tomando por base processos padrão da organização que, em última instância, controlam o projeto. Antes de passarmos a discutir essa abordagem, é útil examinar características já relatadas de projetos e processos de SL.

Reis (2003) procurou levantar e quantificar aspectos essenciais do processo de software utilizado nos projetos de software livre, discutindo questões como: De que forma os projetos de software livre trabalham para produzir software? Existe um processo comum entre os projetos de software livre? De que forma estes processos diferem do processo de software convencional documentado na literatura de engenharia de software? Dentre as conclusões a que ele chegou em sua pesquisa, podemos destacar:

C1. Na maioria dos projetos, a equipe trabalha de maneira descentralizada, envolvendo, muitas vezes, desenvolvedores que não se conhecem pessoalmente. Vale destacar, ainda, que nos últimos anos estão ocorrendo mudanças neste perfil, com organizações investindo na contratação de desenvolvedores para projetos de software livre.

C2. Em grande parte dos projetos, os desenvolvedores são também usuários do produto em desenvolvimento e contribuem efetivamente para determinar sua funcionalidade.

C3. Existe, na maior parte dos projetos, um sistema de liderança, sendo as formas mais utilizadas: líder com delegação informal, líder com delegação formal e comitê.

C4. Há uso amplo de ferramentas para viabilizar a comunicação entre membros da equipe geograficamente dispersa, com destaque para ferramentas de controle de

versão e listas de correio eletrônico, além de sistemas de acompanhamento de alteração / defeitos, serviço de hospedagem de projetos e ferramentas de comunicação em tempo real. No que tange a ferramentas de desenvolvimento, sua utilização aparece majoritariamente centrada em ferramentas de implementação.

C5. A maior parte dos projetos possui equipe pequena.

C6. Há indícios fortes do uso de conhecimento pré-existente para estabelecer os requisitos do projeto, seja por meio da reutilização de levantamentos anteriores feitos por outras equipes, através de padrões estabelecidos ou documentados, seja por replicar de maneira significativa um outro produto de software.

C7. Parece existir muito pouco processo sistemático em relação à qualidade, incluindo testes. Não há evidências de uma política forte de controle e revisão relacionada à aceitação, integração e auditoria de contribuições. Contudo, ter seu trabalho constantemente sob olhar público, parece forçar os desenvolvedores a encararem sua tarefa com atenção e padrão de qualidade superiores. Assim, mesmo que o código não seja revisado e auditado com freqüência, o fato de poder ser avaliado – e publicamente criticado – parece influenciar significativamente a atitude do desenvolvedor na construção de SL.

C8. Pouca atenção é dada à usabilidade do software, o que, muitas vezes, leva software livre a ter uma reputação de complexo ou difícil de usar.

Além das conclusões de Reis sobre características de projetos de software livre, vale a pena destacar outras considerações relatadas na literatura:

C9. À primeira vista não é dada muita ênfase à especificação de requisitos [Scacchi, 2002], seja porque a maioria dos projetos de software livre replica um software já existente ou por partirem de uma motivação pessoal do autor, trazendo requisitos implícitos [Raymound, 1999]. O conceito de um "documento de especificação de requisitos" parece raro entre a comunidade de software livre, embora, existam especificações formais descrevendo partes restritas de alguns sistemas mais complexos [Reis, 2003].

C10. Na fase de projeto, os modelos são de mais alto nível, não sendo possível identificar uma arquitetura bem definida do sistema [Vixie, 1999]. Segundo Reis (2003), existe pouca documentação formal de projeto e é provável que grande parte do trabalho de descrever o projeto fique a cargo do próprio código-fonte.

C11. A melhor estratégia de lançamento de software livre é disponibilizá-lo o quanto antes, de modo que ocorra um crescimento evolutivo por meio de sugestões e críticas dos usuários. Contudo, para que o software seja lançado, deve apresentar funcionalidade interessante o suficiente para atrair atenção [Raymound, 1999].

Apesar de haver pesquisas no sentido de entender e sistematizar o processo de SL, como as comentadas anteriormente, não está ainda definitivamente estabelecido um processo para esta classe de software, sobretudo nos moldes apregoados pelos modelos e normas de qualidade de processo. Assim, no contexto do LabES/UFES, foi definido um processo padrão para SL, a ser aplicado no Projeto ODE Livre. Na próxima seção, uma breve descrição dessa organização e do projeto é feita, com vistas a apontar o porquê de algumas decisões tomadas na definição do processo, discutida na seção 4.

3. O LabES/UFES e o Projeto ODE Livre

O Laboratório de Engenharia de Software (LabES) da Universidade Federal do Espírito Santo (UFES) é um laboratório de pesquisa e ensino em Engenharia de Software, no qual há vários projetos de software em andamento, sobretudo no contexto de projetos de pesquisa, ainda que alguns projetos sejam contratados por organizações externas.

