SUPERVISÃO DE PROCESSOS INDUSTRIAIS USANDO WEB SERVICE

Alessandro J. de Souza*, Francisco Sales de L. Filho*, Adauto L. T. B. da Fonseca*,
Felipe C. Alves do Couto†, Rodrigo P. R. dos Santos†, Luiz Affonso Guedes†

* *Centro Federal de Educação Tecnológica do Rio Grande do Norte*
Tecnologia em Desenvolvimento de Software
Natal - Brasil

† *Universidade Federal do Rio Grande Norte*
Departamento de Engenharia de Computação e Automação
Natal - Brasil

Emails: ajdsouza@dca.ufrn.br, salesfilho@cefetrn.br, adauto6@netscape.net,
lipecouto@gmail.com, rodrigop@dca.ufrn.br, affonso@dca.ufrn.br

Abstract— Nowadays, the great challenge in industrial automation area is the integration of the diverse existing tecnologies. In this context, this paper presenter an arquitecture oriented services to provide interoperability communication among the components that compose an industrial automation systems. To validate the proposed architecture, it is presented an experiment of real time monitoring.

Keywords— WEB SERVICE, SOAP, JNI, WSDL, UDDI, INDUSTRIAL AUTOMATION.

Resumo— Nos dias de hoje, o grande desafio em um ambiente industrial é, sem dúvida, a possibilidade de integração das diversas tecnologias existentes em uma planta de automação industrial. Neste contexto, este artigo apresenta uma arquitetura orientada a serviços com o intuito de resolver os problemas de interoperabilidade entre os diversos componetes de um processo de automação. Para validar a arquitetura proposta, é apresentada uma experiência de monitoramento de uma planta em tempo-real.

Palavras-chave— WEB SERVICE, SOAP, JNI, WSDL, UDDI, INDUSTRIAL AUTOMATION.

1 INTRODUÇÃO

Os processos de automação industrial ganharam força ao longo dos anos. O aumento da produtividade e a qualidade dos produtos refletem esta evolução. O problema é que com esse aumento, tão rápido, dos processos de automação foram criados inúmeros equipamentos, os quais cada um possuindo sua forma específica de comunicação. A necessidade de padronização de comunicação entre os diversos dispositivos tornou-se, então, um desafio. Os diversos dispositivos (sensores, atuadores) ligados por uma rede industrial trouxeram à tona a realidade das aplicações distribuídas na área de automação industrial.

Soluções de software baseados em uma arquitetura de objetos distribuídos como *Distributed Component Object Model-DCOM*, *Remote Method Invocation - RMI*, e *Common Object Request Broker Architecture-CORBA* foram propostas e aceitas pela comunidade industrial e são usados até os dias atuais. Aprimorar a comunicação entre dispositivos industriais é um desafio que ainda cerca os profissionais da área de automação.

Neste contexto, este artigo propõe o uso de uma arquitetura orientada a serviços como uma alternativa aos padrões atuais, fornecendo características como: interoperabilidade, flexibilidade, portabilidade e escalabilidade.

Este artigo foi escrito de forma a apresentar conceitos sobre Comunicação em Ambiente Industrial, *Web Service* e suas tecnologias básicas, como pode ser visto nas seções de 2 e 3, e expor os resultados alcançados conforme descrito na seção 4. Para a obtenção dos resultados foi realizado um estudo de caso utilizando um *Web Service* com a aplicação escrita em linguagem *Java* para gerenciar o CLP SIMATIC S7-200 da SIEMENS.

2 COMUNICAÇÃO EM AMBIENTE INDUSTRIAL

A comunicação de dados em ambiente industrial despertou como uma necessidade desde o início da automação de processos, onde havia a necessidade de transportar os sinais provenientes dos instrumentos até os controladores. Desde então, foi possível observar a evolução desses meios de transporte como: transmissão pneumática, hidráulica, elétrica, ótica e via rádio.

