Diretrizes para Avaliação de Desempenho de Web Services

Júlio Cézar Estrella[1], Regina H. C. Santana [1], Marcos José Santana [1], Thiago Caprone Tavares [1], Bruno Tardiole Kuehne [1], Julio Cesar Frigo Silva [1], Leonardo Camargo Paccanaro [1]

[1]Departamento de Sistemas de Computação
Instituto de Ciências Matemáticas e de Computação – Universidade de São Paulo
Caixa Postal: 668 - CEP: 13560-970 – São Carlos, SP – Brazil

{jcezar,rcs,mjs,thiagocp,btkuehne}@icmc.usp.br

{j.frigo,leopaka}@grad.icmc.usp.br

Resumo. *Neste artigo são apresentadas e discutidas diretrizes gerais visando à avaliação de desempenho de Web services. Os Web services baseiam-se na arquitetura orientada a serviços (SOA) e possibilitam que empresas ofereçam soluções rápidas que podem ser utilizadas sob demanda. Nesse sentido, um ponto crucial dos Web services é o desempenho alcançado. Diversos atributos de Qualidade de Serviço (QoS) são mapeadas, neste artigo, para entidades participantes de uma SOA, visando definir como pode ser avaliado o desempenho de um Web service. Pontos como o que deve ser avaliado e a maneira como deve ser avaliado em um Web service, são abordados neste artigo. Uma arquitetura denominada WSARCH é utilizada para exemplificar como os atributos de QoS definidos podem ser alcançados no oferecimento de Web services.*

Abstract. *In this paper general guidelines aiming at performance evaluation of Web services are presented and discussed. The Web services are based on the Service Oriented Architecture (SOA) and allow companies offering fast solutions that may be used under demand. In this way, a major point of Web services is the performance reached. Several attributes of Quality of Service (QoS) are mapped in this paper on participating entities of a SOA, aiming at defining the way the performance of a Web service may be evaluated. Points such as what must be evaluated and the way it must be evaluated in a Web service, are approached in this paper. An architecture named WSARCH is used to exemplify the way the QoS attributes defined may be reached when offering Web services.*

1. Introdução

A Computação Orientada a Serviços (COS) (*Service-Oriented Computing*) é um paradigma da computação que utiliza os serviços como elementos fundamentais para o desenvolvimento de aplicações [Papazoglou and Georgakopoulos 2003]. A aplicação da COS na Web é representada pelos Web Services (WSs)[W3C 2002]. Web Services interligam um conjunto de tecnologias, protocolos e linguagens que possibilitam comunicação automática entre as aplicações Web por meio da Internet. WSs são aplicações com baixo nível de acoplamento, que expõem suas funcionalidades por meio de uma descrição de sua interface, tornando-se pública para uso por outras aplicações.

Atualmente, a arquitetura adotada pelos Web Services não considera o suporte a atributos de Qualidade de Serviço. Qualidade de Serviço (QoS) foi definida pelo padrão *ITU X.902* [X.902 2007] como um conjunto de requisitos de qualidade sobre o comportamento coletivo de um ou mais objetos. Na área de redes de computadores o termo QoS refere-se aos mecanismos de controle de reservas de recursos.

Com a integração de WSs como uma solução de negócios em muitas aplicações empresariais, a qualidade de serviço apresentada pelos WSs está se tornando uma preocupação de ambos, os provedores de serviços e clientes. Provedores precisam especificar e garantir a QoS de seus WSs para se manterem competitivos e realizarem o melhor atendimento aos seus processos de negócios. Por outro lado, os clientes objetivam que os serviços solicitados tenham bom desempenho (alta disponibilidade, baixo tempo de resposta, baixa latência).

Alguns trabalhos apresentam formas de aplicar atributos de QoS em uma arquitetura orientada a serviços (SOA - Service Oriented Architecture) [Erradi and Maheshwari 2005, Kalepu et al. 2003, Menascé 2002]. Entretanto, nenhum dos trabalhos pesquisados até o momento, explorou a necessidade de mapear os atributos de QoS em relação a uma arquitetura orientada a serviços. Por exemplo, discutir para quais participantes da arquitetura esses atributos de QoS são relevantes e onde devem ser mensurados [Menascé 2004a]. Essa discussão é importante porque vai afetar decisões tomadas na implementação de aplicações de uma arquitetura orientada a serviços que tenha suporte à QoS.

Nesse contexto, o objetivo deste artigo é elicitar e discutir os atributos de QoS necessários para realizar a avaliação de desempenho de Web Services. Esses atributos de QoS serão mapeados para entidades participantes de uma arquitetura orientada a serviços denominada WSARCH (*Web Services Architecture*), esta em fase de implementação. O foco da arquitetura WSARCH é incorporar avaliação de desempenho e Qualidade de Serviço na tradicional arquitetura SOA.

Este artigo está organizado da seguinte forma. Na Seção 2 é dada uma breve introdução sobre Web Services e seus principais problemas de desempenho. Na Seção 3, uma arquitetura orientada a serviço baseada em QoS, em fase de implementação, é apresentada. Na Seção 4 são apresentados os atributos de QoS considerados na arquitetura proposta. Em seguida, esses atributos são relacionados com as entidades participantes da arquitetura. Na Seção 5, são apresentados os principais trabalhos relacionados. Por fim, na Seção 6 são apresentadas as conclusões deste artigo.

2. Web Services

Web Service pode ser definido como um componente, ou unidade lógica de aplicação, acessível através de protocolos padrões da Internet, possuindo uma funcionalidade que pode ser reutilizada sem a preocupação de como é implementada. A W3C define Web Service como uma aplicação identificada por uma URI - (*Uniform Resource Identifier*), cujas interfaces e ligações são definidas, descritas e descobertas utilizando-se como padrão a linguagem XML [Ferris and Farrell 2003].

