WS–EVOLV:
Ambiente de Gestão de Evolução de Serviços Web

Lucas José Kreutz Alves (lucaskalves@gmail.com) - Ciência da Computação/UFRGS
Orientadora: Prof.ª Dr.ª Karin Becker - INF/UFRGS

Introdução

Arquiteturas Orientadas a Serviços se baseiam no modelo de computação distribuída cliente-servidor. Tem como características principais:

- Separação das aplicações entre Provedor e Cliente;
- Interfaces entre aplicações cliente e servidor bem definidas (*WSDL*);
- Baixo acoplamento entre sistemas;
- Ciclo da aplicação servidora desacoplado do ciclo de vida das aplicações clientes.

Por causa do baixo acoplamento entre aplicação servidora e aplicações clientes, o provedor deve ser cuidadoso quanto às alterações na interface para que não impacte negativamente seus clientes. O grupo de pesquisa WS-Evolv preocupa-se em criar métodos, conceitos e ferramentas que auxiliem no gerenciamento da evolução de serviços, buscando minimizar o prejuízo de clientes e provedores de Serviços Web através de modelos de versionamento, avaliação de impacto de mudanças, análise de compatibilidade entre versões e apoio ao ciclo de vida dos serviços. O presente projeto contribui com a implementação de um Sistema de Gestão de Versões de Serviços que visa a avaliação precisa do impacto de mudanças em sua interface.

Objetivo

O objetivo desse projeto de Iniciação Científica é: elaborar uma maneira mais eficiente de realizar o versionamento de serviços criando um repositório de versões cujas informações possam ser extraídas e manipuladas fácil e eficientemente. Esse objetivo foi alcançado através de duas modificações na implementação original: a) utilização de um Sistema Gestor de Banco de Dados orientado a Grafos para a persistência do repositório de versões e b) otimização do processo de versionamento baseado em assinaturas de features para detectar mudanças.

Sistema de Gestão de Versões Baseadas em Features

O Sistema de Gestão de Versões de Serviços é baseado em um modelo de versionamento mais granular baseado em features (características). Operações disponibilizadas por um serviço e tipos de dados trocados em requisições são exemplos de features. Este sistema converte uma descrição padronizada de um serviço (WSDL) em uma representação abstrata baseada neste modelo, persistindo as versões das features modificadas em um Repositório de Versões.

O protótipo original criado pelo grupo verificava a necessidade de criação de novas versões baseado na comparação da descrição textual da feature (fragmento da definição WSDL) e de uma árvore de dependências de outras features (Figura 1). O Repositório de Versões neste protótipo consistia de um arquivo XML que segue um esquema pré-definido. Este processo de criação de versões era ineficiente, pois a avaliação de cada feature para verificar a existência de mudanças requer uma varredura em subárvores complexas, e o repositório que contém as versões antigas usadas nas comparações são manipuladas com APIs (Application Programming Interfaces) como DOM e Sax. Estas características resultavam em duas grandes desvantagens: dificuldade de extrair as informações das versões dos serviços para outros componentes do framework WSEvolv e ineficiência para criação e recuperação dos dados.

