colloquium exactarum

PROCESSOS, MÉTODOS E PRÁTICAS DE ENGENHARIA DE SOFTWARE EM PROJETOS SOFTWARE LIVRE: UM ESTUDO DE CASO OWNCLOUD E NEXTCLOUD

Processes, methods and practices of Software Engineering in open source projects: A case study of OwnCloud and Nextcloud

Juliano Rodrigues Ramos [1]; Tamiris Fernanda Malacrida [2]

[1]Universidade do Oeste Paulista – Unoeste
Graduado em Sistemas de Informação pela Faculdade de Informática de Presidente Prudente – FIPP
Presidente Prudente – São Paulo, Brasil.
juliano_rr@yahoo.com.br

[2]Universidade do Oeste Paulista – Unoeste
Graduada em Ciência da Computação pela Faculdade de Informática de Presidente Prudente – FIPP
Presidente Prudente – São Paulo, Brasil.
tamirismalacrida@gmail.com

RESUMO – As práticas da Engenharia de *Software* (ES) têm por fundamento tentar melhorar a qualidade dos *softwares*, nos quais se incluem os *Open Source Software (OSS)*. O objetivo deste trabalho é identificar as práticas da ES implementadas nos projetos *OSS* OwnCloud e Nextcloud. Para tanto, realizou-se uma revisão da literatura científica, a partir de diferentes bases de dados digitais, bem como foi considerada a literatura cinza. Foi concluído que os projetos de armazenamento em nuvem OwnCloud e Nextcloud adotam boas práticas de engenharia de *software*. Porém, foi constatado que, o maior foco é ainda na codificação, e que, portanto, os demais processos do desenvolvimento, como as documentações, controle efetivo do gerenciamento das mudanças e processos de gestão da qualidade, devem receber mais ênfase.
Palavras-chave: Software Open Source; OwnCloud; Nextcloud; Engenharia de Software.

ABSTRACT – Software Engineering practices aim to improve the softwares quality, which includes Open Source Software (OSS). This work aims to identify software engineering practices in the OSS projects OwnCloud e Nextcloud. We perform a scientific literature review, using digital databases such as the Google Scholar and CAPES Periodic Portal, and also search using the gray literature. We conclude that good practices of software engineering are applied into both cloud storage projects OwnCloud and Nextcloud. However, we found that the major focus is still on coding, and therefore, other development processes, such as documentation, effective control of change management and quality management processes, should receive more emphasis.
***Keywords*:** Open Source Software; OwnCloud; Nextcloud; Software Engineering.

Recebido em: 02/06/2018
Revisado em: 04/09/2018
Aprovado em: 17/09/2018

1. INTRODUÇÃO

O desenvolvimento de *software* tem crescido nos últimos anos devido a sua enorme necessidade na sociedade contemporânea, e um aspecto de destaque nesse contexto é o volume de *Open Source Software* (*OSS*). A Engenharia de *Software* (ES) surge no sentido de melhorar a qualidade dos *softwares* em geral e aumentar a produtividade no desenvolvimento de tais produtos (ALGARRÃO, 2018).

Diversas plataformas *OSS,* para computação em nuvem, foram lançadas nos últimos tempos e dentre algumas das mais populares, tem-se: *Eucalyptus*, *OpenStack*, *Nimbus*, *OpenNebula*, OwnCloud e Nextcloud (THOMÉ; HENTGES; GRIEBLER, 2013). A computação em nuvem é a convergência e a evolução de vários conceitos da virtualização, projeto de aplicativos distribuídos, da grade e do gerenciamento de Tecnologia da Informação (TI) corporativo, para permitir uma abordagem mais flexível à implantação e ao dimensionamento de aplicativos (THOMÉ; HENTGES; GRIEBLER, 2013).

Em 2010, Frank Karlitschek propõe a primeira versão do OwnCloud, sendo sua versão beta disponibilizada em março do mesmo ano (FELDMAN, 2018). Dois anos depois, surge o OwnCloud In*c,* encerrado em abril de 2016, quando Frank Karlitschek propõe o Nextcloud (FELDMAN, 2018). Assim como o OwnCloud, o Nextcloud é um serviço *OSS* de armazenamento e sincronização de arquivos privados, similar ao *Dropbox* (proprietário), porém derivado (*fork*) do OwnCloud. O objetivo do *Nextcloud* é trazer diversas funcionalidades e melhorias ao projeto inicial, o OwnCloud*,* o qual sofreu algumas divergências na comunidade para se manter no mercado (OwnCloud e Nextcloud: Comparando Serviços de Armazenamento em Nuvem).