Segundo [Kreger 2001], a tecnologia Web Services poderá ser tão importante para as interações entre aplicação-aplicação (B2B), quanto a Web foi para as interações en-

tre aplicação-usuário (B2C). Web Services possibilitam empresas reduzirem custos de *e-bussiness*, fornecer soluções mais rápidas e criar novas oportunidades de negócios.

Além disso, Web Services permitem que empresas desenvolvam funções que podem ser utilizadas sob demanda ou utilizadas em conjunto para prover um serviço de negócio [Chen et al. 2003b]. Web Services estão emergindo para fornecer uma infraestrutura sistemática e extensível para interações aplicação-aplicação, construída sobre protocolos Web já existentes e baseada em padrões XML abertos [Curbera et al. 2002]. A arquitetura adotada pelos WSs é mostrada na Figura 1.

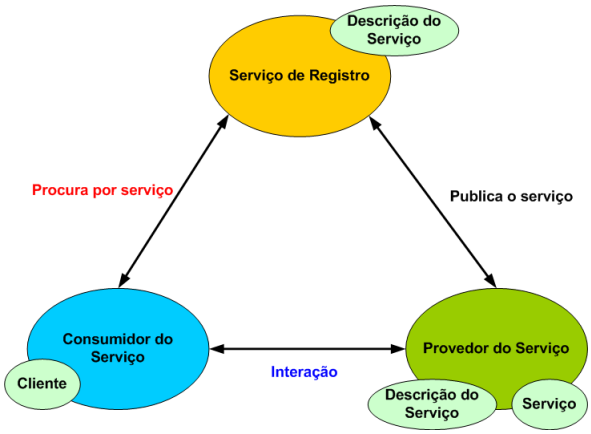

Figure 1. Arquitetura SOA e suas entidades

A arquitetura, mostrada na Figura 1, é formada por três entidades participantes: provedor de serviço, consumidor de serviço e registro de serviços. O **provedor de serviço** é o repositório do serviço. É função do provedor de serviço descrever de forma padronizada cada serviço criado e publicá-lo em registros de serviços. O **consumidor de serviço** ou cliente é a entidade que vai utilizar um serviço criado por um provedor de serviço. O cliente é capaz de utilizar o serviço por meio das informações encontradas na descrição realizada pelo provedor. A busca pelo serviço adequado é feita através de uma consulta a um registro de serviços. O **registro de serviços** é a entidade com o qual ambos, o provedor e o cliente, interagem. Os provedores publicam seus serviços no registro para que os clientes consigam encontrá-los e utilizá-los. O registro de serviços é essencialmente um repositório de dados baseado em XML.

Vale ressaltar que os Web Services são baseados em três padrões Web fundamentais: SOAP, WSDL e UDDI. O *Simple Object Access Protocol* (SOAP) [W3C 2003] é um protocolo de comunicação baseado em XML para a interação de aplicações. Nos Web Services, o SOAP é utilizado para definir a estrutura das mensagens trocadas entre as entidades participantes. A linguagem *Web Services Description Language* (WSDL) [W3C 2001] é um padrão da W3C utilizado para descrever as interfaces dos Web Services. O (UDDI) - *Universal, Discovery, Description and Integration* [OASIS 2005] é um padrão da OASIS[1] que fornece aos usuários uma forma unificada e sistemática para descobrir serviços por meio de um registro centralizado. Em resumo, um Web Service pode ser definido como um serviço de software publicado na Web, que troca mensagens

[1]Organization for the Advancement of Structured Information Standards - http://www.oasis-open.org

utilizando o protocolo SOAP, é descrito por um arquivo na WSDL e registrado em um repositório conhecido como UDDI.

Ainda que a interface WSDL permita a descrição de atributos funcionais dos Web Services [Garcia and de Toledo 2006], há uma carência em permitir que outros atributos não-funcionais e transacionais também sejam anexados. Permitir que atributos não-funcionais (tempo de resposta, vazão, segurança, etc) sejam anexados à WSDL seria uma possibilidade para melhorar a provisão de QoS em Web Services. Como um quesito essencial para a COS, QoS tem-se demonstrado um fator difícil de se gerenciar devido às diferenças entre as organizações e localização dos Web Services [Ludwig 2003]. Nesse sentido, QoS e Avaliação de desempenho para Web Services são assuntos pesquisados na atualidade, atraindo tanto pesquisadores da academia quanto da indústria.

2.1. Desempenho

Para permitir que sistemas troquem informações com um mínimo de qualidade, é primordial que questões de desempenho sejam levadas em consideração. No entanto, o desempenho esperado pelo cliente de um serviço nem sempre é alcançado. No universo dos Web Services, isso ocorre por falhas nas especificações de arquiteturas que focam mais em como o serviço pode ser acessado e não em questões de qualidade, isto é, o foco é mais voltado para aspectos funcionais do que para aspectos não-funcionais de serviços. A arquitetura SOA e suas extensões tem possibilitado a interação entre diversos sistemas, mas ainda carece de critérios mínimos de QoS. Diante de pesquisas efetuadas na literatura de Web Services e qualidade de serviço, foram identificadas lacunas que precisam ser corretamente abordadas para trazer benefícios para as áreas de pesquisa em questão e também para aqueles que queiram se beneficiar da arquitetura orientada a serviço como forma de integrar aplicações, equipamentos e processos de negócios. Os principais problemas de desempenho encontrados para WSs são [Mani and Nagarajan 2002]:

HTTP : O protocolo HTTP possui características que impedem que a transmissão das mensagens SOAP trocadas entre as aplicações seja feita de forma confiável [Mani and Nagarajan 2002]. Não há garantias das mensagens serem entregues ao destino e também que a ordem destas seja mantida. Assim, um modo para o fornecimento de qualidade de serviço em Web Services é a utilização de protocolos que realizam o transporte confiável das mensagens trocadas entre as aplicações envolvidas num processo de comunicação. Durante alguns anos, a IBM e HP trabalharam no desenvolvimento de uma extensão do protocolo HTTP denominada HTTPR [Banks et al. 2002].
Apesar da especificação do protocolo HTTPR, não há registros de uso deste pela indústria de TI. Isso fez com que IBM e HP parassem o projeto e iniciassem trabalhos em outra frente denominada *Web Services Reliable Messaging*. Trata-se de um especificação padronizada, especificando um protocolo que permita que mensagens sejam entregues de forma confiável entre aplicações distribuídas, independente da presença de falhas na rede e componentes de software utilizados [Bilorusets et al. 2007].