Visando colocar em prática o que é ensinado, o LabES adota uma política de desenvolvimento de software com qualidade, seguindo o preceito de que a qualidade do produto pode ser mais facilmente obtida se o mesmo for desenvolvido segundo um processo de qualidade. Assim, tomando por base modelos e normas de qualidade, principalmente as normas ISO/IEC 12207 [ISO, 1995], IEEE 1219 [IEEE, 1998] e o modelo de referência do MPS.BR [Softex, 2005b], foi definido um processo padrão para o LabES. Além desse processo, seguindo uma abordagem de definição de processos em níveis [Rocha et al., 2001], foram definidos processos especializados para os paradigmas orientado a objetos e estruturado, seguindo a abordagem mostrada na Figura 1. Essa abordagem se caracteriza pela existência de um processo padrão da organização, processos padrão especializados a partir do processo padrão da organização e processos instanciados ou de projeto. No caso do LabES, o processo raiz é o Processo Padrão LabES (PPLabES). A partir dele, são especializados os Processo Padrão LabES Orientado a Objetos (PPELabES-OO) e Estruturado (PPELabES-Est), a partir dos quais são instanciados os processos de projeto.

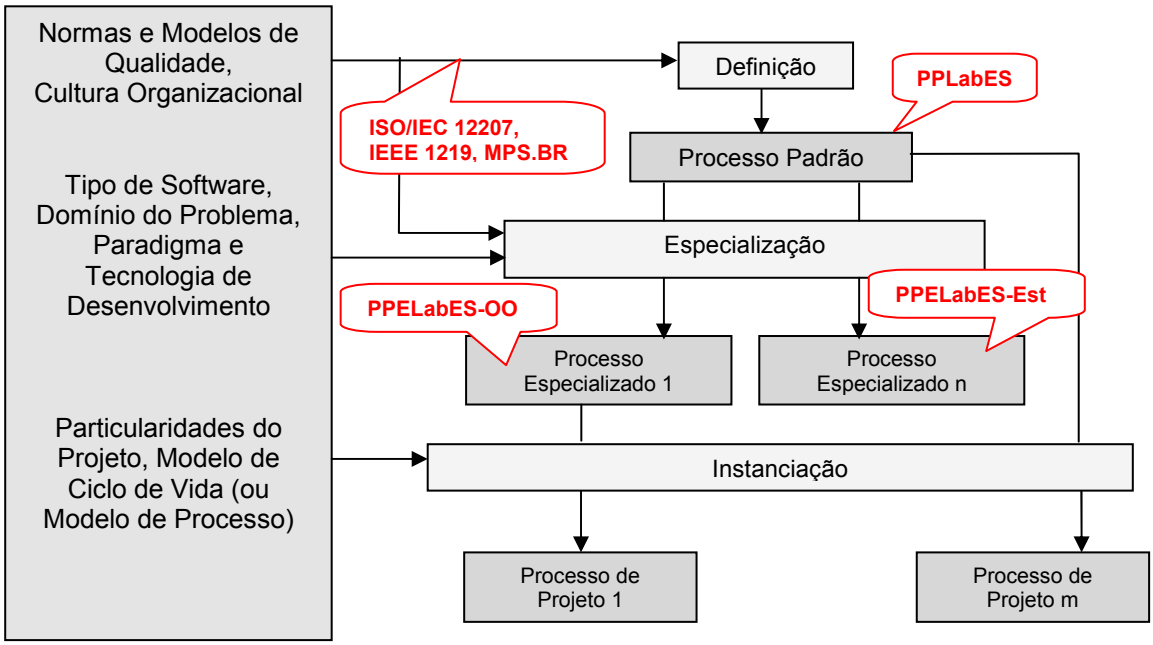

Figura 1 – Abordagem de Definição de Processos em Níveis.

Dentre os projetos do LabES, destaca-se o Projeto ODE [Falbo et al., 2003], que visa ao desenvolvimento de um Ambiente de Desenvolvimento de Software (ADS) centrado em processo. Um ADS pode ser definido como uma coleção de ferramentas integradas que apóiam as atividades de engenharia de software durante todo o ciclo de vida do software ou, pelo menos, em porções significativas dele [Harrison et al., 2000]. Um ADS centrado em processo é um ADS que suporta a definição de processos de

software e se utiliza desta para estabelecer uma ligação explícita entre as ferramentas do ambiente e os processos definidos.

