Uma proposta de método para análise e desenho de uma Arquitetura Orientada a Serviços

José Valdvogel de Almeida Junior *

2014, v-1.0

Resumo

Com a evolução tecnologica e o aumento da computação distribuida a arquitetura orientada a serviço (SOA) tem se tornado uma abordagem cada vez mais utilizada como paradgima arquitetural para projetos de software. No entanto, alguns projetos que utilizam esse paradigma não obtiveram sucesso na sua adoção pelo fato de sua aplicação ir além dos limites tecnologicos. Esse trabalho visa estudar os problemas ocorridos e realizar uma proposta de método que auxilie arquitetos de software a montar um processo de desenvolvimento voltados para os princípios do paradigma orientado a serviço.

Palavras-chaves: SOA. Modelagem. Serviço. Catalysis. Arquitetura.

Introdução

Problemas na construção de software são desafios recorrentes desde que o desenvolvimento de aplicações tornou os processos de negócio mais ágeis e o investimento em TI uma necessidade. Estudos econômicos na década de noventa concluiram que a produção e o uso de TI contribuiu em mais da métade para o crescimento da produtividades americana e que o investimento em software subiu de US$ 82 bilhões de dólares em 1995 para US$149 bilhões em 1999 (RIEMENSCHNEIDER; HARDGRAVE; DAVIS, 2002).Os gastos com tecnologia da informação no Brasil atingirão US$ 125,3 bilhões em 2015. A estimativa, divulgada pelo Gartner, aponta para um crescimento da ordem de 5,7% sobre os US$ 118,5 bilhões projetados pela consultoria para 2014. Os gastos mundiais vão superar US$ 3,9 trilhões em 2015, um aumento de 3,9% em relação a 2014. (COMPUTERWORLD, 2014). Com base nessa demanda o desenvolvimento de software ganhou um papel de importante destaque na vida organizacional de qualquer empresa.

Arquitetura orientada a serviço (SOA) atualmente é muito utilizada em empresas de diversos ramos que pretendem criar estruturas de software com flexibidade, integridade e reuso. No entanto, muito dessas empresas fracassam devido a trabalharem da mesma forma que trabalhavam com outros estilos arquiteturais. As estruturas e processos de sistemas antigos são inadequados e podem portanto levar a uma desaceleração ou perda de oportunidades(ROTHENBERGER; HAINES, 2010). Esse artigo tenta propor um método para construção de software quando SOA é sugerido como estilo arquitetural para empresas que precisam de funcionalidade altamente distribuida, arquitetura desacoplada e padronização de interfaces.

1 Arquitetura Orientada a Serviço

Existem várias definições para SOA e normalmente ela esta associada à tecnologia Web Services, porém sua definição vai muito além, pois está em torno de conjunto de princípios e padrões. SOA é uma arquitetura de software que define o uso de serviços para dar suporte aos requisitos do usuário(LEE; CHAN; LEE, 2006). As características desses serviços são baixo acoplamento, contrato de serviço, autonômia, abstração, reuso, composição, gerenciamento de estado e descoberta(ERL, 2005). Características essas oriundas da orientação de objetos e da computação distríbuida que visam aspectos de hardware com máquina independentes e autônamas, e software onde um determinado sistema se apresenta ao usuário como uma única máquina. Características que se fazem presente atualmente pelo crescimento tecnologico, mas que por outro lado vem apresentando problemas constantes pelo fato de os métodos utilizados para sua concepção serem os mesmos.

2 Desafios nas fases de desenvolvimento

Desde a crise de software uma série de métodos foram criados para auxiliar na construção de peças de software voltados para o paradigma de programação procedimental. Com a mudança do paradigma procedimental para o orientado a objeto o foco no desenvolvimento ágil ganhou destaque pela necessidade das empresas serem cada vez mais ágeis. Com isso as metodologias clássicas que continham fases de desenvolvimento bem definitas começaram a ter cortes para atender a demanda no tempo e custo combinados. Cortes que nem sempre trouxeram benefícios para os desenvolvedores e que por muitas vezes trouxeram problemas para desenvolvedores e clientes.

De acordo com o estudo realizado por Rothenberger com empresas que haviam adotado SOA como pararadigma arquitetural, algumas mudanças na forma de planejar, analisar, desenhar, implementar e testar serviços foram necessárias para que a iniciativa arquitetural fosse desenvolvida com sucesso, pois foi constatado que existiam diferenças significantes no contexto de SOA com Web services em comparação ao desenvolvimento de software anterior (ROTHENBERGER; HAINES, 2010). Abaixo segue um resumo das fases e as mudanças necessárias para adoção de SOA.