A partir do desafio fundamental da Engenharia de *Software* que se resume a 'como construir *softwares* melhores?', este trabalho tem por objetivo identificar as práticas da ES implementadas nos projetos *OSS* OwnCloud e Nextcloud.

2. MÉTODO

O método empregado para atingir o objetivo proposto deste trabalho consistiu da revisão da literatura, a partir de diferentes bases de dados digitais como *Google Scholar* e Portal de Periódicos CAPES. Além disso, considerou-se a denominada literatura cinza, a qual inclui a pesquisa em *blogs* e *sites*, isto devido ao fato tanto da dificuldade de se encontrar trabalhos científicos relacionados ao tema em estudo, quanto por observar ser recorrente o uso da literatura cinza no desenvolvimento de trabalhos científicos na área da computação como, por exemplo, realizado por Afzal et al. (2016) e Garousi, Ferderer e Hacaloglu (2017; 2018).

3. RESULTADOS E DISCUSSÃO

Os resultados e as respectivas discussões consistem das práticas da engenharia de *software* aplicadas nos projetos OwnCloud e Nextcloud*,* e são apresentados nas subseções que seguem.

3.1. Gerenciamento de Projetos

O planejamento de projeto é a documentação que reúne e organiza todos os processos da fase de planejamento e, normalmente, é elaborado por uma equipe que trabalha somente para definições de projetos. Nesse planejamento, define-se como o projeto será executado, monitorado, controlado e encerrado, planejando todas as ações necessárias para que os objetivos propostos sejam alcançados (ALGARRÃO, 2018).

Em se tratando do *fork* do OwnCloud para o Nextcloud*,* não houve um planejamento de projeto adequado de como seria o processo de migração, se comparado aos itens da Engenharia de *Software*. Isto porque foi uma surpresa para todos os usuários quando o fundador da OwnCloud, Frank Karlitschek, uma comunidade com mais

de 10 milhões de membros, anuncia sua saída para formar uma nova empresa (OwnCloud e Nextcloud: Comparando Serviços de Armazenamento em Nuvem). Com isso, o OwnCloud teve como planejamento continuar fornecendo *software* e suporte, com sua equipe original, para não sofrer decadência e correr o risco de ir à falência (BHARTIYA, 2016). Nesse contexto, o Nextcloud ganhou forças para se construir devido grande parte de sua equipe ser oriunda do OwnCloud, a qual possuía vasta experiência na área para a construção de um novo projeto (BHARTIYA, 2016).

3.2. Engenharia de Requisitos

A gestão de requisitos em projetos de código aberto, geralmente, é complexa de se fazer uma vez que os requisitos de usuários estão espalhados em um volume extenso e em uma variedade de fontes, como fóruns de discussão, *blogs* e etc, além de apresentarem uma alta porcentagem de duplicações em vários projetos de alto nível (HECK; ZAIDMAN, 2013).

Boa parte dos projetos de *software*, *OSS* ou projetos industriais, utilizam algum tipo de metodologia de desenvolvimento ágil de *software*, onde a forma mais comum de representação dos requisitos é no formato de *User Stories* (*US*). Apesar de sua popularidade na indústria de *software*, quase não se encontram pesquisas de avaliação quanto à sua qualidade. Os *post-its* utilizados são, muitas vezes, fonte de textos informais que podem conter duplicidades ou informações incoerentes (LUCASSEN et al., 2015).

Quanto aos projetos OwnCloud e Nextcloud*,* estes permitem o apontamento de *bugs* e novas funcionalidades em formato de *issues* na própria ferramenta que gerencia o repositório de cada um dos seus projetos, o *GitHub (*OwnCloud *Core – Issues)* e (Nextcloud *Core – Issues*). Além disso, tais projetos possuem um portal central de atendimento para a recepção de situações apontados pelos usuários que desejam reportar problemas ou colaborar com os projetos de alguma forma. Esses portais de inclusão de *US* permitem a localização das histórias por categoria e popularidade, o que auxilia na localização de uma situação já existente, evitando duplicidades (Central de suporte OwnCloud) e (Central de suporte Nextcloud).