O Projeto ODE teve seu início em 1999, motivado pela existência de poucos ambientes deste tipo disponíveis. ODE vem sendo desenvolvido no meio acadêmico, sendo o resultado do esforço conjunto de vários alunos de graduação e mestrado. Com o amadurecimento do trabalho, em 2004, foi possível experimentar sua aplicação prática. Para tal, foi feita uma parceria com uma organização de desenvolvimento de software para que ODE pudesse ser testado num ambiente corporativo. Dessa interação universidade-empresa, diversas sugestões foram dadas e uma nova versão de ODE foi desenvolvida e implantada em março de 2006 para uso efetivo nesta organização. Nesta versão, ODE possui as seguintes ferramentas: controle de projetos, definição do escopo de projetos, definição de processos (incluindo processos padrão, especializados e de projeto), apoio à alocação de recursos, apoio à realização de estimativas, apoio à gerência de riscos e apoio à gerência de requisitos. Outras ferramentas, já desenvolvidas ou em desenvolvimento, estão integradas a ODE e há a previsão para a distribuição de uma nova versão do ambiente em julho de 2006, incluindo um sistema de gerência de configuração, um sistema gerência de conhecimento, uma ferramenta de apoio à garantia da qualidade e uma ferramenta de apoio à documentação.

Pode-se notar, pela variada gama de ferramentas, que o projeto de um ADS não é uma tarefa simples. Há muitas ferramentas potencialmente úteis a serem desenvolvidas e, além disso, organizações diferentes podem necessitar customizar algumas dessas ferramentas para melhor adequá-las à sua forma de trabalho. Assim, o modelo de software livre passou a ser considerado potencialmente interessante para a evolução do projeto, dando origem ao Projeto ODE Livre. Pensando ODE como software livre, a comunidade poderia se privilegiar de um ADS poderoso e, em contrapartida, o LabES contaria com a ajuda da comunidade para torná-lo um ambiente mais completo e que atendesse melhor às necessidades de várias organizações.

Conforme discutido na seção anterior, muitas das características requeridas para o sucesso de um projeto de software livre se aplicam ao Projeto ODE, tais como: (i) usuários do produto podem contribuir efetivamente para determinar sua funcionalidade, podendo vir a se interessar em se tornar também desenvolvedores (C2); (ii) a liderança do projeto pode ser exercida pelo LabES, de forma compartilhada por seus membros, ficando um membro ou um grupo de pessoas responsável por analisar as contribuições enviadas para uma ferramenta ou funcionalidade do sistema (C3); (iii) o ambiente ODE já tem funcionalidade interessante o suficiente para atrair a atenção da comunidade de desenvolvimento de software (C11), como pode ser comprovado pela sua implantação em uma organização de desenvolvimento de software; (iv) o LabES continuará fornecendo desenvolvedores para o projeto, sendo importante força de trabalho para o projeto, mas novas instituições, sejam de ensino sejam de mercado, poderão se engajar ao projeto (C1). Por exemplo, muitos ex-integrantes do LabES podem levar o ambiente para suas novas organizações, tornando-se colaboradores.

Contudo, algumas características do Projeto ODE Livre são um pouco diferentes da maioria dos projetos de software livre. Primeiro, como há muito poucos ambientes deste tipo disponíveis, geralmente de difícil acesso, bem como não há padrões documentados para a maior parte das ferramentas de um ADS (C6), atenção especial

tem de ser dada à atividade de levantamento de requisitos, inclusive no que tange à elaboração de um documento formal de especificação de requisitos (C9). Segundo, usabilidade (C8) é uma característica de qualidade essencial para um ADS. Assim, esse aspecto tem que ser cuidadosamente tratado. De fato, por um ADS estar muito em linha com o desenvolvimento de software de qualidade, é natural que o Projeto ODE tenha um cuidado especial em relação à qualidade (C7) e, portanto, padrões devem ser estabelecidos e deverá ser verificada a conformidade das contribuições a esses padrões. Por fim, dado que um ADS é um sistema muito complexo, em que a integração de ferramentas é uma questão fundamental, a documentação de sua arquitetura e do projeto de suas ferramentas é vital para a evolução do ambiente (C10).

Tendo em vista as características que aproximam e afastam o Projeto ODE da prática corrente do movimento de software livre, procurou-se chegar a uma abordagem própria para o desenvolvimento do Projeto ODE Livre. Essa abordagem consiste na definição de processos padrão de desenvolvimento e manutenção de software livre a serem seguidos pelos colaboradores do projeto. Esses processos foram definidos tomando por base os processos padrão do LabES, seguindo a abordagem de definição de processos em níveis, ilustrada na Figura 2. Como o LabES vem trabalhando sistematicamente com o desenvolvimento segundo o paradigma orientado a objetos, os processos padrão de desenvolvimento e manutenção de software livre foram definidos como especializações dos processos já especializados para esse paradigma (PPELabES-OO). A idéia é que, para que uma contribuição seja incorporada ao ambiente ODE, ela terá de ser desenvolvida segundo um processo em conformidade com os processos padrão de software livre do LabES. Ou seja, processos têm que ser instanciados a partir dele, para cada sub-projeto de contribuição, levando-se em conta as características desse sub-projeto e da organização colaboradora. Na seção que se segue, discutimos as principais características do processo padrão especializado para software livre definido e os fatores que motivaram a inclusão dessas características no processo.

