

Avaliação de Atributos de Qualidade em Ambiente de Software Livre e Aberto

Jussara Pimenta Matos¹

¹ Instituto Federal de Educação, Ciência e Tecnologia de São Paulo
Avenida Salgado Filho, 3501, Guarulhos – SP. CEP 07115-000.

jussara.pimenta@ifsp.edu.br

***Abstract.** This paper evaluates the tactics to apply on web platform applications, regarding an open source software development and that attends the quality attributes. The selected quality attributes are performance, availability and security. This type of application needs to meet high volume of requests generated and should be able to meet these requests in a timely manner. Moreover, the open source software development is growing, both in universities and in the public and private companies, which adds a challenge to developers to meet the needs of the software and its quality.*

***Resumo.** Este trabalho avalia táticas para o desenvolvimento de aplicações para plataforma web, considerando um ambiente que utilize software livre e aberto, de maneira que atenda aos atributos de qualidade. Os atributos de qualidade selecionados são desempenho, disponibilidade e segurança de acesso. Esse tipo de aplicação necessita atender a grandes volumes de requisições geradas e deve ser capaz de atender a essas requisições em tempo hábil. Por outro lado, o uso de software livre e aberto vem crescendo, tanto nas universidades quanto nas empresas públicas e privadas, o que acrescenta um desafio aos desenvolvedores em atender as necessidades do software quanto a sua qualidade.*

1- Introdução

Cada vez mais sistemas corporativos são construídos adotando-se uma plataforma de software livre e aberto. E, em muitos casos, é necessário atender a grandes volumes de requisições geradas pela comunidade *internet*, com capacidade de responder a essas requisições em tempo hábil. Um dos desafios é construir um sítio de sucesso para atender milhões de requisições diárias e, se houver uma concentração de requisições num período curto de tempo, pode colocar em risco a disponibilidade do sistema [Long, 2006]. A utilização de software de código livre e aberto vem crescendo, tanto nas universidades quanto nas empresas do governo e privadas [Özel; Çilingir; Erkan, 2006], para diferentes tipos de aplicações. Empresas com tradição em software proprietário estão colaborando no seu desenvolvimento [IBM, 2012; ORACLE, 2012; SUN, 2012].

O estudo acerca da qualidade de software é analisado através de diferentes perspectivas, principalmente em relação à qualidade de processos e à qualidade de produtos. Existem diversas normas e modelos de qualidade, tais como, ISO/IEC 12207 [ISO/IEC-12207, 2008], ISO/IEC 15504 [ISO/IEC-15504, 2003], MPS.BR [Softex, 2012], ISO/IEC 25010 [ISO/IEC-25010, 2011] e ISO/IEC 14598 [ISO/IEC-14598,

1999]. Um vocabulário próprio tem sido desenvolvido para os atributos de qualidade e há uma variedade de taxonomias e definições do problema de definições e a caracterização de cada atributo [Moro; Falbo, 2008; Guizzardi, 2005; Matos, 2005]. Conforme apresentado em Bass, Clements e Kazman (2003), a qualidade das aplicações *web* pode ser melhorada com a adoção de táticas. Existem trabalhos relacionados que tratam de *web* e atributos de qualidade [Böger et al., 2008; Chiuchi, 2011], todavia, este avalia a aplicabilidade das táticas para ambiente de software livre e aberto.

2- Qualidade de Software

A terminologia adotada para qualidade de software relacionado aos atributos e táticas tem como base a publicação de Bass, Clements e Kazman (2003). Os atributos de qualidade disponibilidade, desempenho e segurança são típicos para uma aplicação *web*. Estes apresentados de forma isolada podem gerar problemas de definição de seus conceitos, dificultado o seu entendimento e sua concretização [Shatnawi, 2010] e não têm significado isoladamente, é necessário um contexto para qualificá-los.

Para caracterização do sistema de software existem diversas considerações a serem feitas [Matos, 2005], entre elas, verificar o grau de novidade do sistema a ser desenvolvido, a integração com sistemas legados, se as interfaces entre o sistema legado e o novo sistema a ser desenvolvido necessitam de adaptações. O processo de coletar e organizar as informações ajuda a distinguir os aspectos relevantes do domínio da aplicação. Os softwares que as aplicações *web* executam são distribuídos, implementados em várias linguagens de programação e estilos, incorporam componentes de terceiros e devem permitir interfaces com usuários, outros sítios e banco de dados.

