Edeyson

Adapting the "staged model for software evolution" to free/libre/open source software.

Resumo

Pesquisas em evolução de software têm abordado duas perspectivas:

1. Foco em processos, métodos e técnicas para implementar e evoluir software (como evoluir) e a inovação plausível neste cenário.
2. Aquisição e compreensão de suas causas e características (por que evoluir), objetivando inferir as atividades de engenharia de software relacionadas através de observação sistemática de dados empíricos.

As duas perspectivas ligam-se, pois o estudo da causa da evolução direciona o desenvolvimento de alternativas que determinam como evoluir.

O artigo em questão aborda o modelo de estágios de evolução de software – um conjunto de passos que representa o ciclo de vida de um software - e como o mesmo pode ser empregado (apresentando suas devidas alterações) a projetos de software de código livre e aberto (FLOSS). Como estes têm suas peculiaridades em relação a projetos de código proprietário, elas são apresentadas e comparadas.

A principal peculiaridade de projeto FLOSS é ter código fonte aberto, disponível em repositório público (geralmente para leitura). Com isso, apresenta a disponibilidade contínua ao código o que o diferencia no primeiro estágio, desenvolvimento inicial, por apresentar release contínua.

Em contrapartida, os softwares proprietários normalmente disponibilizam uma release quando possui um núcleo de funcionalidades realmente usável.

Outra peculiaridade de projeto FLOSS é que seu grupo de desenvolvimento não é fixo, variando em quantidade de integrantes em função do tempo e do interesse em desenvolvimento de novas funcionalidades.

A evolução de projetos FLOSS apresenta com maior freqüência novas releases e correções do sistema, fator inferido pelos desenvolvedores como vitalidade do projeto, embora exista uma vertente de planejamento de releases.

A fase de manutenção de projetos FLOSS (servicing) é afetada por sua característica de constante evolução. O fato de existirem releases constante dificulta a identificação concreta desta fase. Destaca-se, então, que projetos FLOSS podem sair da fase de manutenção para a de evolução, embora isso necessite ser identificado entre versões distintas.

Outra peculiaridade de projetos FLOSS é que, mesmo em phase out, a disponibilidade do código fonte possibilita que novos interessados assumam o projeto, retornando-o ao estágio de evolução.

Crítica

O artigo é sucinto e preciso no refinamento do modelo de estágios para FLOSS, embora não aborde a evolução do modelo de estágios com versões, apresentando suas diferenças principais para sistemas proprietários.

Como principal contribuição, destaca-se a expansão do modelo de estágios para refletir as transições de servicing e phase out para evolution, bem como das justificativas para que elas ocorram.

Devido ao crescente interesse e uso de software livre, apresenta-se bastante relevante ao tema de evolução de software.

Questões para discussão

1. Como identificar, automaticamente em repositórios de projetos FLOSS, que o mesmo passou de evolution para servicing? Isso, dado que existe release contínua?
2. Como seria a adaptação do modelo de estágios versionados, proposto por Keith Bennet e Vaclav Rajlich, para projetos FLOSS?
3. Esse modelo com versões seria aplicável a quais tipos de projetos FLOSS?

Referência

Capiluppi, A., González-Barahona, J. M., Herraiz, I., and Robles, G. 2007. Adapting the "staged model for software evolution" to free/libre/open source software. In Ninth international Workshop on Principles of Software Evolution: in Conjunction with the 6th ESEC/FSE Joint Meeting (Dubrovnik, Croatia, September 03 - 04, 2007). IWPSE '07. ACM, New York, NY, 79-82.