Na década de 60 foi possível observar o surgimento da comunicação distribuída na automação com o surgimento dos *Distributed Control System - DCS*. Estes tinham a incumbência de distribuir suas funções de controle garantindo que se uma estação de controle falhasse, somente parte do processos pararia, possibilitando que o restante do processo continuasse em operação.

A utilização das redes de comunicação trouxeram grandes avanços para a automação de processos, porém um grande obstáculo para a comunicação inter-processos surgiu com a falta de padronização dos protocolos de comunicação.

A partir do surgimento do modelo OSI-ISO,

os fabricantes de equipamentos passaram a utilizar seus *Drivers* de comunicação como camada de aplicação, fazendo com que somente atráves destes fosse possível acessar os dados existentes em seus equipamentos. Este tipo de estratégia trouxe grandes dificuldades para a comunicação entre diferentes fornecedores de equipamentos.

Com o objetivo de eliminar as barreiras de comunicação impostas pela falta de padronização entre fabricantes, foi criado em 1996 a *OPC Foundation* (OPC, 2006). Esta organização procurou estabelecer regras de comunicação entre aplicações, encapsulando *Drivers* de fabricantes e oferecendo interfaces padronizadas de comunicação, Figura 1. Para esta padronização foi dado o nome de *OLE for Process Control*, e tem sua implementação centrada na tecnologia *Distributed Component Object Model - DCOM* para objetos distribuidos da Microsoft. O padrão OPC vem evoluindo ao longo do tempo, trazendo padrões abertos e independência de plataforma, como é possivel observar através da especificação OPC XML.(Celso, 2004)

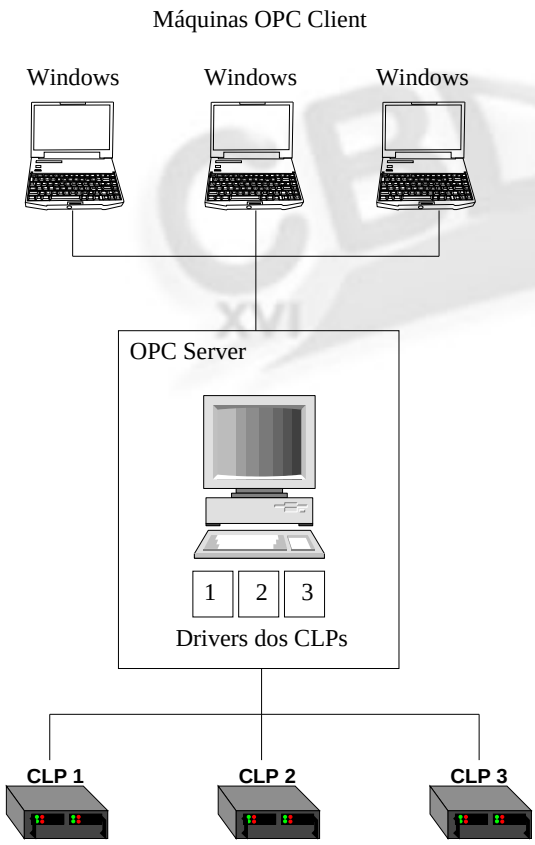

Figura 1: Comunicação usando o padrão OPC.

Apesar da padronização oferecida pelo OPC, ainda é possivel observar que o uso desta tecnologia prende-se a uma determinada plataforma dificultando a portabilidade dos sistemas atuais e limitando-se a esta. Há algum tempo, pesquisadores da área vêm pensando em alternativas para integração de processos de automação que possam cobrir as deficiências existentes nos sistemas atuais. A mudança de paradigma de tecnologias orientadas a objeto para tecnologias orientadas a serviços vêm se mostrando uma alternativa promissora.

Desta forma podemos concluir esta seção dizendo que os meios de transmissão dentro da automoção industrial constituem-se em um canal de informação extremamente rico entre o chão-de-fábrica e os sistemas de controle, sendo assim objeto permanente de avaliação e melhorias.