Para um sistema de comércio eletrônico, as funções de interação com os usuários/navegação são realizadas através de um *Web Browser*, as regras de negócio e as funções de aplicação são alocadas a servidores de aplicação e servidores de transação; a camada de serviços de dados é alocada a uma base de dados; e na maioria dos casos é conectada a um sistema legado. Uma plataforma comercial, de hardware e/ou de software, pode direcionar a maior parte do desenvolvimento, implicando em cuidados especiais para mapear os atributos de qualidade.

Táticas são decisões de projeto tomadas pelo arquiteto com o objetivo de atender os requisitos para os atributos de qualidade. As táticas de disponibilidade podem ser divididas em três classes: táticas de detecção, recuperação e prevenção de defeito. As táticas de desempenho podem ser divididas em três classes: táticas para controle de demanda de recursos, gerenciamento de recursos e arbitragem de recursos. As táticas para alcançar a segurança de acesso podem ser divididas em três classes: táticas para resistir a ataques, para detectar ataques e para recuperar de ataques.

3- Ambiente de Software de Código Livre e Aberto

Software de código livre e aberto é um termo amplo utilizado para designar os softwares com códigos sob licença que permitem sua execução para qualquer propósito, sua avaliação, modificação e redistribuição. É uma alternativa a soluções comerciais de código fechado e é um modelo que cresce cada vez mais nas indústrias de software

[Kivekäs, 2008; GNU PROJECT, 2012]. Ao contrário do que geralmente é afirmado, softwares de código livre e aberto não são necessariamente gratuitos [Monden; Okahara, 2011], e podem ter fins comerciais. Geralmente não tem custos iniciais ou esses são mínimos, além da liberdade para sua execução.

Apesar de boa parte dos softwares de código livre e aberto aparentemente não acarretar em custo direto, custos indiretos, tais como, suporte técnico, manutenção e desenvolvimento devem ser avaliados antes de sua adoção. Assim como os softwares proprietários, os softwares de código livre e aberto possuem uma variedade de licenças, cada uma delas impondo diferentes permissões e restrições. Para a plataforma, o sistema operacional indicado é o Linux, o termo GNU/Linux se refere ao sistema operacional de núcleo Linux, mais o conjunto de utilitários e aplicativos desenvolvidos pelo projeto GNU. Em conjunto outros softwares livres, o Linux pode formar um ambiente seguro e estável para computadores de mesa, servidores e sistemas embarcados.

Java Platform, Enterprise Edition (Java EE) é o padrão da indústria para desenvolvimento de aplicações Java portáteis, robustas, escaláveis e seguras no lado do servidor. Construída sobre a base da Plataforma Java, Edição Padrão (Java SE), Java EE fornece *web services*, modelo de componentes, gerenciamento e interfaces de comunicação, para implementação de aplicações *web* e arquiteturas orientadas a serviço [SUN, 2012]. O servidor de aplicação é uma plataforma *middleware* padrão para aplicações de negócios baseadas em *web*, possuem um servidor *web* embutido, assim como, um *container* de aplicação. As características que definem um servidor de aplicação típico são: interatividade, concorrência, distribuição, heterogeneidade, oferta de múltiplos serviços, e atendimento a alto volume de transações. O servidor de aplicação Jboss é um projeto de código livre e aberto, o JbossAS atualmente atua sob licença Lesser GNU Public License (LGPL), e tem várias de comunidades de desenvolvedores [JbossAS, 2012].

Servidor web é um serviço que trata de requisições e respostas http a clientes fazendo alterações necessárias para encontrar e entregar o recurso solicitado [Scambray; Shema, 2004]. Aplicações *web* requerem o uso de banco de dados para facilitar o desenvolvimento, organizar e acessar informações. O MySQL é o banco de dados de código livre e aberto, está disponível sob duas licenças uma GPL (community Edition), e outra proprietária tradicional e conhecida [Lee; Ware, 2002; MYSQL, 2012] está disponível em várias plataformas tais como, Linux, Windows, OS/X, Netware.

4- Avaliação das Táticas

Para o sistema são consideradas duas máquinas para atuarem como servidores web, duas para atuarem como servidores de aplicação, duas para banco de dados. A solução considera que os servidores recebem uma instalação do sistema operacional Linux Ubuntu Server Edition. Em cada máquina *Web Server* é feita uma instalação do Apache HTTP Server [Apache, 2012]. Os servidores de aplicação recebem, em cada um, instalações do Jboss e nos servidores de dados tem instalações do Mysql. A figura 1 apresenta a disposição das máquinas e seus respectivos softwares.

