Lições Aprendidas na criação de uma Fábrica de Software *Open-Source*

Alexandre Alvaro, Thiago Luiz Vieira de Lima Santos, Paulo Rogério P. Andrade[2], João Marcos P. Vasconcelos, Jones Albuquerque, Silvio Romero de Lemos Meira

Centro de Informática – Universidade Federal de Pernambuco (UFPE)
Caixa Postal 7851 – 50732-970 – Recife – PE – Brasil

{aa2, tlvls, jmpv, joa, srlm}@cin.ufpe.br, [2]prpa0807@hotmail.com

Resumo. *Pesquisa e esforços envolvendo fábricas de software tem sido apresentados ao longo dos anos na literatura. Entretanto ainda existe uma carência de relatos da criação de fábricas de softwares open-source para o desenvolvimento distribuído. Neste artigo serão apresentadas, brevemente, as etapas de criação de uma fábrica de software open-source e as lições aprendidas durante este processo.*

1. Introdução

Ao longo da última década, o desenvolvimento de software *open-source* tem atraído cada vez mais atenção. A rápida evolução de projetos de software de código aberto (OSS) como o sistema operacional Linux, *browser* Mozilla, IDE Eclipse, entre tantos outros, faz com que empresas tradicionais procurem meios de aproveitar o potencial do desenvolvimento de código aberto em suas atividades de desenvolvimento de software.

A criação de fábricas de software, principalmente aquelas com times geograficamente dispersos (Fábrica de Software Distribuída), tem-se mostrados um processo um tanto quanto árduo para a área de Engenharia de Software. Se considerarmos uma fábrica de software o*pen-source* com o desenvolvimento distribuído, as dificuldades aumentam significativamente.

Com o objetivo de garimpar esta área, dois professores da Universidade Federal de Pernambuco (UFPE) tiveram a idéia de criar uma disciplina de pós-graduação, onde grupos de alunos pudessem se unir e criar fábricas de software *open-source*, com objetivos de: aplicar conceitos, técnicas e ferramentas de Engenharia de Software tradicional; analisar as intersecções e disjunções entre engenharia de software e o desenvolvimento de software o*pens-source*; e definir, montar (projeto piloto) e avaliar (aplicação real) uma Fábrica de Software *Open-Source*, gerando relatórios sobre o experimento que possibilitem um melhor estudo sobre esse ambiente tão complexo.

Assim, focando nos objetivos da disciplina, um grupo de oito estudantes de Mestrado ou Doutorado, se propuseram a estender estes objetivos no sentido de prover artefatos do processo de desenvolvimento de software (processo, modelos, procedimentos, *guidelines*) para que a comunidade de software o*pen-source* pudesse ser beneficiada com as experiências e conhecimentos adquiridos. Desta forma surgiu a Fábrica de Software *Open-Source* USINA (**U**nited **S**oftware **IN**fra-structure **A**lliance).

O artigo está organizado da seguinte forma: a seção 2 apresenta o processo de criação da fábrica de software *open-source*, ressaltando as lições aprendidas durante este processo; a seção 3 apresenta o estudo de caso que está sendo desenvolvido pela fábrica; e na seção 4 são apresentadas as conclusões e perspectivas de trabalhos futuros.

2. Visão Geral do Processo de Criação da Fábrica

Em função do conhecimento dos membros da equipe em projetos reais em empresas de desenvolvimento de software, constatou-se que a primeira prioridade era disponibilizar uma boa infra-estrutura (como sistemas de controle de versão, listas de discussão, ferramenta para atribuição de tarefas, entre outras). Tais ferramentas possibilitam que os membros da USINA, geograficamente distribuídos, trabalhassem colaborativamente e de forma organizada. Para atender a essa necessidade, estão sendo utilizados dois *sites* que provém suporte para tal necessidade, o *tigris*[1] e o *sourceforge*[2]. No primeiro está hospedado o *site* da fábrica USINA[3], contendo o processo e os artefatos a serem gerados durante o desenvolvimento. No segundo foi criada uma instância do processo da fábrica USINA para o desenvolvimento de um estudo de caso específico, no caso de simulação de um aquário, chamado Simulare[4], o qual será melhor descrito na seção 3.

Após esta primeira etapa, a equipe da fábrica se concentrou no desenvolvimento do processo da USINA, identificando os artefatos a serem gerados pelo processo e construindo modelos e procedimentos de uso para tais documentos. Nesta fase, diversos processos e metodologias de desenvolvimento presentes na literatura serviram como base, como o XP [1], o RUP [2], entre outros [3,4,5]; assim como diversos relatos de experiências no desenvolvimento de software *Open-Source* [6,7,8,9,10]. Visto isso, foi definido um macro-processo, o qual pode ser vista na Figura 1. Nele observamos cinco módulos distintos responsáveis por cada uma das fases de desenvolvimento do software desde a sua fase de pré-venda, até a pós-venda.