SOAP : Em geral, o protocolo SOAP degrada o desempenho na interação entre os Web Services, pois um processador XML deve ser carregado, instanciado e alimentado com dados XML [Mani and Nagarajan 2002]. Algumas inconsistências, como as listadas abaixo devem ser consideradas:

- A extração das informações da mensagem SOAP é custosa em relação ao tempo;
- Interpretar a informação em XML da mensagem SOAP utilizando um processador XML também tem um custo temporal;
- Falta de otimização com dados XML;
- Codificação de dados binários em forma aceitável para XML resulta em um *overhead* adicional de bytes, bem como o *overhead* de processamento;

Processadores XML : Outra questão importante, é o papel do processador XML no desempenho do protocolo SOAP. Como discutido anteriormente, processadores XML são custosos em termos de tamanho e tempo de processamento. Isso ocorre porque esses processadores precisam suportar checagem de tipo e conversão de dados, o que leva ao elevado uso de recursos computacionais.

WSDL : A interface que possibilita a descrição de um determinado WS para o mundo externo, também possui problemas que necessitam ser investigados. A WSDL se preocupa somente com aspectos funcionais e de sintaxe, mas não verifica se a semântica do documento está correta ou se critérios mínimos de QoS podem ser alcançados. Devido a ausência de tais características, o desenvolvedor necessita entender as características de QoS dos Web Services, enquanto desenvolvem aplicações que invocam WSs.

Canal de Comunicação : Um fator limitante no bom desempenho de Web Services é o canal de comunicação. É preciso destacar que os WSs atingiram um estágio atual de maturidade devido ao fato da comunicação entre os serviços serem feitas utilizando a Internet. No entanto, quando se fala na interligação de sistemas de missão crítica pela Internet, tais como bolsa de valores, centros empresariais, alguns detalhes como baixa latência, baixo congestionamento e garantia de entrega não podem deixar de ser considerados na transmissão das mensagens. Segundo [Mani and Nagarajan 2002], esses detalhes poderiam ser garantidos caso fosse construída uma rede mundial exclusiva para WSs.

3. Arquitetura WSARCH

Em complemento à arquitetura original de Web Services (SOA), o modelo de arquitetura WSARCH, tem como objetivo permitir melhor provisão de qualidade de serviço entre as mensagens trocadas pelas aplicações Web. É um arquitetura em fase de implementação, utilizando tecnologias e sistemas abertos como Linux e Java. O propósito da construção dessa arquitetura é caracterizar o desempenho de Web Services, iniciando em primeira instância, um estudo e aplicação dos atributos de QoS em cada uma das entidades participantes. O correto levantamento de atributos de qualidade de serviço faz-se necessário, pois aplicações distintas requerem qualidade de serviço de forma específica. Ou seja, cada aplicação possui um requisito de qualidade, que pode não ser adequado para a utilização de outra. Muitas têm sido as publicações sobre Web Services, mas poucas abordam em detalhes a importância de prover qualidade de serviço num ambiente altamente dinâmico como a Web. A arquitetura WSARCH, seus componentes e as possíveis interações são apresentados de modo simplificado na Figura 3.

A arquitetura WSARCH, assim como a arquitetura SOA, possui o provedor de serviço, o cliente e o registro de serviços. Além dessas entidades, a arquitetura WSARCH possui um *broker* de QoS. Esse *broker* deve funcionar como um centro de informações

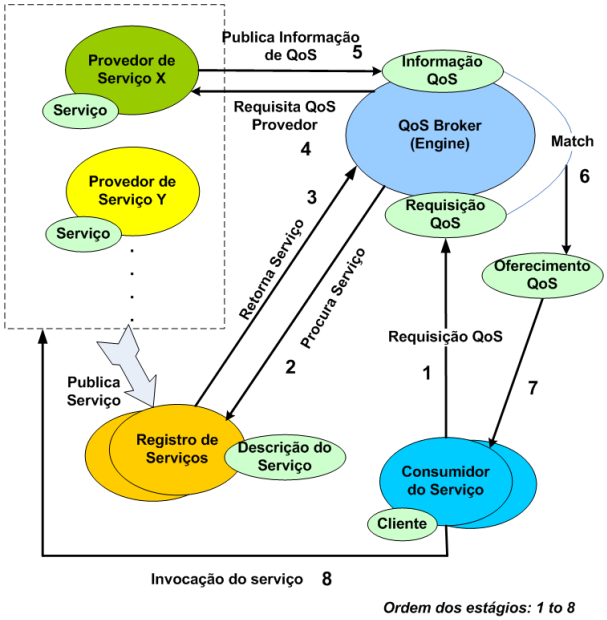

Figure 2. WSARCH - Modelo simplificado

sobre a qualidade de serviço das outras entidades participantes da arquitetura. Deve extrair informações de QoS das requisições enviadas por um cliente e utilizá-las para contactar um provedor de serviço que atenda as necessidades de qualidade do cliente. No estágio atual da arquitetura WSARCH, o *broker* apresenta os seguintes componentes:

- **Modelo de Desempenho**: Deve computar o tempo de resposta estimado para uma requisição de serviço de acordo com uma requisição de QoS;
- **Descritor de Recursos**: Receberá informações de QoS do provedor de serviços. Esse componente armazenará informações para serem utilizadas futuramente (detalhes sobre os recursos computacionais (processador, carga de CPU, memória, ocupação de disco, etc). Tais parâmetros podem ser utilizados/modificados de acordo com necessidades das diferentes aplicações;
- **Descritor de Preço/Custo**: A função deste componente é determinar o custo de uma requisição de serviço em função dos parâmetros presentes na requisição de serviço. De posse das informações enviadas pelo web service cliente e também das informações de desempenho de um possível provedor de serviço que venha efetuar o atendimento da requisição, o broker pode determinar se o processamento da requisição é compensatório.
  Por exemplo, deve gerar um modelo de preço para uma classe de aplicação.
- **Descritor de Desempenho**: Computar/Estimar o tempo a execução do serviço requisitado;
- **Gerenciador de Recursos**: Funcionará como uma interface para:
  - Obter informações atualizadas sobre a disponibilidade de recursos computacionais do provedor de serviços;
  - Gerar um descritor de recursos contendo detalhes de recursos privados alocados temporariamente e que devem ser utilizados como entrada para o modelo de preço.