Tabela 1 – Resumo dos problemas nas fases de desenvolvimento com SOA

Fase	Importantes características e mudanças
Planejamento	Experiência empírica; comunicação entre stakeholder e TI; utilização de padrões;
Análise	Necessário mais rigor do que demais arquiteturas;
Desenho	Desenho de contratos; Granularidade; Atenção aos serviços remotos;
Implementação	Interoperabilidade; Gerenciamento e rastreabilidade;
Teste	Ciclo de vida; Segurança ; Desempenho;

Em seu estudo Rothenberger deixa claro que as fases de planejamento, analise e desenho necessitam de uma abordagem mais tradicional, pois alguns aspectos do desenvolvimento orientado a serviço podem necessitar de mais disciplina e mentalidade holística do que os cenários de desenvolvimento antigos. Dessa perspectiva as abordagens de desenvolvimento ágeis devem constituir uma boa opção para composição de aplicações clientes, enquando os serviços que compõem a infraestrutura podem necessitar de abordagens que enfatizem o planejamento e rigorosos desenhos(ROTHENBERGER; HAINES, 2010). Logo, o foco da pesquisa num primeiro momento é um estudo de um método voltado

para construção de componentes em busca de possíveis melhorias para abordagem orientada a serviço. Posteriormente, o estudo de uma abordagem para formalização de processos de negócio e por fim a união entre o processo formal com as fases para o desenvolvimento de componentes.

2.1 Catalysis

Catalysis é uma técnica e método que fornece desenvolvimento baseado em componente, desenho de alta integridade, desenho orientado a objeto e reengenharia. Usa notação baseada em padrões indústrias como *Unified Modeling Language* (UML) padronizado pelo *Object Modeling Group* (OMG) (DSOUZA; WILLS, 1998). No Catalysis, existem dois conceitos básicos para descrever a análise de desenho dos domínios do usuário. Esses conceitos básicos são os *objetos* que representa um conjunto de informações e funcionalidade e as *ações* representando qualquer coisa que acontece: um evento, trabalho, mensagem, mudança de estado (DSOUZA; WILLS, 1998) conforme indicado na figura 1.

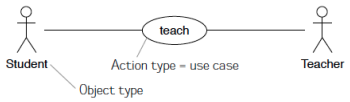

Figura 1 – Objeto participa de ações

Baseado em três constructos básicos - tipo, colaboração e refinamento - fornece a construção de uma grande variedade de modelos e desenhos (ver figura 2). Esses elementos juntos formam um quarto constructo chamado de *framework* que irá representar a união dos constructos em determinado situação.

Tem como base três níveis de modelagem: o domínio do problema ou negócio, o componente ou especificação do sistema (comportamento visível externamente) e o desenho interno do componente ou sistema (estrutura interna e comportamento) indicados na figura 3.

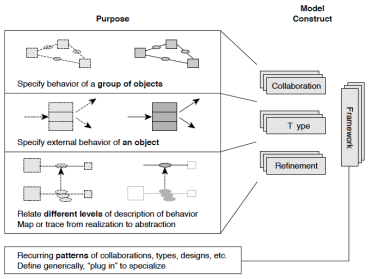

Figura 2 – Três constructos que padronizam um framework

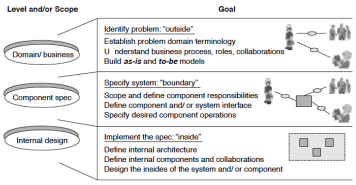

Figura 3 – Três níveis de modelagem

Assim como SOA é regido por uma série de princípios, o método Catalysis também contém três princípios básicos que fazem parte do seu processo - abstração, precisão e acoplamento. Esses princípios que podem ser vistos na figura 4 e nos guiam como referência para a criação de componentes de software mais integros e flexíveis, pois um software contruido sem o uso de componentes bem definidos será inflexível: dificuldade para mudar quando os requisitos mudam (DSOUZA; WILLS, 1998).

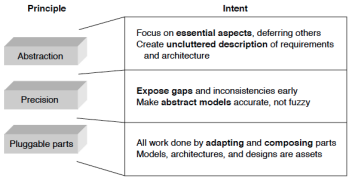

Figura 4 – Três princípios do Catalysis

Catalysis contém uma série de características que são comuns a SOA , logo um estudo a partir de seus fundamentos nos auxilia a pensar em meios adaptativos para que as fases de análise e desenho indicadas no estudo como necessidade de mudança sejam realizadas de uma forma segura e bem formada, levando em consideração caracte-

rísticas básicas como baixo acoplamento e reuso de componentes.