4. A Definição do Processo Padrão Especializado LabES para Software Livre

O estudo para a definição do Processo Padrão Especializado LabES para Software Livre (PPELabES-SL) começou por um exame de características relevantes de produtos livres e por uma avaliação de quais processos do MPS.BR se aplicariam a um processo desse tipo. A seguir, procurou-se encontrar no Processo Padrão do LabES (PPLabES) os elementos de processo correspondentes, visando a compatibilidade. Tendo em vista que é um objetivo do LabES manter seus processos em linha com as diretrizes do MPS.BR, a definição do PPELabES-SL serviu de base, também, para a melhoria do PPLabES, que passou a incorporar algumas das recomendações do MPS.BR ainda não tratadas.

O modelo de referência do MPS.BR [Softex, 2005b] define 23 processos, agrupados em três categorias:

- Processos Fundamentais: Aquisição, Gerência de Requisitos, Desenvolvimento de Requisitos, Solução Técnica, Integração do Produto, Instalação do Produto e Liberação do Produto.
- Processos de Apoio: Garantia da Qualidade, Gerência de Configuração, Validação, Verificação, Medição e Treinamento.

- Processos Organizacionais: Gerência de Projeto, Adaptação do Processo para Gerência de Projeto, Análise de Decisão e Resolução, Gerência de Riscos, Avaliação e Melhoria do Processo Organizacional, Definição do Processo Organizacional, Gerência Quantitativa do Projeto, Análise e Resolução de Causas e Inovação e Implantação na Organização.

Esses processos são organizados em sete níveis crescentes de maturidade (de G a A), sendo que, inicialmente, foram considerados apenas os processos dos quatro primeiros níveis (G a D), uma vez que esses níveis já proporcionam um elevado nível de qualidade. Assim, foi avaliada a aplicabilidade a software livre dos seguintes processos: Gerência de Projeto e Gerência de Requisitos (nível G), Garantia da Qualidade, Aquisição, Gerência de Configuração e Medição (nível F), Adaptação do Processo para Gerência de Projeto, Definição do Processo Organizacional, Avaliação e Melhoria do Processo Organizacional e Treinamento (nível E) e Desenvolvimento de Requisitos, Solução Técnica, Integração do Produto, Instalação do Produto, Liberação do Produto, Verificação e Validação (nível D).

Um desses processos foi considerado não aplicável a software livre: Instalação do Produto. Este processo foi considerado não aplicável, porque um produto de software livre, tipicamente, é disponibilizado em um *site* da *Web* para ser baixado pelos interessados, que são responsáveis pela sua instalação. Contudo, um manual de instalação a ser disponibilizado junto com o produto foi considerado importante para minimizar a ausência desse processo, bem como informações no Portal do Projeto, incluindo perguntas freqüentes e resolução de problemas.

Todos os demais processos foram considerados aplicáveis e suas atividades foram classificadas, segundo a importância de seus resultados esperados, em: obrigatória, fortemente recomendável, recomendável e desejável. Atividades obrigatórias devem ser executadas e os artefatos por elas produzidos têm de ser entregues como parte da contribuição para que a mesma seja aceita. Para facilitar sua realização, ferramentas de apoio deverão ser providas. Atividades fortemente recomendáveis, ainda que não obrigatórias, também são muito importantes e será incentivada a sua realização, provendo-se ferramentas de apoio. Para as demais atividades, inicialmente, não será provida nenhuma facilidade nem será exigida nenhuma evidência de sua realização, ainda que as mesmas estejam definidas no processo.

Algumas das atividades do processo ficarão sob responsabilidade do colaborador, ou seja do indivíduo ou organização que esteja trabalhando uma contribuição, e outras ficarão sob responsabilidade da organização que detêm o controle do projeto, no caso, o LabES. Assim, informações sobre a responsabilidade de execução do processo / atividade também foram definidas. Visando tornar o processo mais enxuto para os colaboradores, processos que são integralmente de responsabilidade do LabES não foram incorporados ao PPELabES-SL. Este é o caso dos processos de Definição do Processo Organizacional e Avaliação e Melhoria do Processo Organizacional.

No que tange ao processo de aquisição, vale destacar que, nos sub-projetos do Projeto ODE Livre, um novo projeto de desenvolvimento ou manutenção se iniciará a partir da demanda de um usuário de ODE que registrará sua crítica ou sugestão (*bug* no sistema, oportunidade de melhoria, sugestão de nova funcionalidade etc) no Portal do

Projeto. Algumas vezes, esse usuário também vai ser o colaborador responsável pelo desenvolvimento de sua própria requisição, mas acredita-se que, na maioria das vezes, outro colaborador desenvolverá um novo projeto baseado nos requisitos deixados pelo usuário idealizador. Esses fatores comprovam a falta de uma identificação clara da relação cliente/fornecedor para projetos de software livre, levando à simplificação deste processo. Apesar disso, critérios para aceitação e seleção de contribuições são fortemente recomendáveis.