- **Gerenciador de QoS**: Coleta de informações do gerenciador de recursos, do modelo de desempenho e do modelo de preço. O propósito é decidir se um contrato mínimo com o cliente será cumprido, de modo a gerar um oferecimento de QoS (que seria o serviço mais adequado para a solicitação inicial do cliente (aplicação), isto é, o que o provedor de serviços poderia oferecer naquele momento).

Devido a inclusão do *broker* na arquitetura original SOA e considerando a proposta da arquitetura WSARCH, o fluxo de mensagens entre as entidades participantes (cliente, *broker* e provedor de serviços) também é diferente. Como forma de entender o funcionamento da arquitetura WSARCH uma breve descrição de suas fases (seguindo a ordem dos estágios mostrada na Figura 2) é apresentada a seguir:

1. O cliente faz uma requisição ao *broker* de QoS em busca de um serviço. Além das informações comuns, a requisição contém informações sobre atributos de QoS que devem ser atendidos. Os atributos de QoS serão especificados pelo usuário no qual a uma política de atendimento no provedor de serviços deve ser associada.
2. Com base nas informações de QoS solicitadas pelo cliente do serviço, o *broker* busca em seu repositório de dados o serviço que melhor se adequa aos requisitos de QoS solicitados. Em seguida, o *broker* de QoS realiza uma busca no registro de serviços com o objetivo de encontrar o serviço mais adequado;
3. O registro de serviços retorna especificação do serviço apropriado;
4. Com a localização do serviço apropriado no repositório de serviços, o *broker* solicita ao provedor de serviços qual sua capacidade de atender a uma requisição (levando em consideração os parâmetros de QoS feitos pelo cliente). Se positivo as informações do provedor são retornadas ao *broker*;
5. Nesta fase o *broker* deve cruzar as informações solicitadas pelo cliente com aquelas disponíveis pelo provedor de serviços;
6. Se tais informações coincidirem (*match*), um oferecimento de QoS deve ser realizado;
7. O oferecimento de QoS é então retornado ao cliente. No entanto, esse oferecimento pode não ser aquele que o usuário solicitou. Nesse caso, uma QoS de valor próximo ao solicitado (melhor esforço) é apresentada ao cliente caso a solicitação inicial não possa ser atendida.
8. De posse da QoS adequada o cliente faz a invocação do serviço ao provedor adequado.

4. Relação de Atributos de QoS com a WSARCH

Em [Thio and Karunasekera 2005] é discutido como fazer medidas de QoS tanto do lado do cliente quanto do servidor. Qualidade de serviço de um WS é um fator importante que diferencia serviços similares oferecidos pelos vários provedores. Tal medida permitiria a um cliente de um WS escolher e ligar (se comunicar com) um WS adequado em tempo de execução, baseado em atributos de QoS.

Nesta seção, os atributos de QoS são relacionados com as entidades participantes da arquitetura WSARCH proposta. Inicialmente é discutida a perspectiva de qualidade de serviço para cada uma das entidades. Em seguida, os atributos de QoS que serão considerados na implementação da arquitetura são apresentados. Esses atributos são definidos e relacionados com as entidades participantes.

4.1. QoS e Entidades da WSARCH

Cada uma das partes envolvidas na arquitetura SOA e suas extensões possuem uma perspectiva distinta de qualidade de serviço. A seguir serão discutidas as perspectivas de QoS para cada uma das entidades da arquitetura proposta.

4.1.1. Cliente

Geralmente, para um cliente, o atributo primordial em termos de qualidade de serviço é o tempo de resposta. Isso significa que o único modo de um usuário aferir a qualidade de um serviço prestado por uma entidade é verificar se a resposta à sua solicitação foi atendida num tempo adequado. Quando se fala em comunicação entre aplicações independente de linguagens de programação, como é o caso dos *web services*, outros fatores que contribuem para o aumento ou diminuição do tempo de resposta precisam ser considerados: o processamento de mensagens XML, a possibilidade de compressão de mensagens visando diminuir o tempo de resposta, o tempo de acesso e recuperação de informação de uma base de dados, caso as mensagens trocadas entre as aplicações façam uso de um repositório de dados e atraso da rede.

4.1.2. Provedor

O provedor de serviços precisa ser eficiente ao disponibilizar um serviço requisitado pelo cliente. A seleção e a execução devem ser realizadas em tempo hábil de modo a evitar degradação no tempo de resposta. Por isso, é necessário considerar:

- As estimativas de tempo de execução de uma aplicação baseadas nas características da aplicação (enviadas através de uma requisição Web na forma de mensagens SOAP;
- A divisão das aplicações em classes, de modo a determinar um tratamento diferenciado no atendimento. Por exemplo, um serviço multimídia requer alta vazão (a quantidade de serviços que o provedor atende por segundo, por exemplo), enquanto um serviço bancário online requer segurança e qualidade de serviço transacional. As classes de serviços podem ser determinadas por meio de alguma parametrização, na qual uma vantagem em relação ao atendimento poderia ser concedida. Tal parametrização envolve a definição de uma política de atendimento associada à classe de serviço.
- Determinar se o serviço requisitado deve fazer parte de um processo de composição de serviços e se a composição será feita de forma centralizada, isto é, no próprio provedor. Um exemplo de composição ocorre quando um usuário precisa fazer a reserva de uma viagem via Web. Neste caso, ocorrerá a interação entre o sistema de reserva do hotel, a verificação da entidade certificadora do cartão de crédito e também a disponibilidade do vôo para o destino que o cliente solicitou. É importante salientar que esse processo, também conhecido como processo de negócios, somente finaliza se as entidades participantes trocarem dados de forma correta. Em resumo, tal processo de negócios também pode contribuir para um impacto significativo no tempo de resposta retornado ao cliente;
- O cache de requisições;