3.3. Gerência de configuração de software

A Gerência de Configuração de *Software* (GCS) se resume no conjunto de atividades que permite a absorção ordenada das mudanças inerentes ao desenvolvimento de *software*: controlar e acompanhar mudanças (Controle de Mudança), registrar a evolução do projeto (Controle de Versão) e estabelecer a integridade do sistema (Integração Contínua) (DIAS, 2016).

O controle de mudanças na maioria dos projetos de *software* livre, utiliza a própria aplicação onde está o repositório para o reporte de *bugs* e solicitação de mudanças. Isto não é muito diferente para as aplicações OwnCloud e Nextcloud. Ambos, a partir do seu MD (Manual do Desenvolvedor), oferecem acesso ao nomeado *Bugtracker*, um portal onde é possível encontrar as diretrizes para o envio de problemas (Nextcloud *Developer's Manual*) e (*Support Channels* - OwnCloud). Trata-se de um FAQ (*Frequently Asked Questions*), no qual os *reports* podem ser classificados com as categorias *label* e *tags*. Além disso, *bugs* e novas funcionalidades, também, são relatadas em formato de *issues* no *GitHub*, no qual pedem que sejam realizadas no repositório correspondente, o repositório do projeto cliente ou servidor.

Quanto ao controle de versão, esta é a parte principal da GCS, pois gerencia o registro histórico de cada incremento (configuração) de implementação de solicitação de mudança. Além do registro das configurações, o controle de versão possibilita a edição concorrente sobre os arquivos e a criação de variações dos projetos (FELDMAN, 2018).

Em projetos *OSS*, a implementação de código, em sua maioria em formato de requisições de pacotes de código (*pull-*

requests), é a parte priorizada pelas comunidades *open source* (STEINMACHER et al., 2018). As submissões voluntárias de correção de erros e propostas de novos recursos são de extrema importância para a continuidade dos projetos do tipo *OSS*.

Os projetos OwnCloud e Nextcloud possuem seu controle de versão de código em repositórios do *GitHub*. O passo a passo das submissões está bem documentado nos manuais do desenvolvedor de cada uma dessas aplicações (CHEN; JIANG, 2017) e (*Developer's Manual* OwnCloud). Ambos os manuais são bem semelhantes e bem completos, contendo informações desde o *download* do código fonte, requisitos de *plugins* e aplicações a serem instaladas, até orientações sobre como deve ser feita a submissão da contribuição, por exemplo, referência quanto à estrutura do código e obrigatoriedade de testes. Além disso, está bem clara a orientação para que sejam, primeiramente, consultadas as listas de *issues* existentes em cada respectivo repositório para evitar submissões não aceitas como em (STEINMACHER et al., 2018).

Em relação à Integração Contínua (CI), esta consiste em construir o sistema a partir dos itens registrados em uma configuração, e em sua maioria é feita através de *scripts* que automatizam a construção, testes e coleta de métricas de qualidade (FELDMAN, 2018). As ferramentas de integração contínua acompanham o controle de versão e disparam os *scripts* cada vez que uma nova configuração é registrada (FELDMAN, 2018).

Inicialmente, a OwnCloud utilizou as ferramentas de integração contínua *OSS Travis* e *Jenkins*. No entanto, utilizando apenas a versão gratuita do *Travis,* e definindo não ser possível o investimento pelo alto custo da versão *premium*, a empresa se deparou, ainda, com problemas de complexidade, utilização na produção e demora na execução dos testes (JACKSON, 2018). Na utilização do *Jenkins*, a equipe do OwnCloud teve muitos problemas relacionados a atualização, espaço em disco, gerenciamento dos *containers*, além de problemas de lentidão (JACKSON, 2018).

Com o intuito de minimizar tantos impeditivos de CI, a equipe de engenheiros e arquitetos da OwnCloud optou por adotar a ferramenta Drone, também um sistema de CI de código aberto com empacotamento em *containers*. Além de eliminar os problemas de CI existentes, esta mudança permitiu que a empresa aumentasse a quantidade de testes de unidades e integrações feitas em recursos novos e ou atualizados (JACKSON, 2018). Conforme relatado por um dos desenvolvedores da Nextcloud (WURST, 2018), a empresa utiliza atualmente como práticas de CI as ferramentas *open source Xmllint* e *Travis*.