3 WEB SERVICE

A integração entre aplicações corporativas tem sido objeto de pesquisas constantes entre pesquisadores acadêmicos e industriais. Diversas tecnologias surgiram com o propósito de solucionar os problemas relacionados com a integração entre aplicações corporativas, entre elas podemos citar (Yi Zhi, 2005): CORBA, DCOM, JAVA RMI entre outros. Cada tecnologia possui um conjunto de protocolos e serviços de Middleware capaz de implementar um núcleo de funcionalidades requeridas, Tabela 1.

Tabela 1: Comparação de Tecnologias.

	DCE	**CORBA**	**JAVA RMI**
Formato dos Dados	NDR (Network Data Representation)	CDR (Common Data Representation)	Objetos Java Serializado
Formato de Envio	PDU (Protocol Data Units)	GIOP (General Inter-ORB Protocol)	Stream
Interfaces	DCE IDL (Interface Definition Language)	CORBA IDL (Inerface Definition Language)	Interfaces Java
Protocolo	RPC CO (Connect Oriented Protocol)	IIOP (Internet Inter-ORB protocolo)	JRMP
Forma de Divulgação	DCS (Cell Directory Service)	COS Naming (CORBA Object Service)	Java Registry
Invocação de Métodos	RPC (Remote Procedure Call)	CORBA RMI (Remote Method Invocation)	Java RMI (Remote Method Invocation)

Porém, a chegada da Arquitetura Orientada a Serviços -*Service-Oriented Architecture SOA* - reascendeu as discussões sobre integração dos sistemas corporativos. Nos últimos anos, as companhias têm usado SOA na integração de aplicações corporativas, obtendo o retorno do investimento através dos bons resultados alcançados (Vinoski., 2005). Neste contexto, os *Web Service* impulcionaram este tipo de arquitetura, mesmo que no início estivessem sendo considerados como mais uma encarnação da tecnologia de objetos distribuídos (Vogels., 2003).

Os *Web Services* são uma tecnologia cada vez mais utilizada, podendo ocupar lacunas deixadas pela rapidez da evolução tecnológica existentes, tanto no seguimento comercial como na

indústria, onde os parques tecnológicos são altamente heterogênios e sofrem freqüentes mudanças evolutivas.(François and Smit, 2005)

Para o grupo de trabalho do consórcio *World Wide Web - W3C*, a definição de *Web Service* é: "*Um sistema de software projetado para suportar interoperabilidade entre computadores em rede usando, tipicamente, o protocolo HTTP com serialização de documentos XML.*"(W3C, 2004)

Há três tecnologias básicas em torno dos *Web Services*:

- *Simple object Access Protocol -SOAP*
- *Web Service Description Language - WSDL*
- *Universal Description, Discovery and Integration - UDDI*

A Figura 2 ilustra como é desencadeado o processo de publicação e acesso a um *Web Service* por um cliente. O processo pode ser resumido da seguinte forma:

1. **Registro:** Este é o primeiro passo para que um cliente possa ter acesso às funcionalidades do *Web Service*, ou seja, o provedor de serviços deve registrar seus serviços junto a um registro UDDI.
2. **Pesquisa:** O cliente, então, pesquisa o serviço no registro UDDI.
3. **Descrição:** O cliente obtém uma descrição das variáveis métodos e tipos de parâmetros do provedor, isto é feito através do WSDL que identifica quais serviços estão disponíveis no *Web Service*.
4. **Comunicação:** O cliente, então, invoca o *Web Service* utilizando SOAP, que funciona com uma gramática para os XML existentes nas requisições e respostas entre usuário e *Web Service*.

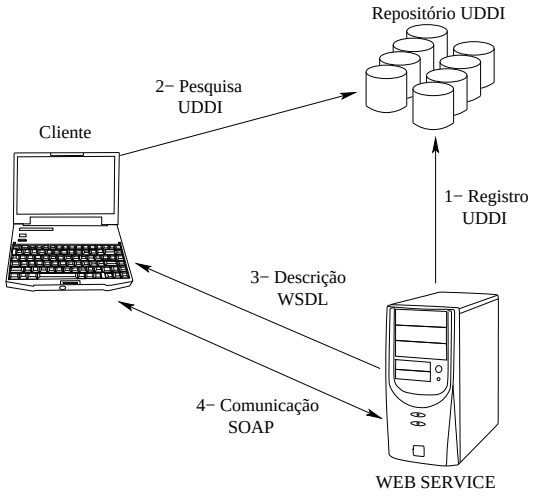

Figura 2: Tecnologias que compõe os *Web Service*.