	Entidades da Arquitetura		
Atributos de QoS	Cliente	Provedor	Broker de QoS
Tempo de Resposta	X		
Vazão		X	
Memória Principal		X	
Compressão de Mensagens	X	X	X
Atraso da Rede	X	X	X
Composição de Serviços	X	X	
Disponibilidade		X	X
Custo do Serviço			X
Confiabilidade			X
Classe do Serviço		X	X
Carga do Servidor		X	X
Latência		X	
Reputação	X		X
Atomicidade		X	
Consistência		X	
Isolamento		X	
Durabilidade		X	

Table 1. Relação dos atributos de QoS com as entidades da WSARCH

- O balanceamento de carga.

4.1.3. Broker de QoS

O *broker* deve fazer o papel de intermediador entre o cliente e o provedor de serviços e interagir ativamente com diversos provedores em busca de serviços similares. Sua função é interceptar as requisições de um cliente em particular e tentar oferecer a este o melhor serviço disponível de acordo com a sua solicitação. O *broker* deve se preocupar com questões como:

- Gerenciar as solicitações feitas pelo cliente e iniciar a interação com os possíveis provedores na busca do melhor serviço;
- Estimativa de tempo para o cruzamento de informações entre o que o cliente deseja e o que o provedor de serviços pode oferecer;
- Interação com um repositório de serviços para realizar a seleção do serviço adequado. Isso deve envolver tempo de busca e tempo de acesso;
- Lista de provedores sobrecarregados e que apresentam Web Services que causam *overhead* de comunicação gerado pelo processamento de mensagens SOAP.

Na implementação da arquitetura WSARCH os atributos de QoS considerados são relacionados com as entidades participantes. Esse relacionamento é apresentada na Tabela 1.

4.2. Atributos de Qualidade de Serviço

Nesta seção serão discutidos os atributos de qualidade de serviço que devem ser mapeados para permitir que aplicações baseadas em WSs possam se comunicar com qualidade. Considera-se importante um estudo dessa natureza pelo fato de haver uma convergência de aplicações executando na Web. Nesse sentido, a qualidade do serviço oferecido aos clientes dos serviços devem ser abordadas com cuidado. Dessa forma, a interação entre QoS e as entidades cliente, provedor e *broker* da arquitetura WSARCH deve ser caracterizada. Inicialmente, os principais atributos de QoS considerados na arquitetura WSARCH são apresentados a seguir.

4.2.1. Atributos de QoS

- **Tempo de resposta** (Cliente): Muito importante quando se pretende avaliar qual o impacto visível pelo cliente em relação ao desempenho do sistema e também características da rede de comunicação. Esse tempo de resposta pode ser o tempo de resposta total, envolvendo rede de comunicação em conjunto com a aplicação, ou somente o tempo necessário para que determinado serviço seja finalizado;
- **Vazão** (Provedor): A capacidade que um determinado provedor de serviço apresentará em termos de serviços executados por unidade de tempo. Esse atributo depende de outros fatores, tais como a capacidade da processamento, de memória disponível, velocidade do disco, característica da aplicação, etc;
- **Memória principal** (Provedor): Outro componente que tem papel importante no tempo de execução de serviço e no tempo de resposta resultantes para as diferentes aplicações;
- **Compressão de mensagens** (Cliente, Provedor, Broker): Pretende-se realizar um estudo da técnica de compressão de mensagens [Wu et al. 2003] de modo que o tempo de transmissão das mensagens e o tempo de resposta trocadas entre as entidades comunicantes sejam reduzidos.
- **Atraso da rede** (Cliente, Provedor, Broker): O tempo de transmissão requerido para receber o serviço. Componente importante para serviços como conteúdo multimídia tal como vídeo ou gráficos. O atributo largura de banda é também essencialmente importante para o *broker* porque irá decidir se um serviço deve ser invocado, se um cliente está utilizando uma largura de banda inferior, etc.
- **Composição de serviços** (Cliente, Provedor): Em função da natureza dinâmica dos WSs e de inúmeros serviços que dependam um do outro para compor um serviço maior, é importante caracterizar e aplicar técnicas para seleção dos serviços mais apropriados baseados em requisitos de QoS;
- **Disponibilidade** (Provedor, Broker): A indisponibilidade de um serviço pode conduzir a problemas durante a composição de serviços. Por exemplo, para finalizar um serviço composto, com a melhor QoS possível, é necessário que determinado serviço esteja disponível. A indisponibilidade pode ser tanto de software, quanto de hardware. Para melhorar a disponibilidade de serviços em software é necessário a utilização de algoritmos de otimização de rotas. Em relação a indisponibilidade de hardware, esse problema pode ser amenizado utilizando técnicas como Linux-HA (*Linux High Availability*). Linux-HA provê capacidade básica de alta disponibilidade (*failover*) em uma gama grande de plataformas, suportando milhares de websites em missão crítica no mundo;
- **Custo do serviço** (Broker): A análise do custo é um quesito fundamental para verificar o quão viável será determinar uma qualidade de serviço para uma aplicação em particular. Os WSs possuem características diferentes e o custo para a computação de cada serviço irá depender dessas características;
- **Confiabilidade** (Broker): Seleção de um serviço que melhor se adequa as necessidades da aplicação;
- **Classe de serviço** (Provedor, Broker): Uma classe de serviço é uma coleção de WSs individuais com funcionalidades comuns, mas com diferentes propriedades não-funcionais;

- **Carga do servidor** (Provedor, Broker): Quantidade de serviços presente no provedor e quantidade de requisições simultâneas para cada tipo de serviço.
- **Latência** (Provedor): Tempo gasto entre a chegada da requisição e o envio da resposta.
- **Reputação** (Cliente, Broker): Medida de confiabilidade do Web Service que pode ser calculada pela experiência dos clientes.