Pelas características de software livre, o Processo de Treinamento, ainda que considerado desejável, não foi definido, uma vez que cabe a cada organização definir como tratar essa questão. Tutoriais sobre Engenharia de Software e sobre como realizar atividades do processo podem ser disponibilizados no Portal do Projeto. Essa porção do Processo de Treinamento, contudo, ficaria a cargo do LabES e, por isso, não foi incluída no processo. No caso do treinamento de usuários, material de apoio também deve ser produzido e disponibilizado no Portal do Projeto.

Os Processos de Gerência de Projeto e Adaptação do Processo para Gerência de Projeto foram tratados como um processo único, denominado Processo de Gerência de Projetos, composto das seguintes atividades, tomando por base o processo padrão LabES: Planejamento de Projeto, Acompanhamento de Projeto e Encerramento de Projeto. A atividade de Planejamento de Projeto é decomposta nas seguintes sub-atividades: Definição do Escopo do Projeto (obrigatória), Definição do Processo de Projeto (obrigatória), Realização de Estimativas (fortemente recomendável), Gerência de Riscos (desejável) e Elaboração do Plano de Projeto (obrigatória). No acompanhamento, as atividades do planejamento são realizadas novamente. O encerramento inclui análises *post-mortem*. Todas essas atividades são realizadas pelo colaborador, com exceção do encerramento que pode envolver também membros do LabES.

Ainda no que se refere à Gerência de Projetos, julgou-se que não se aplica a software livre atividades relacionadas a planejamento e acompanhamento de custos, tendo em vista que, por princípio, o desenvolvimento de software livre não prevê custo financeiro. Vale destacar, ainda, que a realização de estimativas é considerada apenas fortemente recomendável, e não obrigatória, pela inexistência de prazos fixos para um projeto de software livre. Cabe ao colaborador gerenciar seu tempo, ainda que estimativas de tamanho, esforço e tempo, juntamente com o cronograma do projeto, facilitem a organização e viabilidade do projeto.

No Processo de Medição, toda a parte de definição de métricas deve ficar a cargo do LabES, bem como a definição das atividades de medição e a análise dos resultados. Caberia às organizações colaboradoras apenas a coleta dos dados, bem como o uso dessas informações para apoiar suas decisões. Para tal, é muito importante prover ferramentas de apoio para que esse processo se torne viável, incluindo um sistema de gerência de conhecimento. Assim, optou-se por postergar a introdução desse processo para uma futura melhoria do PPELabES-SL. De fato, esse processo não foi incorporado ao PPLabES.

Os demais processos foram todos considerados muito importantes e incorporados ao PPELabES-SL.

Visando à simplicidade e para manter a compatibilidade com o PPLabES, os processos de Garantia da Qualidade, Verificação e Validação foram agrupados em um único processo, denominado Processo de Garantia da Qualidade. Assim, o processo de garantia da qualidade proposto envolve atividades de verificação e validação, incluindo a garantia da conformidade a padrões e revisão conjunta. Vale destacar que no âmbito da definição deste processo, diversos modelos de documento foram propostos, tais como Modelo de Plano de Projeto, Modelo de Especificação de Requisitos e Padrão de Codificação, e, portanto, verificar a aderência a esses padrões é fundamental. Assim, as atividades desses processos deverão ser realizadas tanto pelos colaboradores quanto pelos membros do LabES, durante a avaliação de uma contribuição.

O Processo de Gerência de Configuração é de extrema importância para os projetos de software livre. Os itens de configuração devem ser muito bem controlados, pois disso depende o sucesso do projeto livre. Os colaboradores devem ter acesso aos itens para poderem colaborar efetivamente e as versões de cada item devem ser controladas e documentadas de modo a evitar retrabalho e para permitir um bom entendimento por parte dos colaboradores.

Os demais processos – Gerência de Requisitos, Desenvolvimento de Requisitos, Solução Técnica, Integração do Produto e Liberação do Produto – foram todos incorporados a um processo de engenharia. Na verdade, seguindo o PPLabES, foram definidos dois processos de engenharia: um Processo de Desenvolvimento e outro de Manutenção. Esses processos têm um esqueleto muito similar, embora o primeiro esteja focado no desenvolvimento de novos artefatos, enquanto, o foco do segundo é a manutenção de artefatos previamente desenvolvidos (ainda que novos artefatos possam ser criados). Esse esqueleto é composto das seguintes atividades: Levantamento e Análise de Requisitos, Projeto, Implementação e Teste de Unidade, Testes de Integração e Validação, e Liberação do Produto. Todas essas atividades, com exceção da Liberação do Produto, são de responsabilidade do colaborador. Contudo, atividades de Integração e Testes de Integração e Validação são realizadas também por membros do LabES. Por fim, cabe aos membros do LabES decidir quando liberar uma nova versão do ambiente, devendo descrever, ainda, as modificações efetuadas e os responsáveis por elas.