3.1 SOAP

SOAP é um dos elementos fundamentais dos *Web Services*, pois compõe a base da pilha de protocolos de comunicação. Diferentemente dos protocolos binários utilizados por tecnologias como CORBA, COM e DCOM, o SOAP utiliza XML para codificar as mensagens, agregando mais força e legibilidade, tendo em vista a forte utilização de XML em aplicações distribuídas, principalmente baseados na Web (Szolkowski, 2003). O protocolo SOAP está dividido em três elementos básicos:

- Envelope: É o elemento principal do XML que representa a mensagem.
- Header: É um mecanismo genérico de adição de características à mensagem SOAP de maneira descentralizada, sem acordo anterior entre as partes comunicantes.
- Body: Contém a codificação atual de uma chamada a um método e todos os argumentos de entrada, ou uma resposta codificada que contém o resultado de uma chamada a um método.

Como o SOAP utiliza o XML para suas definições, qualquer tipo de dado pode ser representado, desde que ele possa ser especificado em algum XML Schema.

3.2 WSDL

WSDL é uma linguagem de descrição de serviço baseada em XML que tem como função, exibir os métodos disponíveis, suas localizações e os protocolos que um *Web Service* oferece. A idéia é similar a uma interface java, ou a uma *Interface Definition Language*-IDL CORBA. Desta forma, WSDL oferece independência de linguagem e componentes, não importando que tipo de tecnologia está sendo utilizada (Francisco, 2002)

3.3 UDDI

UDDI é o mecanismo que oferece uma sistemática unificada para busca de provedores de serviços através de um repositório centralizado de registros. Fazendo uma analogia com a busca de um *site Web*, o UDDI equivaleria a um serviço de busca tipo *Google*, *Yahoo* entre outros. Desta forma, os provedores de serviço registram-se junto ao UDDI *Registry* e assim os clientes podem localizar os serviços consultanto o repositório.

Os registros UDDI, permitem localizar negócios e serviços de três formas:

- *"Yellow Pages"* - Localização de negócio baseado em tipos de serviços.
- *"While Pages"* - Localização de negócios por nome e detalhes do contato.

- *"Green pages"* - Localizam informações sobre os serviços que um negócio oferece.

Assim, podemos encerrar esta seção concluindo que os sistemas baseados em uma arquitetura orientada a serviços possuem como característica um fraco acoplamento e ligações dinâmicas entre seus módulos, diferentemente do pradigma orientado a objetos, que possui forte acoplamento e ligações estáticas entre seus módulos. Um exemplo desta arquitetura é o OPC, visto na seção anterior.

A possibilidade de encapsulamento dos *Drivers* dos fabricantes por um *Web Service*, como ilustrado na figura 3, permitirá que clientes tenham acesso às informações dos CLPs agregando as seguintes vantagens:

- Mobilidade de serviços
- Suporte automático a vários tipos de clientes
- Maior reuso
- Maior escalabilidade
- Maior disponibilidade