3.4. Qualidade de Software

A premissa básica de vários estudos sobre a qualidade de *software* é que a qualidade do produto depende da qualidade do processo de desenvolvimento (KOSCIANSKI; SOARES, 2007). Neste sentido, a busca por um *software* de qualidade passa naturalmente pela necessidade de melhoria no processo de desenvolvimento.

Existem diversas evidências em projetos *open source* sobre a utilização de boas práticas de melhoria de qualidade nas aplicações produzidas, conforme relatado, por exemplo, em (*Developer's Manual* Owcloud), (*Developer's Manual* Nextcloud).

Sendo o foco no controle da codificação, as duas etapas conhecidas para consideração da qualidade nos projetos *OSS* são revisão do código-fonte e testes automatizados (STEINMACHER et al., 2018) e (CHEN; JIANG, 2017). A primeira é realizada, em grande parte dos projetos, por um ou mais moderadores, isto é, integrador(es) que revisa(m) as requisições de pacotes de código (*pull-requests*) antes de integrá-los ao *branch* principal e torná-los parte do produto (CHEN; JIANG, 2017). A etapa de testes automatizados trata da codificação de testes que são executados de forma automática e que devem garantir a cobertura dos módulos principais executados sem erro

(STEINMACHER et al., 2018). Projetos OSS grandes e mais controlados utilizam como critério não somente a validação da qualidade do código apresentado (existência de anti-padrões) (CHEN; JIANG, 2017), mas também a existência da codificação dos testes para aceitar ou não uma submissão de alteração (STEINMACHER et al., 2018). As comunidades OwnCloud e Nextcloud não são diferentes nestes aspectos. Os manuais de desenvolvedores de ambas as empresas exprimem objetivamente e da mesma forma que qualquer submissão de *pull-request*, a submissão não é aceita antes de ser revisada no mínimo por dois colaboradores ainda que seja referente a uma única linha (*Developer's Manual* Nextcloud).

De acordo com (CHEN; JIANG, 2017) e (*Developer's Manual* Owcloud), as comunidades *OwnCloud* e *Nextcloud* não aceitam pedidos de *pull-request* enviados com código não testado. Testes de unidade são realizados com o *framework PHPUnit* e testes de interface utilizando *Selenium* com apoio da aplicação *Source Labs*. Referente às documentações ambas, OwnCloud (*https://doc.OwnCloud.com/)* e Nextcloud (*https://docs.Nextcloud.com/),* possuem um *wiki* específico para as documentações, onde mantém manuais de usuários, administração e desenvolvedor para cada uma das versões disponíveis, além de manterem um manual de orientação de contribuições.

3.5. Gerenciamento de aquisições

Grande parte dos produtos *OSS* licenciados utilizam de um tipo de licenciamento específico para projetos de código aberto, como a *GPL*. Em vez de comprar o produto, você compra suporte e serviços, garantindo qualidade e continuidade *(Why the (A) GPL is great for business users)*.

Desenvolvidas com informações de advogados de todo o mundo, as licenças de *software GPL* e sua derivada para *software Web*, *AGPL*, são as licenças *copyleft* mais usadas, aceitas, comprovadas e testadas pela comunidade de software, e projetadas principalmente para proteger o destinatário do código contra abuso por parte do fornecedor *(Why the (A) GPL is great for business users)*.

Tanto o OwnCloud quanto o Nextcloud são licenciados sobre a *AGPL v3*. Sendo que, em junho de 2017, a Nextcloud tomou a iniciativa de ser a primeira solução de sincronização e compartilhamento de arquivos corporativos para verificar a conformidade total da licença por meio do *OpenChain,* um projeto importante da *Linux Foundation* que identifica os principais processos recomendados para o gerenciamento eficaz de código aberto *(Plans and Pricing for Nextcloud Files)*.

4. CONCLUSÃO

Este trabalho realizou a identificação das práticas da Engenharia de *Software* aplicadas em projetos de *software* livre de armazenamento em nuvem, o OwnCloud e o Nextcloud. Foi constatado que, existem diversas pesquisas referentes às práticas de Engenharia de *Software* em aplicações *open source.* Todavia, em sua maioria, o maior foco é na codificação, não se atentando aos demais processos do desenvolvimento, como as documentações, controle efetivo do gerenciamento das mudanças e processos de gestão da qualidade. Embora boa parte dos processos de desenvolvimento não sejam abertos ao público, acredita-se que muitos processos estão incorporados e inclusive procedimentos de melhoria de processos são realizados no dia a dia de algumas das comunidades *OSS*, principalmente em projetos grandes como OwnCloud e Nextcloud que possuem parte dos seus produtos ou serviços comercializados e uma vasta equipe de colaboradores efetivos.