É importante destacar, ainda, algumas características do Processo de Manutenção. Ele tem como objetivo principal a solução de Solicitações de Alteração feitas por usuários. Esse processo, assim como o Processo Padrão de Manutenção do LabES, é fortemente baseado na norma IEEE 1219 [IEEE, 1998]. Assim, além das atividades comuns ao processo de desenvolvimento, há uma atividade inicial de Identificação do Problema, que envolve as seguintes sub-atividades:

- Submeter Solicitação de Alteração: O processo é iniciado quando um usuário de ODE submete uma solicitação de alteração pelo Portal do Projeto. A solicitação é numerada automaticamente e armazenada;
- Analisar Validade e Classificar Solicitação de Alteração: Membros de LabES são designados para analisar a validade da Solicitação de Alteração registrada e, caso seja verificada sua real necessidade de execução, a mesma será classificada segundo tipo (corretiva, adaptativa, perfectiva ou preventiva) e área do projeto;

- Efetuar Estimativa Preliminar de Tamanho / Magnitude da Manutenção: Depois de classificada, o tamanho e a magnitude da Solicitação devem ser estimados. Essas estimativas são feitas por membros do LabES e são importantes por auxiliarem a decisão de colaboradores, indicando recursos necessários para a sua realização;
- Priorizar Solicitação de Alteração: As solicitações devem ser agrupadas por classe e áreas do projeto e submetidas a uma análise, efetuada em reunião entre membros do LabES, na qual são priorizadas e posteriormente publicadas como um novo sub-projeto no Portal do Projeto, estando disponível para os colaboradores. As reuniões são periódicas.

A partir deste ponto o colaborador é o agente ativo na solução de Pacotes de Solicitações de Alteração. Através do Portal do Projeto, o colaborador se disponibiliza a solucionar um pacote de solicitações e obtém os artefatos necessários para tal. Uma vez selecionado um sub-projeto de manutenção por um colaborador, um processo semelhante ao processo de desenvolvimento é realizado, incluindo as interações com os processos de apoio.

A Figura 2 mostra esquematicamente a estrutura do Processo Padrão LabES para Software Livre. Finalmente, vale destacar que durante a elaboração desse processo, buscando aderência ao processo padrão desta organização, os processos foram agrupados em duas categorias: processos fundamentais, incluindo os processos de desenvolvimento e manutenção, e processos de apoio, englobando os processos de gerência de projetos, garantia da qualidade e gerência de configuração, como mostra a Figura 2.

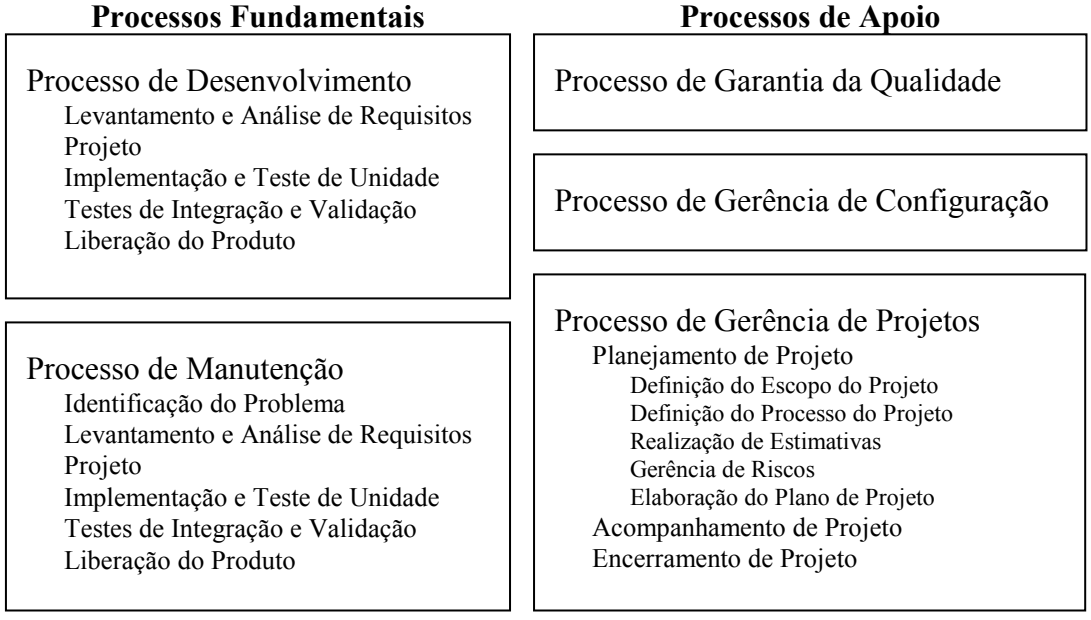

Figura 2 – Estrutura Básica do Processo Padrão LabES para Software Livre.