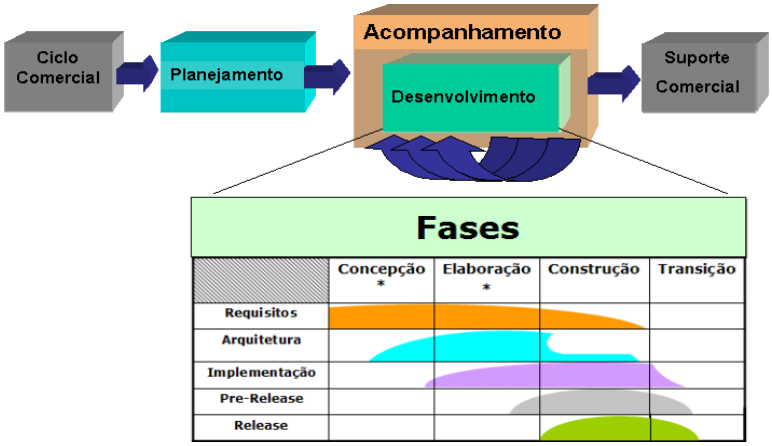

Figura 1. Processo de Desenvolvimento

As etapas do processo são:

1. **Ciclo comercial**: O ciclo comercial define o processo de pré-venda do software. Esta etapa envolve atividades como: identificação de oportunidades de negócio; apresentação da fábrica para possíveis clientes (necessária pela abordagem singular que é a de desenvolvimento de OSS); recebimento e resposta a requisições de propostas (RFP); negociação de contratos incluindo definição do nível de serviços prestado (SLA);

---

[1] http://tigris.org

[2] http://sourceforge.net

[3] http://usina.tigris.org

[4] http://sourceforge.net/projects/simulare

2. **Planejamento**: Esta etapa define a infra-estrutura necessária à realização de projetos de acordo com o que foi definido na RFP e na SLA. Suas atividades podem ser antecipadas de forma a tornar as respostas a RFPs e SLA mais completas;
3. **Acompanhamento**: Esta etapa define todo o procedimento de monitoramento da etapa de desenvolvimento. Todo desenvolvimento deve ser acompanhado de perto para permitir a re-priorização de requisitos, gerenciamento de riscos, resolução de conflitos, e fornecimento de relatórios ao cliente indicando o *status* do desenvolvimento;
4. **Desenvolvimento**: Esta etapa é dividida em diversas fases. O esforço necessário em cada fase é variável de acordo com o estágio de evolução do projeto. Em cada iteração da etapa de desenvolvimento, atividades de todas fases são executadas; e
5. **Suporte comercial**: Após a geração de cada *release* do sistema a fábrica pode fornece uma série de serviços associados, como: a instalação do software, treinamentos para a utilização da ferramenta, e a confecção de manuais e tutoriais do software.

Estas etapas abrangem, além das atividades de desenvolvimento de software, todo o processo de pré e pós venda, não contemplado por metodologias consolidadas como o RUP [2]. Para cada uma das etapas são disponibilizados modelos dos documentos que devem ser confeccionados antes que a etapa seguinte tenha início.

Maiores detalhes do processo de desenvolvimento da fábrica de software *open-source* USINA, pode ser encontrado no documento de Processo da USINA em seu *site*.

2.1. Lições Aprendidas

Através do processo de criação da fábrica e do processo de desenvolvimento de software *open-source*, algumas lições puderam ser observadas, como seguem:

- **Diferenças de conhecimento:** No começo da criação da fábrica, é muito importante que alguma(s) pessoa(s) tenham iniciativa, pois muitas idéias boas surgem, entretanto nada é colocado em prática;
- **Infra-Estrutura:** Antes de iniciar a criação de uma fábrica de software *open-source*, a primeira prioridade é a disponibilização de uma boa infra-estrutura para não impactar nas atividades da fábrica futuramente;
- **Processo Leve:** O processo de desenvolvimento de uma fábrica de software distribuída o*pen-source* deve ser o mais leve possível. Processos pesados simplesmente seriam ignorados por colaboradores externos ou pior, desencorajaria a participação destes. O processo deve contemplar o mínimo que: garanta o entendimento do funcionamento da fábrica por parte do cliente e facilite a colaboração entre os envolvidos no desenvolvimento do software;
- ***Releases* Freqüentes:** A geração de *releases* freqüentes afeta positivamente a motivação dos colaboradores, que podem ver o resultado de seu esforço em pouco tempo. Estas versões intermediárias também ajudam a refinar os requisitos junto ao cliente;
- **Processo Iterativo e Incremental:** De uma forma geral, a maioria dos OSS que obtiveram sucesso são desenvolvidos de forma iterativa e incremental. Assim, o processo de desenvolvimento deve contemplar isso;
- **Integração Contínua:** As vantagens da integração contínua [1,11] são ressaltadas em projetos *open-source*, contudo, é necessário cuidado durante sua implementação. Um controle sobre quem pode colaborar diretamente é indispensável;
- **Disseminação de responsabilidades entre os colaboradores, conceito de *core groups***: Mais de um colaborador divide a mesma responsabilidade de um papel dentro da fábrica. Isso evita "gargalos" no processo quando se da a ausência de um colaborador e, facilita a distribuição de atividades colaborativa;
- **Distribuição de atividades colaborativa**: Com o conceito de *core-groups*, a delegação de atividades se torna mais ágil. Uma vez finalizada uma etapa, seus responsáveis (*core-groups*) delegam tarefas para colaboradores responsáveis das etapas seguintes; e