4.2.2. Atributos Transacionais de QoS

Qualidade de serviço transacional refere-se ao nível de confiabilidade e consistência em que as transações são executadas. Segundo [Mani and Nagarajan 2002] são fundamentais para manter a integridade de um Web Service.

- **Atomicidade** (Provedor): Uma transação é atômica. Isso significa que na situação que tal característica é exigida, duas fases separadas ocorrem: ou a transação é permitida ou ela não é permitida. Não é possível a ocorrência das duas fases ao mesmo tempo;
- **Consistência** (Provedor): Uma execução correta da transação deve levar o sistema de um estado consistente a outro;
- **Isolamento** (Provedor): Uma transação não deveria deixar sua utilização visível para outras transações até que ela realmente seja executada. Ou seja, a transação deve executar como se não soubesse da ocorrência de outras;
- **Durabilidade** (Provedor): Uma vez que uma transação finaliza, as mudanças finalizadas nunca devem ser perdidas no evento de qualquer falha.

4.3. Avaliação de Desempenho

O projeto e desenvolvimento da arquitetura *WSARCH* está sendo desenvolvido com foco em avaliação de desempenho. Desta forma, estão sendo definidos procedimentos para avaliar o desempenho da diversidade de interações que podem ocorrer entre os componentes da arquitetura proposta. Para isso, alguns parâmetros de qualidade de serviço e fatores devem ser considerados nas interações.

Devido a natureza dinâmica da *Internet* e o surgimento cada vez maior de diferentes aplicações disponíveis na *web* é preciso um estudo que defina quais atributos analisar e em que situação essa análise deve ser feita. O critério para a escolha dos atributos de *QoS* devem ser baseados na característica das aplicações. Por isso, é importante detalhar o comportamento (isto é, identificar as necessidade de *QoS*) dessas aplicações e descobrir que parâmetros são mais adequados e como eles serão correlacionados. Por exemplo, um serviço multimídia requer alta vazão, enquanto um serviço bancário requer segurança e qualidade de serviço transacional.

A seguir foram levantados alguns fatores que estão sendo considerados nas interações entre os diferentes módulos e sub-módulos para avaliar o desempenho da arquitetura *WSARCH*.

Interação Cliente x Broker

- Quais o parâmetros de QoS enviados ao *broker* devem ser considerados?

- Pode ser pensado uma avaliação baseada em características de determinadas aplicações, cada uma com requisitos de QoS diferentes;
- Qual o preço dessa interação?

Interação Broker x Registro de Serviços

- Considerar o tempo gasto para o *broker* fazer uma solicitação ao registro de serviços e também o tempo que o registro de serviços levará para encontrar o serviço mais adequado (com base nas informações de *QoS* do cliente).
- No caso da busca no registro de serviços, que algoritmos / heurísticas considerar para diminuir o tempo de resposta global?

Interação Broker x Provedor de Serviços

- Tempo necessário para o provedor de serviços selecionar o serviço requisitado e retornar tal informação para o *broker*;
- Considerar o tempo de resposta do provedor de serviços de acordo com a solicitação feita pelo *broker*;
- No *broker* deve ser considerado o tempo para fazer o "casamento" entre a *QoS* solicitada pelo cliente e *QoS* retornada pelo provedor do serviço;
- O provedor também pode trabalhar com uma *cache* de requisições e balanceamento de carga, o que faria com que o tempo para a consulta aos serviços mais acessados diminuísse. O provedor também pode:
  - Categorizar o tráfego do *web service* pelo volume de tráfego;
  - Categorizar o *web service* pelo tráfego de diferentes categorias;
  - Categorizar pelo tráfego de diferentes fontes.
- Retorno da resposta ao cliente.

Interação Cliente x Provedor de Serviços

- Tempo de invocação do serviço pelo cliente;
- Tempo para o processamento do *web service* no provedor;
- Tempo de resposta.

5. Trabalhos Relacionados

As pesquisas com *QoS* em Web Services têm sido necessárias uma vez que uma aplicação na Internet pode invocar muitos serviços - um WS de comércio eletrônico, por exemplo, poderia invocar um serviço de pagamento, que poderia então invocar um serviço de autenticação [Menascé and Woodside 2006]. Considerar especificações de qualidade de serviço é um problema complexo, expondo muito desafios que necessitam ser abordados [Menascé 2004a]. Métricas de QoS para selecionar Web Services e provedores de serviços também têm sido proposta em [Kalepu et al. 2003]. Para medir como o provedor de serviços está em conformidade com os acordos de níveis de serviços SLA (*Service Level Agreements*), os autores tentam quantificar a consistência em níveis de aceitação e introduzem um novo atributo de QoS denominado *verity*, propondo uma arquitetura para quantificá-lo. O ponto negativo é que o artigo não mostra resultados sobre avaliação de desempenho, somente indicando uma nova medida de qualidade de serviço.

Em [Yeom and Min 2005] é apresentado um Web Service *broker* que monitora ativamente a QoS (disponibilidade, desempenho e confiabilidade) de Web Services. Com

esta informação, um usuário pode selecionar um Web Service que melhor se adequa as suas necessidades. Destaca-se também o trabalho de [Chen et al. 2003a] em que é apresentada uma arquitetura utilizando um *broker* entre clientes e provedores de serviços e também as garantias (disponibilidade, latência, confiabilidade) que as aplicações necessitam quando estão trocando mensagens.