Esses processos são definidos em termos de atividades e sub-atividades e, para cada atividade, são definidos, ainda: nome, descrição, critérios de entrada e saída, responsáveis, participantes, artefatos requeridos e produzidos, pré e pós-atividades,

ferramentas, procedimentos, resultados esperados (em termos dos resultados esperados definidos no MPS.BR), observações e um indicador da necessidade de execução (obrigatório, fortemente recomendado, recomendado, desejável).

5. Trabalhos Correlatos

Ainda que não tenhamos encontrado na literatura trabalhos descrevendo iniciativas de definição de processos de software livre com base em modelos de qualidade de processo, como fizemos neste trabalho, é possível apontar diversos trabalhos correlatos, alguns já comentados. Vale destacar que muitos deles serviram de inspiração para o processo aqui definido, como é o caso de [Reis, 2003]. Nesse trabalho, Reis identificou características de processos de desenvolvimento de software livre e definiu um ciclo de vida para projetos desta natureza. O ciclo de vida proposto por Reis é composto de quatro grandes fases: Criação do Projeto, Lançamento Público, Crescimento e Organização, e Maturidade. De fato, sua proposta não está focada no processo de software diretamente, mas no ciclo de vida de um projeto típico de software livre. Neste sentido, estamos seguindo esse modelo, estando no presente momento em uma fase de pré-lançamento do Projeto ODE Livre.

O trabalho de Johnson (2001) visa a prover um modelo de processo descritivo para o desenvolvimento de software livre. Neste contexto entende-se por modelo descritivo um modelo que captura o que é realmente feito, em contrapartida a um modelo prescritivo, que caracteriza o que se supõe que deve ser feito. O resultado principal, como em [Reis, 2003], é a identificação de características encontradas em diversos projetos de software livre, dentre elas: uma versão inicial deve ser produzida como um protótipo fechado e evoluída iterativamente; o desenvolvimento é concorrente, colaborativo e descentralizado; revisão por pares é largamente empregada; requisitos são centrados no usuário, mas não formalmente definidos; a liderança é exercida com base na confiança e competência; a motivação é interna, incluindo trabalho comunitário e status; a comunicação é assíncrona; o planejamento é informal; há participantes de diferentes níveis. Em contrapartida, o processo apresentado neste trabalho é prescritivo, como o são os processos padrão definidos com base em modelos e normas de qualidade. Contudo, pode-se notar que muitas das considerações feitas por Johnson são consideradas no processo aqui proposto. Outras, porém, são questionadas, como requisitos não formalmente definidos. Neste aspecto estamos mais de acordo com Nakagawa (2004), que, através de dois experimentos, aponta a importância de se ter uma especificação de requisitos em projetos de software livre.

Outros trabalhos correlatos incluem [Mockus et al., 2002] e [Reis and Fortes, 2002], o primeiro discutindo características dos processos de desenvolvimento dos Projetos do Servidor de HTTP Apache e do Navegador Mozilla; o segundo dando uma visão geral do processo de software e das ferramentas usadas no Projeto Mozilla. Ambos foram úteis para apoiar a tomada de decisão durante a definição do processo padrão apresentado neste trabalho.

6. Conclusões e Perspectivas Futuras

O movimento de Software Livre vem ganhando importância no Brasil e no mundo, não só por questões de custos, mas principalmente por introduzir uma nova filosofia na

engenharia de software, centrada na cooperação. Porém, a qualidade dos produtos de software livres ainda está muito dependente de quem os desenvolve, sem seguir um processo de qualidade.

Visando avançar em direção a processos de software de qualidade para o desenvolvimento e manutenção de software livre, no contexto do Laboratório de Engenharia de Software (LabES) da UFES, definiu-se um processo padrão para software livre, tomando por base as principais características típicas de projetos de software livre relatadas na literatura, normas e modelos de qualidade de processo, sobretudo MPS.BR, ISO/IEC 12207 e IEEE 1219, e características específicas da organização e do Projeto ODE Livre, que visa ao desenvolvimento do ambiente de desenvolvimento de software ODE segundo o modelo de software livre.

ODE já apresenta funcionalidade suficiente para atrair a comunidade de Engenharia de Software e, assim, este ambiente já pode ser lançado como software livre. Contudo, tomando por base outros trabalhos, nota-se que é fundamental para o sucesso de um projeto que o mesmo possua uma infra-estrutura de suporte, incluindo um portal do projeto e diversas ferramentas de apoio ao processo definido, com destaque para ferramentas de apoio à gerência de configuração e de apoio ao trabalho cooperativo (ferramentas de *groupware*). O Portal do Projeto ODE Livre já está em desenvolvimento e inclui algumas facilidades básicas de trabalho cooperativo.

