Web Services

Juliano Moraes, Marcus Breda, Paulo Gil, Rafael Medaglia

1. INTRODUÇÃO

Web Services disponibilizam uma maneira de diferentes tipos de aplicação, possivelmente rodando em diversas plataformas e sistemas operacionais interagirem. Originalmente foi criado como uma interface padronizada baseada em tecnologias já conhecidas, basicamente a especificação XML e o protocolo HTTP.

Essa padronização foi desenvolvida pela W3C, um consórcio de empresas de tecnologia que tem como o objetivo criar padrões comuns para conteúdo da Web, apoiada por grandes empresas como a Microsoft, IBM, HP entre outras.

A definição de *Web Services* de acordo com a W3C diz que é um sistema de software responsável por proporcionar a interação entre duas máquinas através de uma rede. Para possibilitar essa interação uma interface descrita em um formato específico, WSDL (*Web Services Description Language*), permite que sistemas interajam com um *Web Service* usando essa interface e enviando mensagens SOAP (*Simple Object Access Protocol*) ou utilizando outros protocolos. As mensagens SOAP basicamente são documentos XML serializados seguindo o padrão W3C enviados em cima de um protocolo de rede.

O WSDL é uma linguagem para descrever *Web Services*, é como um índice dos métodos disponíveis em um certo *Web Service*. Qualquer aplicação pode requisitar esse índice para saber como deve enviar as suas requisições. Independente de tecnologios bastando essas terem acordado quanto a descrição do serviço.

2. CONTEXTO HISTÓRICO

Por volta de 1999, surgiu a necessidade de se padronizar a comunicações entre diferente plataformas (PC, Mainframe, Mac, Windows, Linux entre outros) e linguagens de programaçao (PHP, C#, Java, etc).

Diversos padrões foram propostos, DCOM e CORBA mas nenhum obteve êxito suficiente, tanto pela independencia de plataforma como pelo modelo proposto. Por exemplo, a utilização de RPC (*Remote Procedure Call*) para a comunicação entre diferentes nós pode apresentar problemas de segurança que impossibilitariam sua utilização em ambientes como a Internet devido a requisitos de segurança que podem bloquear esse tipo de mensagens.

Finalmente surgiram os *Web Services* para a comunicação entre as diferentes plataformas, inclusive atualmente muitos sistemas legado estão sendo integrados aos novos sistemas através de *Web Services*.

3. WEB SERVICES

Um *Web Service* é uma noção abstrata que deve ser implementada por um agente concreto. O agente por sua vez é um pedaço de software ou hardware que envia e recebe mensagens. Isso nos permite que tenhamos a mesma interface funcionando com diferentes agentes implementados em diferentes linguagens de programação ou em diferentes sistemas operacionais.

Podemos entender que o *Web Service* é uma interface que precisa ser implementada por uma aplicação independente de linguagem e de sistema operacional para viabilizar a troca de informações entre aplicações.

Na figura abaixo temos duas entidades que conversam entre si, a que provem o serviço (*Web Service*) e a que requisita o serviço (uma aplicação qualquer). Ambas as entidades conversam através de pacotes SOAP que concordam com a descrição do serviço e a semântica utilizada o UDDI (*Universal Description, Discovery and Integration*).