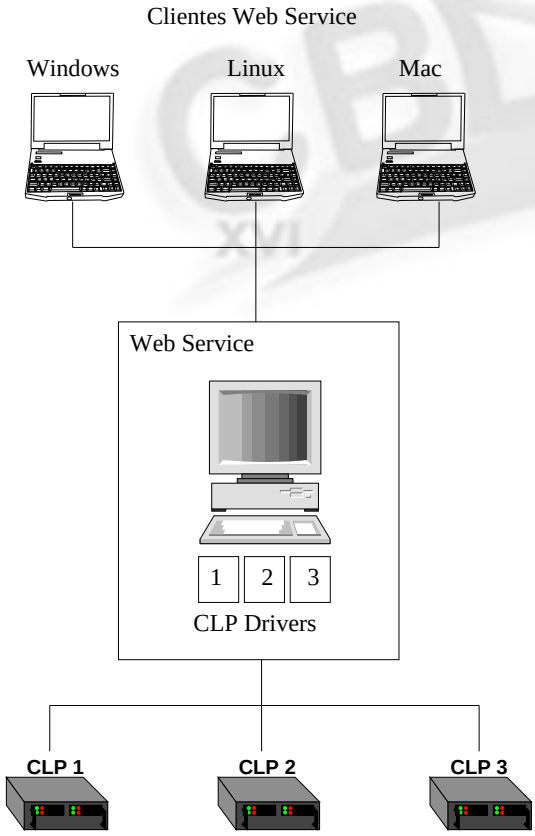

Figura 3: *Web Service* encapsulando *Drivers* de CLP

4 RESULTADOS

Dentro dos modelos de comunicação de dados explanados nas seções anteriores, este trabalho implementa uma nova abordagem da comunicação utilizando *Web Service* e tecnologias abertas com o objetivo de possibilitar a Gerência das Informações de Processos Industriais. Pretende-se com esta abordagem, obter um diferencial em relação aos padrões utilizados na indústria atualmente, através da mudança de foco da programação distribuída, orientada a objetos, para a programação distribuída orientada a serviço. A Figura 4 ilustra os componentes básicos da arquitetura.

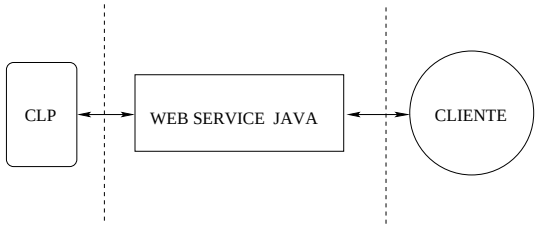

Figura 4: Visão Macro da Arquitetura.

Como neste trabalho utilizou-se uma implementação *Java* de *Web Services*, foi necessário fazer uso da API *Java Native Interface-JNI* para possibilitar a carga e utilização dos *Drivers* dentro do *Web Service*. A JNI é uma interface nativa de programação. Ela permite que códigos nativos sejam executados dentro da Máquina Virtual Java para interoperar com aplicações e bibliotecas escritas em outras linguagens de programação como C, C++ e *Assembly* (Sun, 2006).

A Figura 5 mostra que para o *Web Service* fazer uso dos *Drivers*, de acesso ao CLP, foi necessário implementar um adaptador (DLL - Adapter). Este é responsável por encapsular as funções (*write/read*) de acesso ao CLP e disponibilizá-las para o cliente através do Web Service. A partir desta arquitetura é possível observar a flexibilidade dos clientes, desde o uso de qualquer sistema operacional até o desenvolvimento de aplicações clientes, independentes de linguagem de programação.

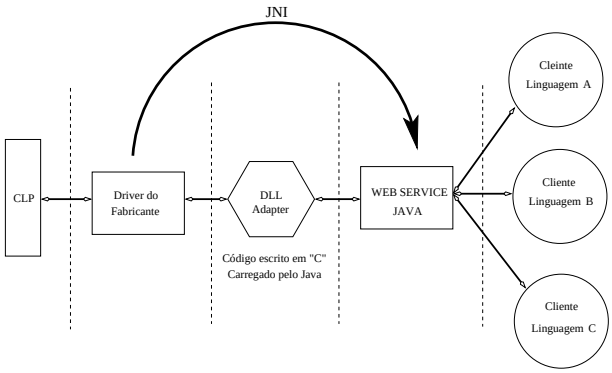

Figura 5: Visão detalhada da Arquitetura.