- **Dispositivos de comunicação assíncronos:** A incompatibilidade existente entre os horários disponíveis dos membros da USINA reflete bem a realidade de um desenvolvimento globalmente distribuído, o que é bastante comum para projetos *open-source*. Meios de comunicação assíncronos, como e-mail e listas de discussão, são favorecidos por possibilitarem que cada um colabore quando tiver disponibilidade.

3. Estudo de Caso

Para validar o processo de desenvolvimento da fábrica USINA, foi requisitado pelo cliente (professores da disciplina) a realização do projeto e implementação de um Simulador de Aquário. Dentro das diversas tipologias de sistemas ambientais, o aquário pretendido classifica-se como sistema isolado, controlado, complexo e organizado. Possui como fundamento algorítmicos, os modelos matemáticos probabilísticos de otimização. Com isso, pretende-se obter um modelo de conteúdo plenamente especificado e estruturado estocástico.

Este sistema esta sendo desenvolvido com o intuito de validar o processo proposto de uma fábrica de software *open-source*, cujos *releases* iniciais encontram-se no *site* do projeto Simulare. Após isto, haverá a re-calibração da fábrica de software e o projeto piloto será estendido, podendo haver mudanças de requisitos por parte do cliente, no sentido de validar o processo em um caso real dentro de uma comunidade o*pen-source*.

Maiores detalhes do andamento do projeto podem ser vistos no *site* do projeto Simulare.

4. Conclusões e Trabalhos Futuros

Este trabalho apresentou algumas experiências relevantes para a criação de uma fábrica de software *open-source*. As lições relatadas auxiliam e servem como ponto de partida para que novas comunidades de software *open-source* possam surgir mais estruturadas e organizadas.

Como trabalhos futuros, está o amadurecimento do processo de software *open-source* proposto, através da extensão do estudo de caso analisado e da interação com a comunidade *open-source*. Ainda, ao final da disciplina, todo o conhecimento adquirido e mais o estado atual do projeto estarão abertos para a comunidade *open-source* para novas contribuições.

5. Referências

[1] K. Beck, *Extreme Programming Explained: Embrace Change*, Addison-Wesley, 1st Edição, 1999.

[2] Rational Software Corporation, *RUP - Rational Unified Process*, Disponível em: http://www-306.ibm.com/software/awdtools/rup. Consultado em Maio 2004.

[3] L. A. Norin, *Open-Source software development methodology*. Dissertação de Mestrado em Desenvolvimento de Sistemas e Engenharia de Software. Luleå University of Technology, 1999.

[4] P. Abrahamsson, et. al., *Agile software development methods*. Relatório Técnico, Finlândia, 2002.

[5] C. Larman, *Agile and Iterative Development: A Manager's Guide*. Addison-Wesley. 1st Edição, 2003.

[6] D. Hart, *Faster, Better, Cheaper: Open-Source Practices May Help Improve Software Engineering*. Disponível em: http://www.nsf.gov/od/lpa/news/03/pr03132.htm. Consultado em Maio 2004.

[7] OSAPortal, *Free Software Development Process*. Disponível em: http://www.opensourcearmenia.com/education/business/Free_Software_Development_Process.html. Consultado em Maio 2004.

[8] B. Fitzgerald, D. L. Parnas, *Making Free/Open-Source Software (F/OSS) Work Better*. Proceedings do Workshop da Conferência XP2003, Genova, 2003.

[9] W. Scacchi, *Understanding the Requirements for Developing Open Source Software Systems*, IEEE Proceedings, 2002. pp. 25-39.

[10] *The Cathedral and the Bazaar*. Disponível em: http://www.catb.org/~esr/writings/cathedral-bazaar/cathedral-bazaar. Consultado em Abril de 2004.

[11] M. Fowler, M. Foemmel, *Continuous Integration*. Disponível em: http://www.martinfowler.com/articles/continuosIntegration.html. Consultado em Maio de 2004.