Outra abordagem para melhorar QoS em Web Services é apresentada em [Erradi and Maheshwari 2005]. Neste trabalho é proposta uma técnica baseada em um barramento de mensagens denominada wsBUS (*Web Services Bus*), ou seja, um *middleware* orientado a serviço para a interação de WSs confiáveis e tolerante a falhas. A proposta dessa arquitetura é bastante interessante, mas por outro lado o artigo não apresenta nenhuma proposta de avaliação de desempenho e validação.

Segundo [Garcia and de Toledo 2006] há uma lacuna em mecanismos para provisão de qualidade de serviço para Web Services. Não basta somente o uso de aspectos funcionais presentes na WSDL como forma de escolher o melhor serviço para uma aplicação específica. Os requisitos dos consumidores do serviço devem incluir tanto aspectos funcionais e não funcionais. Os autores sinalizam para uma abordagem que extende o framework WS-Policy (*Web Services Policy*), padrão para complementar as descrições da WSDL utilizando web semântica (OWL) e uma linguagem para incluir QoS no UDDI denominada ABLE Rule Language (ARL).

Assim, nenhum dos trabalhos apresentados relacionou atributos de QoS com as entidades da arquitetura que foi proposta. Isso contraria uma recomendação de [Menascé 2004a] que destaca a importância de saber quais os atributos a serem medidos e onde serem medidos.

Outro assunto que envolve QoS é a composição automática de WSs. A composição automática e eficiente de serviços é outro problema ao utilizar Web Services em aplicações de negócios [Menascé 2004b],[Jaeger et al. 2005], [Zeng et al. 2003], [Liu et al. 2005]. Em [Jaeger and Ladner 2005] são gerados exemplos de composições e candidatos de serviços através de uma ferramenta de simulação, com o objetivo de avaliar possíveis substituições na composição de serviços. A natureza altamente dinâmica e distribuída dos Web Services freqüentemente torna os provedores de serviços sobrecarregados em determinadas ocasiões. Baseado em teoria das filas, [Wang et al. 2004] apresenta uma estratégia de seleção e execução de serviço para prover garantias de QoS para provedores de serviços com recursos limitados.

Benefícios da computação orientada a serviços também podem ser aplicados em um ambiente de *grids* por meio dos chamados *Grids Services* [Atkinson et al. 2005]. Em [Benkner and Engelbrecht 2006] é abordada uma infraestrutura genérica de *grids* baseada em padrões de Web Services, que provê um suporte flexível de QoS no nível de aplicação, determinando os requisitos especiais de tempo de aplicações críticas em um ambiente de *Grid*. Em [Surridge and Taylor 2005], os autores apresentam o projeto *GRIA*, um *middleware* baseado em Web Services construído para satisfazer as necessidades da indústria tanto em relação às questões de segurança quanto à procura e operações de serviços orientados a negócios.

6. Conclusões

O surgimento cada vez maior de diferentes aplicações disponíveis na Web necessita de um estudo que defina quais atributos analisar e em que situação essa análise deve ser feita. Os critérios para a escolha dos atributos de QoS devem ser baseados nas características das aplicações. Por isso, é importante detalhar o comportamento, isto é, identificar as necessidade de QoS, dessas aplicações e descobrir quais parâmetros são mais adequados e como eles serão correlacionados.

Este artigo descreveu com base em trabalhos da literatura da área [Mani and Nagarajan 2002, Garcia and de Toledo 2006, Thio and Karunasekera 2005, Kalepu et al. 2004] os principais atributos de QoS para Web Services. Detalhou-se cada um deles e também em qual das entidades cada um dos atributos pode ser aplicado para a melhor provisão de qualidade de serviço para as aplicações. O levantamento dos atributos de QoS é uma das fases da implementação da arquitetura *WSARCH*, também discutida no presente artigo. A arquitetura está sendo implementada utilizando uma *engine* (motor) de processamento de mensagens *SOAP* denominada *Apache Axis2* [Axis2 2008], juntamente com servidores de aplicação (*Tomcat*) e barramentos de serviços (*Enterprise Service Bus-ESB*). O desenvolvimento do *broker* está sendo realizado com base no *Apache Synapse* [Synapse 2008]. O *Synapse* funciona como proxy/roteador de web services sobre HTTP, fazendo balanceamento de carga, roteamento, transformação e até mesmo a troca de protocolo.

As contribuições esperadas com o desenvolvimento deste trabalho e do levantamento de atributos de QoS irão permitir melhor caracterização da qualidade de serviço para a comunicação entre aplicações utilizando a arquitetura orientada a serviços mencionada anteriormente. Uma forma de análise de desempenho dessa arquitetura é necessária, uma vez que há uma lacuna em termos de abordagem deste tópico na literatura da área. Como objetivo principal, este trabalho tem explorado mais detalhadamente uma forma de provisão de QoS para os Web Services, utilizando técnicas já existentes, além da proposição de novas metodologias com a implementação da arquitetura descrita neste artigo. Dentre os beneficiados com este estudo, destacam-se bolsa de valores, provedores de serviços e de conteúdo, empresas e indústrias que queiram se beneficiar da computação orientada a serviços como forma de integrar aplicações, equipamentos e processos de negócios.

References

Atkinson, M., DeRoure, D., Dunlop, A., Fox, G., Henderson, P., Hey, T., Paton, N., Newhouse, S., Parastatidis, S., A. Trefethen, P. W., and Webber, J. (2005). Web service grids: an evolutionary approach, concurrency and computation: Practice an experiences. *Published by Wiley InterScience*.

Axis2, A. (2008). Apache axis2 - apache software foundation.

Banks, A., Challenger, J., Clarke, P., Davis, D., King, R. P., Witting, K., Donoho, A., Holloway, T., Ibbotson, J., and Todd, S. (2002). Httpr specification.

Benkner, S. and Engelbrecht, G. (2006). A generic qos infraestructure for grid web services. In *Proceeding of the Advanced International Conference on Telecomunications and International Conference on Internet and Web Applications and Services (AICT/ICIW 2006)*. IEEE CS Press.