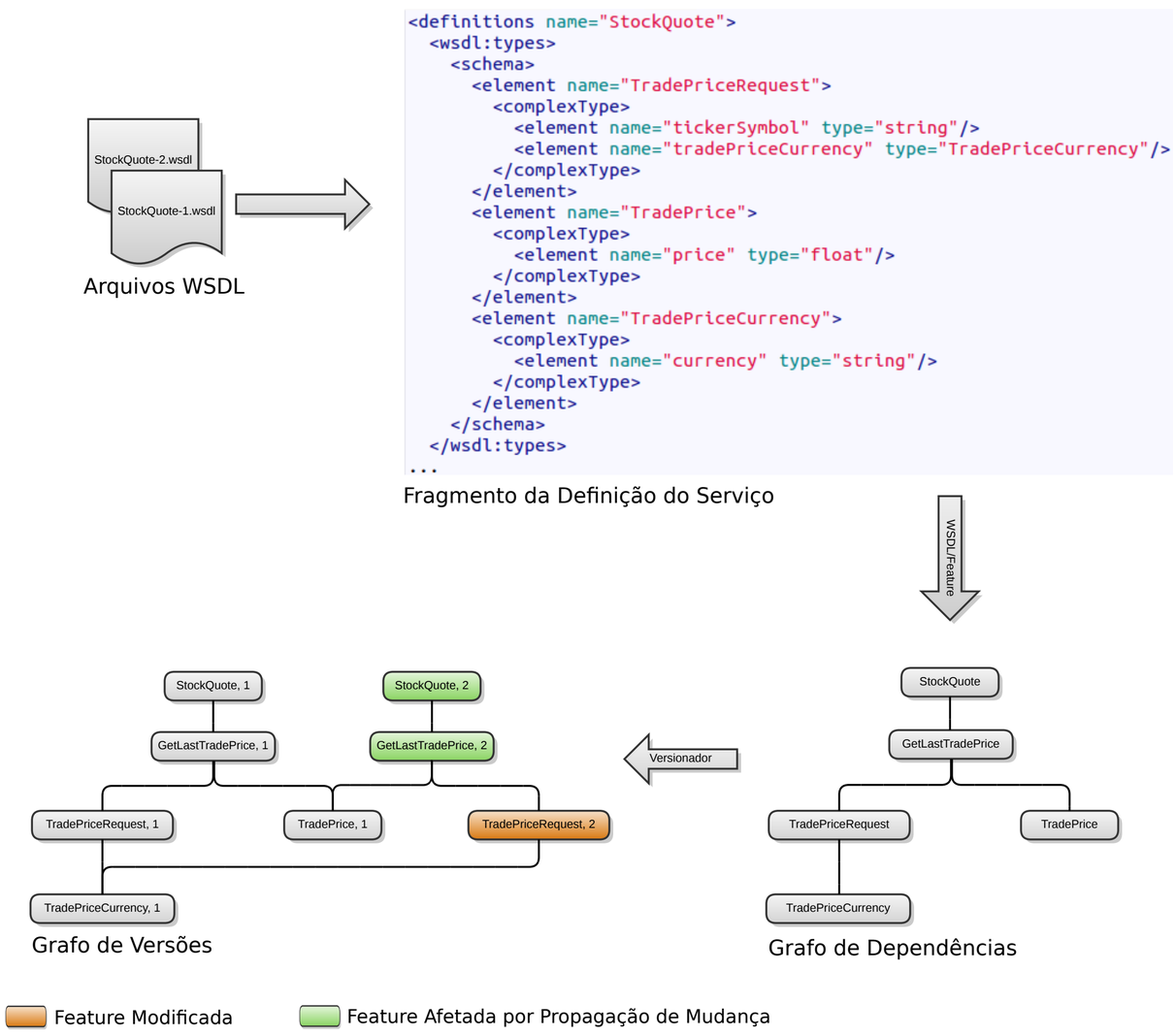

Figura 1: Processo de Versionamento

Banco de Dados para Grafos

Para substituir a persistência do repositório em XML, foi utilizado o OrientDB, um banco de dados para grafos que facilitou o acesso às informações e aumentou significativamente a eficiência da manipulação do repositório. O repositório de versões, segundo o modelo proposto, é representado por um grafo.

Em comparação a bancos de dados relacionais, os bancos de dados para grafos são pouco utilizados e há poucas opções de sistemas gestores. Os dois mais avançados são o OrientDB e o Neo4J. Foi realizada uma pesquisa para definir qual se enquadra melhor ao projeto. O OrientDB foi considerado mais vantajoso pois possui uma licença mais permissiva, mais confiabilidade e mais maneiras de acessar o banco. A implementação do sistema de versionamento utilizando um banco de dados desse tipo aumentou bastante sua eficiência e facilitou consideravelmente o acesso aos dados presentes no grafo de versões. Bancos de dados para grafos vêm sendo utilizados em serviços como o Twitter e o Facebook para representar estruturas que a abordagem relacional não trata com eficiência.

	Usuários [1]	Desenvolvimento	SQL?	Integridade	Licença
OrientDB	620	15 anos [2]	Sim	ACID, BASE, MVCC [3]	Apache 2.0
Neo4J	926	5 anos	Não	ACID	GPL/AGPL

1) Baseado no número de membros do fórum de desenvolvimento
2) Inicialmente, um banco de dados orientado a objetos. Suporta grafos há 2 anos
3) Atomicity, Consistency, Isolation, Durability; Basically Available, Soft-state, Eventually-consistent; Multiversion Concurrency Control

Figura 2: Comparação de Bancos de Dados para Grafos

Versionamento por Asssinatura de Features

Para acelerar a análise de versões, foi criado o conceito de assinatura de feature, que consiste de um código único (*hash*) criado a partir da descrição da feature e da assinatura de suas dependências. Inicialmente, o algoritmo cria o *grafo de dependências* da definição atual do serviço. Utilizando um modelo de varredura *depth-first*, cria a assinatura dos nodos mais básicos até os mais complexos, com dependências. A assinatura dos nodos mais complexos é criada a partir de sua descrição e das assinaturas de suas dependências e a assinatura dos nodos mais básicos é criada somente a partir de sua descrição, já que não possuem dependências.