2.2 BPMN

Processos de negócio são essênciais para as empresas sobreviverem hoje, pois sem ele falhas de comunição tornam-se constantes. Para evitar esse problema e como uma forma de eliminar as falhas entre negócio e tecnologia evitando os dois maiores problema de falhas em software (INTERNATIONAL, 2013), uma notação que seja de entendimento de negócio e que auxílie na abstração de idéias se faz viável. Para realizar esse papel a notação de processo de negócio *BPMN* se faz relevante para criação de processos formais que auxiliem na comunicação e ampliem o nível de abstração entre negócio e tecnologia. Definimos processos de negócio como uma coleção de atividades que possuem um ou mais insumos e geram um ou mais resultados que representam agregação de valor ao cliente (HAMMER; CHAMPY, 2009). Em geral, um processo de negócio é uma sequência coerente de atividades com o objetivo de executar um serviço. A saída e o resultado de um processo de negócio é um serviço requerido e consumido por um cliente interno ou externo (SCHEER, 1998). Com isso, podemos a partir de um processo formal elencar serviços candidatos para infraestrutura da nossa arquitetura assim como serviços que ficarão expostos ao consumo dos clientes.

3 Resultados

De acordo com os princípios e características apresentadas anteriormente foi proposto ajustes nas fases do método Catalysis com um maior foco nas fases de análise e desenho utilizando notações existente para compor uma nova técnica. Para tal, podemos definir as seguintes fases a seguir apresentadas na figura 5.

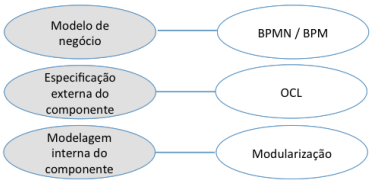

Figura 5 – Possíveis fases a partir do Catalysis

4 Discussão

Como ainda estamos em fase de estudo sobre o método proposto, algumas questões e afirmações em tempo de projeto foram surgindo. Uma dúvida inicial era se o método iria suportar processos informais, porém foi analisado que se não existe uma formalização de processos a abordagem orientada a serviço não é adequada, pois não pode garantir o reuso. Ainda em cima de reuso, existe alguma forma de garantir reuso de um componente nas fases anteriores ao desenvolvimento ? A partir de um processo formal é possível alguma técnica de descoberta dos possíveis serviços que serão utilizados pela empresa ?

Considerações finais

Acredito que o trabalho tem uma grande contribuição para área acadêmica e mercado, pois busca auxílio em um método que foi de grande contribuição para o avanço da modelagem orientada a objeto e que pode ser mais difundida e com grande contribuição para os novos paradigmas e problemas e convivemos hoje.

Referências

COMPUTERWORLD. *Gastos com TI atingirão US125 bilhoes no Brasil em 2015*. 2014. <http://computerworld.com.br/negocios/2014/10/28/gastos-com-ti-atingirao-us-125-bilhoes/-no-brasil-em-2015/>. Citado na página 1.

DSOUZA, D. F.; WILLS, A. C. *Objects, Components, and Frameworks with UML - The Catalysis Approach*. [S.l.]: ADDISON-WESLEY, 1998. Citado na página 3.

ERL, T. *Service-Oriented Architecture Concepts, Technology, and Design*. [S.l.]: Prentice Hall, 2005. Citado na página 2.

HAMMER, M.; CHAMPY, J. *Reengineering the Corporation: Manifesto for Business Revolution*. [S.l.]: HarperCollins e-books, 2009. Citado na página 4.

INTERNATIONAL, T. S. G. *The Chaos Manifesto*. 2013. <http://versionone.com/assets/img/files/ChaosManifesto2013.pdf>. Citado na página 4.

LEE, S. P.; CHAN, L. P.; LEE, E. W. Web services implementation methodology for soa application. *IEEE International Conference on Industrial Informatics*, 2006. Citado na página 2.

RIEMENSCHNEIDER, C. K.; HARDGRAVE, B. C.; DAVIS, F. D. Explaining software developer acceptance of methodologies: A comparison of five theoretical models. *IEEE TRANSACTIONS ON SOFTWARE ENGINEERING*, v. 28, n. 12, Dezembro 2002. Disponível em: <http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=1158287&url=http%3A%2F%2Fieeexplore.ieee.org%2Fiel5%2F32%2F25950%2F01158287.pdf%3Farnumber%3D1158287>. Citado na página 1.

ROTHENBERGER, M.; HAINES, M. How a service-oriented architecture may change the software development process. *ACM*, v. 53, n. 8, p. 135 – 140, Agosto 2010. Citado 2 vezes nas páginas 1 e 2.

SCHEER, A. W. *ARIS - Business Process Modeling*. [S.l.]: Springer, 1998. Citado na página 4.