Foi concluído que, no projeto Nextcloud*,* por se tratar de um projeto *fork* recente, a maioria das características documentadas dos processos de fábrica do *software* ainda estão semelhantes ou exatamente iguais ao OwnCloud. Porém, verificou-se uma evolução na integração

contínua com a adoção da aplicação *Drone* por parte da OwnCloud, o que mostra a tentativa pela busca por algum diferencial competitivo.

Em trabalhos futuros, foi sugerido um estudo que compare outras aplicações de armazenamento de dados em nuvem, elencando as diferenças entre as aplicações estudadas.

REFERÊNCIAS

AFZAL, W. et al. Software test process improvement approaches: a systematic literature review and an industrial case study. **Journal of Systems and Software**, v. 111, p. 1-33, 2016. https://doi.org/10.1016/j.jss.2015.08.048

ALGARRÃO, P. A. **Processos e fases de gerenciamento de projetos**. Universidade de Cândido Mendes, 2018.

BHARTIYA, S. **OwnCloud bifurcou-se para criar o Nextcloud**. 2016. Disponível em: https://www.cio.com/article/3078173/cloud-computing/OwnCloud-forked-to-create-Nextcloud.html. Acesso em: jul. 2018.

CHEN, B.; JIANG, Z. M. Characterizing and detecting anti-patterns in the logging code. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, 39. **Proceedings…** Argentina, 2017.

DIAS, A. F. **O que é Gerência de Configuração de Software**?. maio, 2016. Disponível em: https://blog.pronus.io/posts/o-que-eh-gerencia-de-configuracao-de-software/. Acesso em: mar. 2018.

FELDMAN, D. **A história do OwnCloud e Nextcloud**. Disponível em: https://civihosting.com/blog/Nextcloud-vs-OwnCloud/. Acesso em: jul. 2018.

GAROUSI, V; FELDERER, M.; HACALOĞLU, T. What we know about software test maturity and test process improvement. **IEEE Software**, v. 35, n. 1, p. 84-92, 2018. https://doi.org/10.1109/MS.2017.4541043

GAROUSI, V.; FELDERER, M.; HACALOĞLU, T. Software test maturity assessment and test process improvement: a multivocal literature review. **Information and Software Technology**, v. 85, p. 16-42, 2017. https://doi.org/10.1016/j.infsof.2017.01.001

HECK, P.; ZAIDMAN, A. An analysis of requirements evolution in open source projects: recommendations for issue trackers. In: INTERNATIONAL WORKSHOP ON PRINCIPLES OF SOFTWARE EVOLUTION, August 19-20, 2013. **Proceedings…** Saint Petersburg, Russia. 2013.

JACKSON, J. **How Drone Solved OwnCloud's Continuous Integration Woes**. Disponível em: https://thenewstack.io/how-drone-solved-OwnClouds-continuous-integration-woes/. Acesso em: jul. 2018.

KOSCIANSKI, A.; SOARES, M. S. **Qualidade de Software**. 2. ed. São Paulo: Novatec, 2007.

LUCASSEN, G. et al. Forging high-quality user stories: towards a discipline for agile requirements. In: IEEE. **Proceedings...** 2015.

STEINMACHER, I. et al. Almost there: a study on quasi-contributors in open source software projects. In: INTERNATIONAL CONFERENCE ON SOFTWARE ENGINEERING, SWEDEN. **Proceedings...** 2018.

THOMÉ, B.; HENTGES, E.; GRIEBLER, D. Computação em nuvem: análise comparativa de ferramentas open source para iaas. In: ESCOLA REGIONAL DE REDES DE COMPUTADORES (ERRC),11. **Anais…** 2013.

WURST, C. Developer at Nextcloud GmbH, informal guide of Continuous integration best practices. Mar. 2018. https://blog.wuc.me/2018/03/06/validate-Nextcloud-info-xml.html. Acesso em: jul. 2018.