Dessa forma, o versionamento pode ser acionado a partir de mudanças na assinatura das features, reduzindo a complexidade do processamento, pois anula a necessidade de varrer todas as sub-árvores das dependências para cada feature, levando em consideração que a assinatura de cada é criado em relação às assinaturas de suas dependências.

Eficiência

O tratamento de arquivos XML é bastante custoso tanto em relação ao tempo de desenvolvimento quanto em relação ao tempo e recursos de hardware para o processamento. A adoção do OrientDB para realizar a persistência e a melhoria do algoritmo de versionamento resultaram em um grande ganho de performance em relação ao protótipo original. A utilização do OrientDB para persistir as versões resultou um importante ganho na performance de leitura e escrita do repositório e maior eficiência na busca por dados. A utilização das assinaturas para o versionamento diminuiu o custo do versionamento por diminuir a necessidade de pesquisa nas árvores de dependências.

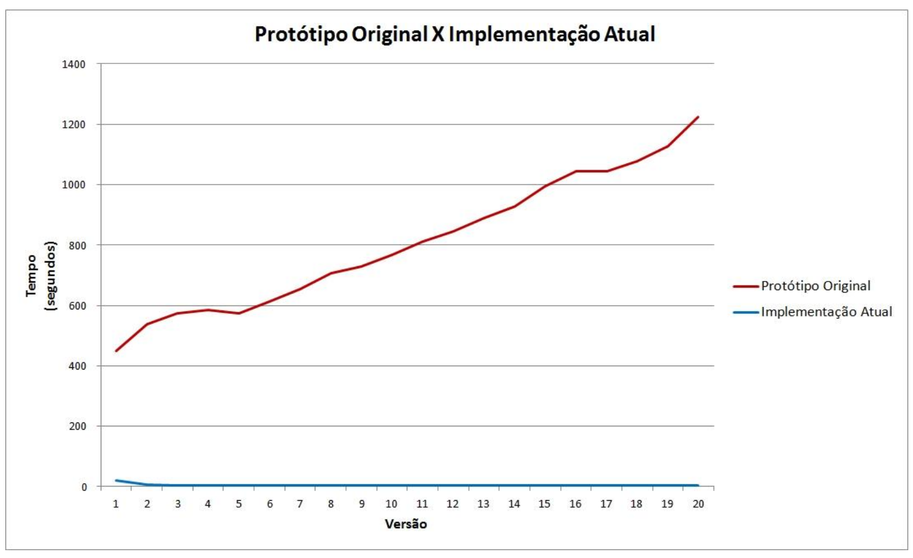

Figura 3: Comparação de Performance entre o Protótipo Original e a Implementação Atual

Considerações Finais

As mudanças propostas no gerenciamento de versões de serviços do modelo criado pelo grupo possibilita sua implantação em sistemas reais com um grau de eficiência maior. O Banco de Dados mostrou-se bastante eficiente e a modificação criada no modo de versionamento simplificou o processo e centralizou o repositório de versões facilitando o acesso a ele. O sistema ainda carece de uma melhor interface para gerência das versões e uma integração maior com outras partes do projeto, como a análise de perfis de uso dos serviços.

Referências

YAMASHITA M., BECKER K., e GALANTE R. A Flexible Approach for Assessing Service Compatibility at Feature Level. SBBD, 2011. p. 105-112.
YAMASHITA M., VOLLINO B., BECKER K., e GALANTE R. Measuring Change Impact based on Usage Profiles. ICWS, 2012. P. p. 226-233.
W3C. WSDL – Web Service Description Language. Capturado em http://www.w3c.org/TR/wsdl (Agosto 2012).
OrientDB - http://www.orientdb.org/ (Agosto 2012).
Neo4J - http://neo4j.org/ (Agosto 2012).
OrientDB the fastest GraphDB available today? Capturado em city.blogspot.com.br/2010/09/orientdb-fastest-graphdb available.html (Agosto 2012).
OrientDB Micro Benchmark. Capturado emhttp://code.google.com/p/orient/wiki/GraphDBComparison (Agosto 2012).
OrientDB Market Share - http://zion-city.blogspot.com.br/2012/05/graphdb-market-share.html (Agosto 2012)