Na máquina servidora (*Web Service*) foram usados os seguintes artefatos de *Software*:

- *Java Developement Kit* 1.5, para desenvolvimento das classes de acesso aos dados e encapsulamento do *driver* do CLP (Sun, 2006).
- *Apache Tomcat 5.5* como servidor *Web* (Apache, 2006a).
- *Apache Axis* para criação do *Web Service* (Apache, 2006b).

Foi criada uma planta didática, Figura 6, contendo um reservatório d'água, onde o nível deste estava sendo controlado por um CLP SIEMENS, tipo S7-200, que recebia um sinal 4 a 20mA em uma de suas entradas analógicas. Para o controle do nível do reservatório foi usado um Transmissor de Pressão (LD).

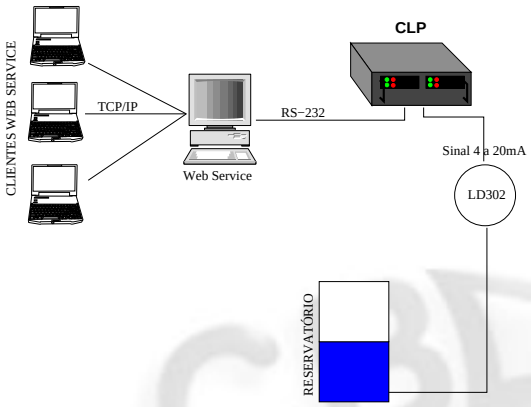

Figura 6: Visão Macro da Arquitetura.

Para que os clientes tivessem acesso ao dados da planta via CLP, foi desenvolvido uma aplicação cliente que se comunicava com o servidor (*Web Service*) através do protocolo de transporte HTTP, possibilitando a visualização e manipulação *on-line* dos dados existentes no CLP. Através desta aplicação cliente foi possível ter acesso às funções do CLP, tipo: leitura, escrita e *status* (ligado/desligado), conforme Figura 7. Para o monitoramento do reservatório d'água foi elaborado uma interface que melhor representasse a planta didática, conforme Figura 8.

A partir do momento que foi inicializado o processo (planta didática) e o *Web Service*, os cliente passaram a obter os dados do CLP. Os clientes e o próprio *Web Service* foram instalados em um seguimento de rede *ethernet* com cerca de trinta máquinas realizando acessos diversos na rede e mesmo assim as respostas às solicitações dos clientes eram em torno de 200ms.

Atualmente existem diversas soluções para aquisição de dados do chão-de-fábrica. O protoloco OPC é um dos mais conhecidos e adota o conceito de encapsular os *Drivers* dos fabricantes de CLPs como este trabalho sugere. Todavia, o OPC adota o uso de programação distribuída orientada a objetos e é baseado na tecnologia DCOM da *Microsoft*. Com a utilização de uma aquitetura orientada a serviços podemos constatar que é possivel oferecer os mesmos serviços que tecnologias como OPC oferecem, acresentando características como interoperabilidade, portabilidade, escalabilidade entre outros.

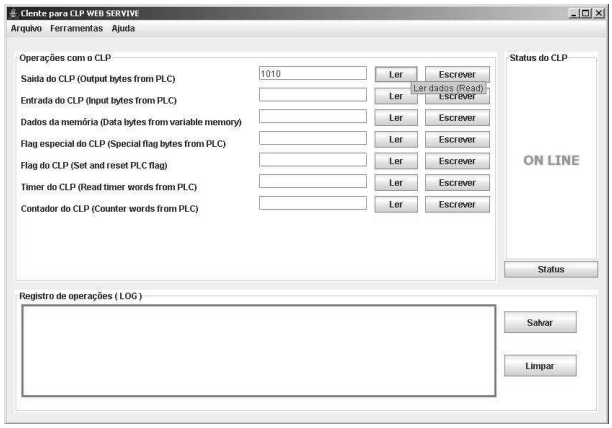

Figura 7: Cliente *desktop*.