Bilorusets, R., Box, D., Cabrera, L. F., Davis, D., Ferguson, D., Ferris, C., Freund, T., Hondo, M. A., Ibbotson, J., Jin, L., Kaler, C., Langworthy, D., Lewis, A., Limprecht, R., Lucco, S., Mullen, D., Nadalin, A., Nottingham, M., Orchard, D., Roots, J., Samdarshi, S., Shewchuk, J., and Storey, T. (2007). Web services reliable messaging.

Chen, H., Yu, T., and Lin, K.-J. (2003a). Qcws: An implementation of qos-capable multimedia web services. In *Proceedings of the Fifth IEEE International Symposium on Multimedia Software Engineering (ISMSE´03)*. IEEE CS Press.

Chen, M., Chen, A. N. K., and Shao, B. B. M. (2003b). The implications and impacts of web services to eletronic commerce research and pratices. *Journal of Electronic Commerce Research*, 4(4):128–139.

Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., and Weerawarana, S. (2002). Unraveling the web services: an introduction to soap, wsdl, and uddi. *Internet Computing*, 6(2):86–93.

Erradi, A. and Maheshwari, O. (2005). A broker-based approach for improving web services reliability. In *IEEE International Conference on Web Services (ICWS'05)*.

Ferris, C. and Farrell, J. (2003). What are Web services? *Communications of the ACM*, 46(6):31–31.

Garcia, D. Z. G. and de Toledo, M. B. F. (2006). Semantics-enriched qos policies for web service interactions. In *12th Brazilian Symposium on Multimedia and the Web, WebMedia 2006*, pages 35–44.

Jaeger, M. C. and Ladner, H. (2005). Improving the qos of ws compositions based on redundant services. In *Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP´05)*. IEEE CS Press.

Jaeger, M. C., Muhl, G., and Golze, S. (2005). Qos-aware compostion of web services: A look at selection algorithms. In *Proceedings of the IEEE International Conference on Web Services (ICWS´05)*. IEEE CS Press.

Kalepu, S., Krishnaswamy, S., and Loke, S. W. (2003). Verity: A qos metric for selecting web services and providers. In *Proceedings of the Fourth International Conference on Web Information Systems Engineering Workshops (WISEW´03)*. IEEE CS Press.

Kalepu, S., Krishnaswamy, S., and Loke, S. W. (2004). Reputation = f(user ranking, compliance, verity). In *ICWS '04: Proceedings of the IEEE International Conference on Web Services (ICWS'04)*, page 200, Washington, DC, USA. IEEE Computer Society.

Kreger, H. (2001). Web services conceptual architecture (wsca 1.0). *IBM*. Disponível em: http://www-306.ibm.com/software/solutions/webservices /pdf/WSCA.pdf. Último acesso: 20/07/2007.

Liu, J., Gu, N., Zong, Y., Ding, Z., Zhang, S., and Zhang, Q. (2005). Web services automatic composition based on qos. In *Proceedings of the 2005 IEEE International Conference on e-Business Engineering (ICEBE´05)*. IEEE CS Press.

Ludwig, H. (2003). Web services qos: External slas and internal policies or: How do we deliver what we promise. In *Proceedings of Fourth International Conference on Web Information Systems Engineering Workshops (WISEW'03) - 2003*.

Mani, A. and Nagarajan, A. (2002). Understanding quality of service for web services.

Menascé, D. (2002). Qos issues in web services. *IEEE Internet Computing*, 6(6):72–75.

Menascé, D. (2004a). Composing web services: A qos view. *Internet Computing*, 8(6):88–90.

Menascé, D. (2004b). Response time analysis of composite web services. *IEEE Internet Computing*, 8(1):90–92.

Menascé, D. and Woodside, M. (2006). Application qos level. *IEEE Internet Computing*, 10(3):13–15.

OASIS (2005). Uddi specifications tc. Disponível em: http://www.oasis-open.org/committees/uddi-spec/doc/tcspecs.htm. Último acesso: 20/07/2007.

Papazoglou, M. P. and Georgakopoulos, D. (2003). Service-oriented computing. *Communications of the ACM*, 46(10):24–28.

Surridge, M. and Taylor, S. (2005). Experiences with gria - industrial applications on a web services grid. In *Proceedings of the First International Conference on e-Science and Grid Computing (e-Science´05)*. IEEE CS Press.

Synapse, A. (2008). Apache synapse enterprise service bus - apache software foundation.

Thio, N. and Karunasekera, S. (2005). Automatic measurement of a qos metric for web service recomendation. In *Proceedings of the 2005 Australian Software Engineering Conference (ASWEC'05)*.

W3C (2001). Web services description language (wsdl). Disponível em: http://www.w3.org/TR/wsdl. Último acesso: 20/07/2007.

W3C (2002). Web services activity. Disponível em: http://www.w3.org/2002/ws/. Último acesso: 20/07/2007.

W3C (2003). Soap specifications. Disponível em: http://www.w3.org/TR/soap/. Último acesso: 20/07/2007.

Wang, X., Yue, K., Huang, J. Z., and Zhou, A. (2004). Service selection in dynamic demand-driven web services. In *Proceedings of IEEE International Conference on Web Services (ICWS´04)*. IEEE CS Press.

Wu, C.-H., Su, D.-C., Chang, J., Wei, C.-C., Lin, K.-J., and Ho, J.-M. (2003). The design and implementation of intelligent transportation web services. In *IEEE International Conference on E-Commerce Technology (CEC'03)*. IEEE CS Press.

X.902, I. (2007). Quality of service.

Yeom, G. and Min, D. (2005). Design and implementation of web services qos broker. In *Proceedings of the International Conference on Next Generation Web Services Practices (NWeSP´05)*. IEEE CS Press.

Zeng, L., Benatallah, B., and Dumas, M. (2003). Quality driven web services composition. *ACM Computing*.