Para avaliar as táticas é adotado um sistema de comércio eletrônico denominado sistema de assinaturas de periódicos. O sistema de assinaturas é composto do módulo sítio e o do módulo administrativo. O módulo sítio é uma camada que implementa a interface com o sistema legado de assinaturas que é composto por um *mainframe*, onde

é realizado o processo de assinatura e de cobrança de forma *batch*. Para processamento *online* de cobrança é utilizado outro sistema, que se encarrega de atualizar os dados no sistema que reside no *mainframe*. O módulo sítio possui funções de assinatura e de retenção de assinantes e é acessível ao usuário através da internet com interface *web*, onde ele pode consultar e assinar um catálogo de periódicos. O módulo administrativo também possui interface *web* e está disponível através da *internet*, restritivamente aos administradores, que gerenciam a configuração do sistema e realizam a manutenção dos dados. O módulo administrativo faz uso do serviço de diretórios e de nomes para autenticação do usuário. Em seguida são apresentados os principais pontos de avaliação das táticas adotadas.

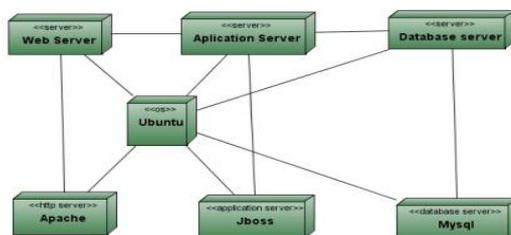


Figura 1- Disposição dos softwares entre as máquinas

Na aplicação das táticas de disponibilidade tem-se: o tratamento de exceções que é obtido utilizando, no código, blocos *try/catch/finally* [Sun, 2012], tanto no módulo sítio como no módulo administrativo são utilizados esses blocos e em casos de exceções ações são tomadas; redundância na camada de dados (*database server*) são utilizadas as facilidades fornecidas pelo *Mysql Cluster* que combina uma arquitetura para agrupamento de base de dados tolerante a falhas, onde todas as atualizações de dados realizadas em qualquer um dos nós de aplicação são replicadas em todos os nós de dados e no caso de falha o outro assume todo o serviço; a redundância na camada de aplicação é realizada utilizando facilidades fornecidas pelo servidor de aplicação *Jboss* e o módulo *mod_jk* para o *Apache*, esse módulo também oferece serviço de balanceamento de carga e *failover*; redundância no servidor HTTP, como os servidores *apaches* não precisam manter o estado dos usuários, é aplicado o algoritmo de *round-robin* para distribuir igualmente a carga e não é preciso manter os dados em sessão, sendo de responsabilidade do servidor de aplicação.

Na aplicação das táticas de desempenho adotou-se: para reduzir as despesas computacionais, os arquivos estáticos (arquivos de imagens, javascript e htmls) são colocados à disponibilidade do servidor *web* que trata da entrega diretamente, reduzindo assim o tráfego entre este e o servidor de aplicação; para melhorar eficiência computacional é aplicado o processamento assíncrono em algumas funções do sistema onde a resposta não é necessária ou não é necessária imediatamente, é utilizada a interface de serviços de mensagem Java (*JMS*); para o escalonamento dinâmico de prioridade *round-robin* é o algoritmo utilizado para a distribuição de carga; para concorrência, o servidor *Apache* pode atender as requisições *http* e *https* de usuários simultaneamente através de threads ou múltiplos processos (se configurado para utilizar *fork*), o *Jboss* recebe as requisições do *apache* através de *servlets* que são componentes Java, que atendem a requisições de maneira concorrentes, o banco de dados *Mysql* atende várias conexões simultâneas realizando operações concorrentemente e as

transações garantem o controle de atomicidades, consistência, isolamento e durabilidade.

Na aplicação das táticas de segurança o objetivo de resistir a ataques e são: autenticação é aplicada tanto no módulo sítio quanto no módulo administrativo, é utilizado o serviço de autenticação para a plataforma Java EE que é integrado com o servidor de aplicação Jboss; autorização, idem a tática de autenticação, é utilizado o serviço de autenticação para a plataforma Java EE; além do sigilo de dados que usam da criptografia, o protocolo SSL possui outras características como proteção da integridade de dados, através da verificação do valor *hash* das mensagens geradas e os algoritmos, tanto para a chave assimétrica quanto para o *hash*, são selecionados através da negociação entre o cliente e o servidor.