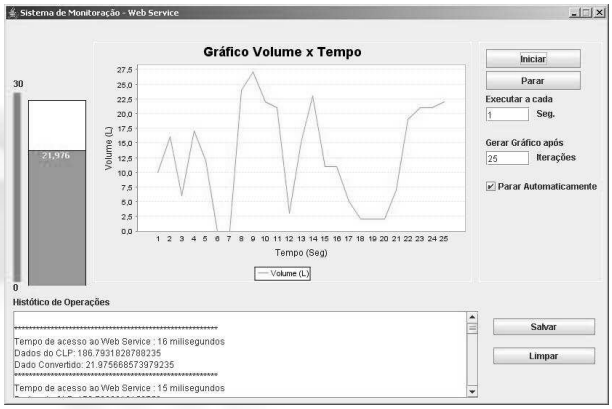

Figura 8: Monitoramento do reservatório.

5 CONCLUSÃO

O setor de automação industrial tem sofrido notória evolução, principalmente impulsionada pela Tecnologia da Informação, que vem sofisticando ainda mais este seguimento. Entretanto, mesmo com essa sofisticação, é possível observar uma forte resistência a novos padrões, principalmente os abertos, ficando o setor dependente de padrões fechados dificultando a adoção de novas e modernas tecnologias. Desfrutar das tecnologias baseadas nos padrões abertos ainda não é uma realidade na indústria, isso graças à filosofia conservadorista adotada por este seguimento, restringindo o desenvolvimento de aplicações à tecnologias proprietárias.

A adoção de uma arquitetura orientada a serviços proporciona uma série de vantagens no que tange às possibilidades de acessar os dados do chão-de-fábrica e dissiminá-los aos diversos setores de uma organização. O uso dos *Web Services*

trouxe essa realizade à tona. A independêcia de acesso a estes, possibilita que clientes via *Browser*, aplicações *Desktops* ou, até mesmo, dispositivos móveis como celulares e PDAs, possam de forma transparente e sem adaptações ter acesso aos diversos dados de um processo de automação.

Entretanto, garantir a segurança destes serviços pode impor desafios devido às limitações dos padrões e à necessidade de oferecer suporte a uma grande variedade de tipos de clientes. A segurança de *Web Services* pode ser aplicada em três níveis: Segurança a nível de transporte; Segurança a nível de aplicação e Segurança a nível de mensagem. Tais estratégias serão abordadas em trabalhos futuros para que se possa avaliar os impactos gerados.

Desta forma, podemos concluir que em um futuro próximo a arquitetura orientada a serviços estará efetivamente aplicada na indústria. Esta atuação será observada desde o chão-de-fábica até os níveis de decisão corporativa. Tal mudança reflete as exigências, atuais, dos processos de automação, tais como: portabilidade, interoperabilidade, robustez e escalabilidade.

Referências

Apache, S. F. (2006a). Apache tomcat. Disponível em: http://tomcat.apache.org/.

Apache, S. F. (2006b). Web services project apache. Disponível em: http://ws.apache.org/.

Celso, J. M. (2004). Considerations about implementation and applications of opc data access and opc xml data access standards in industrial automation, *VI INDUSCON* .

Francisco, C. (2002). Unraveling the web service web an introduction to soap, wsdl and uddi, *IEEE Computer Society* .

François, J. and Smit, H. (2005). Service-oriented paradigms in industrial automation, *IEEE Transactions on Industrial Informatics, Vol. 1, N°. 1, pp 62-70* .

OPC, F. (2006). What is the opc foundation? Disponível em: http://www.opcfoundation.org/.

Sun, M. (2006). Java 2 platform, standard edition (j2se). Disponível em: http://java.sun.com/j2se/.

Szolkowski, M. (2003). *JavaServer Pages Developer's Handbook*, Pearson.

Vinoski., S. (2005). Web service references, *IEEE Computer Society* .

Vogels., W. (2003). Web service are not distributed objects, *IEEE Computer Society* .

W3C (2004). Web services architecure w3c working grop note. Disponível em: http/www.w3.org/TR/2004/NOTE-ws-arch-20040211/.

Yi Zhi, Z. (2005). Service-oriented architecure and technologies for automating integration of manufacturing systems and service, *IEEE Computer Society* .