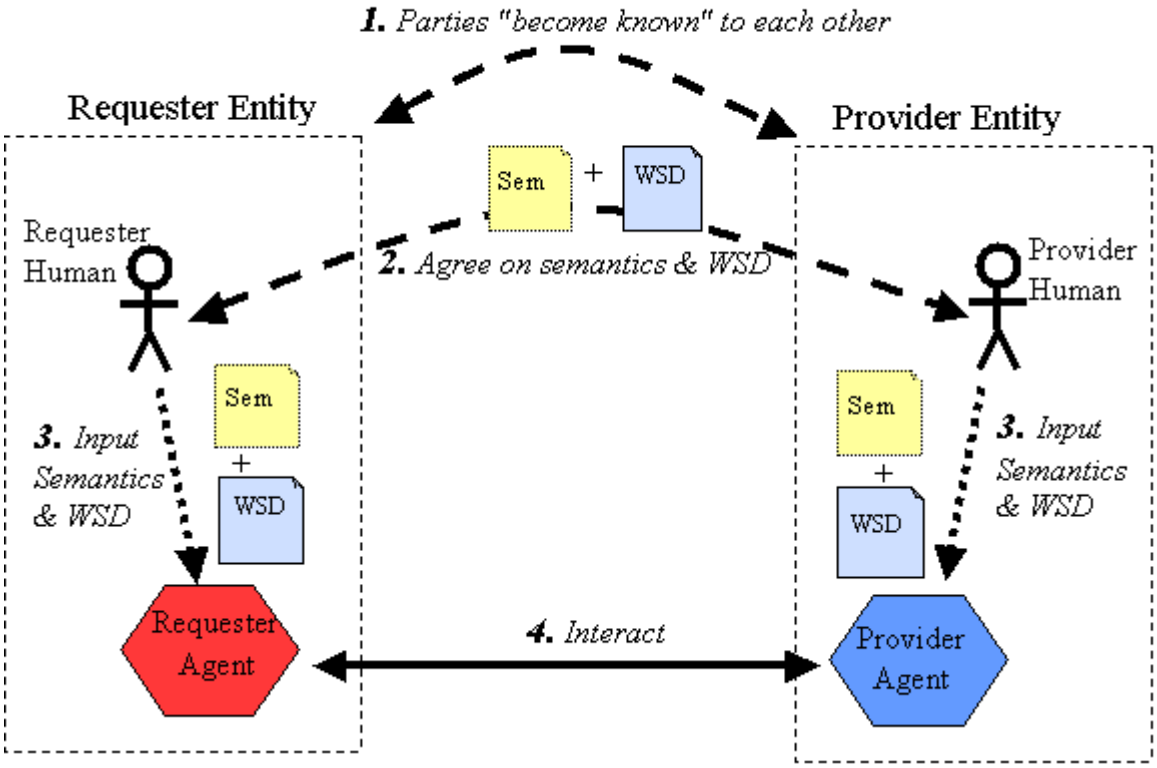

Figura 1: Arquitetura de um *Web Service*.

Entre os vários *frameworks* que existem para auxiliar no desenvolvimento de *Web Services*, um dos mais conhecidos é o *Apache Axis*, uma implementação do servidor SOAP e de APIs para geração e desenvolvimento de aplicações de serviços Web *Open Source* escrito em Java e C++.

Abaixo uma lista de alguns *Frameworks* de *Web Service*:

Nome	Platforma	Tipo	Utilizações	Protocolos
Apache Axis	Java	Cliente/ Servidor	*WS-ReliableMessaging, WS-Coordination, WS-Security, WS-AtomicTransaction, WS-Addressing*	SOAP, WSDL
Java Web Services Development Pack	Java	Cliente/ Servidor	*WS-Addressing*	SOAP, WSDL
Windows Communication Foundation	.Net	Cliente/ Servidor	*WS-Addressing, WS-ReliableMessaging, WS-Security*	SOAP, WSDL
gSOAP	C++	Cliente/ Servidor	*WS-Addressing, WS-Discovery, WS-Enumeration, WS-Security*	SOAP, XML-RPC, WSDL

Entre os protocolos mais importantes citamos os seguintes:

- SOAP;
- UDDI;
- WSDL.

Na figura a seguir a forma te interação entre os protocolos.

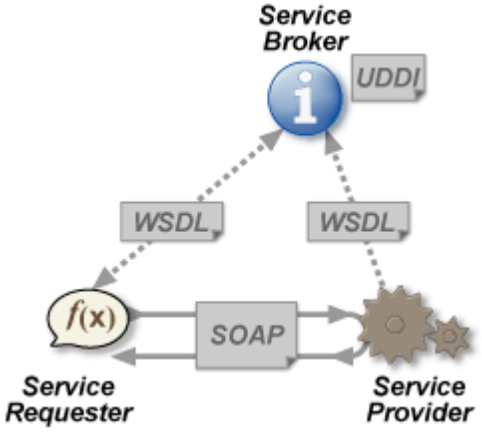

Figura 2: Protocolos.

4. CONCLUSÃO

A principal vantagem do uso de Web Services deve-se a sua Arquitetura Orientada a Serviços (SOA), que difere-se dos sistemas Orientados a Objeto (OO) e de sistemas procedurais devido a sua ligação entre os componentes. Normalmente, sistemas orientados a objetos e sistemas procedurais são conectados através de chamadas com base em nomes e tipos. Em uma SOA a ligação consiste na troca de mensagens que permitem estabelecer vários controles possibilitando restrições para o envio/recebimento de informações e separando o código da assinatura do serviço.

Outra grande vantagem dos *Web Services* é a facilidade e flexibilidade de implementação entre diferentes tecnologias. Principalmente em sistemas legados, onde muitas vezes qualquer alteração torna-se muito cara de outra forma, os *Web Services* são de grande valia.

5. REFERÊNCIAS

- *W3C Web Services Architecture: http://www.w3.org/TR/ws-arch (acesso em 05-Abril-2007)*
- *Wikipédia: (acesso em 05-Abril-2007)*
  - *http://pt.wikipedia.org/wiki/Web_service*
  - *http://en.wikipedia.org/wiki/SOAP*
  - *http://en.wikipedia.org/wiki/Universal_Description%2C_Discovery%2C_and_Integration*
- *D. Fensel and C. Bussler, The Web Service Modeling Framework WSMF - Electronic Commerce Research and Applications, 2002 - http://www1-c703.uibk.ac.at/~c70385/wese/wsmf.paper.pdf (acesso em 05-Abril-2007)*
- *W. Lemahieu: Web Service description, advertising and discovery: WSDL and Beyond, 2001. In J. Vandenbulcke and M. Snoeck (eds.), New Directions in Software Engineering, Leuven University Press, 2001*