5- Conclusão

Esse trabalho avaliou como táticas podem ser utilizadas no desenvolvimento de aplicações para plataforma *web*, considerando um ambiente que utilize software livre e aberto. Pode-se afirmar, de uma forma qualitativa, que parte dessas táticas está presente nos softwares propostos.

A plataforma JAVA EE apresenta facilidades que auxiliam no atendimento dos atributos de qualidade selecionados, e implementações, tais como, as que utilizam o JBoss, possuem táticas presentes em sua arquitetura, o que facilita o projeto. Os softwares de código livre e aberto não são obstáculos para atender a qualidade desejada para um sistema de software, eles apresentam táticas nativas, assim como, os softwares comerciais.

Como trabalho futuro é necessário avaliar outros atributos de qualidade para os sistemas de softwares baseados na arquitetura *web*, atributos tais como, usabilidade, escalabilidade, portabilidade. É importante ressaltar que os atributos de qualidade influenciam um ao outro, alguns são observáveis em tempo de execução, outros na estruturação do sistema e alguns deles não afetam diretamente as decisões de projeto. Logo, também deve ser considerado um trabalho relativo à mensuração desses atributos e apresenta uma avaliação da relação entre estes.

Referências Bibliográficas

- APACHE (2012), Apache Software Foundation, disponível em: <http://www.apache.org/>.
- Bass L.; Clements P.; Kazman R. (2003), “Software Architecture in Practice”, Second Edition, Addison Wesley.
- Böger, D.S.; Wangham, M.S.; Fraga, J.; Mafra, P; Um Modelo de Composição de Políticas de Qualidade de Proteção para Serviços Web Compostos, VIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais, 2008.
- Chiuchi, C.A., Diretrizes para a criação de aplicações *web* com ênfase em portabilidade e eficiência, Dissertação apresentada na Uiversidade Estadual Paulo Mesquita, 2011.
- GNU PROJECT (2012), “The Free Software Definition”, disponível em: <http://www.gnu.org/philosophy/free-sw.html>.
- Guizzardi, G. (2005), “Ontological Foundations for Structural Conceptual Models”, Universal Press, The Netherlands.

- IBM (2012), <http://www.ibm.com>.
- ISO/IEC (2011), ISO/IEC 25010:2011 – “Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models”.
- ISO/IEC (2003), ISO/IEC 15504 – “Information Technology – Process Assessment”.
- ISO/IEC (2008), ISO/IEC 12207 – *System and Software Engineering - Software Life Cycle Processes, 2nd edition*.
- JBossAS (2012), “JBoss Application Server”, disponível em: <http://labs.jboss.com/jbossas>.
- Kivekäs, K. (2008), "Free/Open Source Software Development: Results and Research Methods", Master's Thesis, University of the Helsinki, June.
- Lee J.; Ware B. (2002), “Open Source Web Development with LAMP: Using Linux, Apache, MySQL, Perl, and PHP”, Addison Wesley, December.
- Long S. (2006), “Understanding the Role of Core Developers in Open Source Development”, Journal of Information, Information Technology, and Organizations, Vol.1.
- Matos J.P. (2005), “Concepção de uma Arquitetura de Software com Base nos Estilos de Arquitetura e Requisitos Não Funcionais”, tese de doutorado apresentada na Escola Politécnica da Universidade São Paulo, Agosto.
- Monden, A., Okahara, S. (2011), “Guilty or Not Guilty: Using Clone Metrics to Determine Open Source Licesing Violations”, IEEE Software, March/April.
- Moro, R.D., Falbo, R.A. (2008), “Uma Ontologia para o Domínio de Qualidade de Software com Foco em Produtos e Processos de Software” In Proc. of the 3rd Workshop on Ontologies and Metamodels for Software and Data Engineering (WOMSDE'08).
- Mysql (2012), disponível em: <http://www.mysql.com/>.
- Özel, B., Çilingir, C.B., Erkan, K. (2006), “Towards Open Source Software Adoption Educational, Public, Legal and Usability Practices”, Workshop Proceedings, Italy, June.
- ORACLE (2012), <http://www.oracle.com>.
- Scambray J.; Shema M. (2004), “Hacking Exposed™ Web Applications”, McGraw-Hill/Osborne.
- Shatnawi R. (2010), “A quantitative Investigation on the Acceptable Risk Levels of Object-Oriented Metrics in Open-Source System”, IEEE Transactions on Software Engineering, Vol. 36, no. 2, March/April.
- Softex (2012), <http://www.softex.br>.
- SUN (2012), disponível em: <http://www.sun.com>.