Dada a complexidade do processo definido, para que os colaboradores sintam-se mais estimulados a se engajar ao projeto, está-se estudando a possibilidade de prover outras ferramentas de apoio, menos comuns em projetos de software livre. Na verdade, o estudo está direcionado para o uso do próprio ambiente ODE como ferramenta de suporte aos colaboradores do Projeto ODE Livre.

Por fim, antes do lançamento de ODE como um produto livre, está prevista a realização de projetos piloto controlados usando o processo e a infra-estrutura de suporte definidos. Esses projetos serão desenvolvidos em outras instituições de ensino, sem contato direto entre os desenvolvedores colaboradores e os membros do LabES, simulando uma situação real de um projeto de software livre.

Agradecimentos

Este trabalho foi realizado com o apoio do CNPq, entidade do Governo Brasileiro dedicada ao desenvolvimento científico e tecnológico, da FAPES, Fundação de Apoio à Ciência e Tecnologia do Espírito Santo, e da VixTeam Consultoria e Sistemas, empresa parceira que têm financiado o projeto e dado feedback de sua aplicação a casos reais.

Referências

Davis, M., O'Donovan, W., Fritz, J. (2000) Linux and open source in the academic enterprise. Proc. of the 28th SIGUCCS Conference on User Services, Richmond.

Falbo, R. A., Natali, A. C. C., Mian, P.G., Bertollo, G., Ruy, F.B. (2003) "ODE: Ontology-based software Development Environment", In: Memórias de IX Congreso Argentino de Ciencias de la Computación, p. 1124-1135, La Plata, Argentina.

Free Software Foundation (2005) "O que é software livre?", on-line, disponível em http://www.gnu.org/philosophy/free-sw.pt.html.

Fuggetta, A. (2000) “Software Process: A Roadmap”, Proceedings of the Conference on the Future of Software Engineering, ICSE´2000, Limerick, p. 25-34.

Harrison, W., Ossher, H., Tarr, P. (2000), “Software Engineering Tools and Environments: A Roadmap”, Proceedings of the Conference on the Future of Software Engineering, ICSE´2000, Limerick, p. 261-377.

Hexsel, R.A. (2002) “Propostas de Ações de Governo para Incentivar o Uso de Software Livre”. Relatório Técnico RT-DINF 004/2002, Universidade Federal do Paraná.

IEEE 1219 (1998) IEEE Standard for Software Maintenance.

ISO/IEC 12207 (1995), Amd 1 (2002), Amd 2 (2004) Information Technology - Software life cycle processes.

Johnson, K. (2001) A Descriptive Process Model for Open-Source Software Development, Master Thesis, University of Calgary, Canada.

Nakagawa, E. Y. (2002) “Software Livre: Processo e Produto Livres no Desenvolvimento de Aplicações Web”, Exame de Qualificação ao Doutorado, Instituto de Ciências Matemáticas e de Computação – ICMC/USP, São Carlos.

Nakagawa, E. Y. (2004) “An Investigation of the Open Source Development Process”, Actas de las IV Jornadas Iberoamericanas de Ingeniería del Software e Ingeniería del Conocimiento, JIISIC'2004, Madrid, España.

Mockus, A., Fielding, R. T., Herbsleb, J. (2002) “Two case studies of open source software development: Apache and Mozilla”. *ACM Transactions on Software Engineering and Methodology*, v. 11, n. 3.

Raymound, E. (1999) *The cathedral and the bazaa*r. O’Reilly & Associates.

Reis, C.R. (2003) Caracterização de um Processo de Software para Projetos de Software Livre. Dissertação de Mestrado, ICMC/USP, São Carlos.

Reis, C., Fortes, R.P.M. (2002) “An Overview of the Software Process and Tools in the Mozilla Project”. Proceedings of the Open Source Software Development Workshop. Newcastle, p. 155–175.

Rocha, A.R.C., Maldonado, J.C., Weber, K.C. (2001) *Qualidade de Software: Teoria e Prática*, Prentice Hall.

Scacchi, W. (2002) Understanding the requirements for developing open source software systems. In: *IEE Proceedings – Software Engineerin*g, 149(1), p. 24-39.

Softex (2005a) “O impacto do software livre e de código aberto na indústria de software do Brasil”, Softex Campinas, disponível em www.softex.br.

Softex (2005b) MPS.BR – Melhoria de Processo do Software Brasileiro: Guia Geral, Versão 1.0, disponível em www.softex.br/mpsbr .

Vixie, P. (1999) Software Engineering, In: *Open sources: Voices of the open source revolution*, 1st edition, O’Reilly & Associates, p. 91–100.