UMA ABORDAGEM PARA A INTEGRAÇÃO PLENA DE SISTEMAS EMPRESARIAIS ATRAVÉS DE SERVIÇOS WEB

ANDRÉ P. PIAZZA, RICARDO J. RABELO

GSIGMA, Departamento de Automação e Sistemas, Universidade Federal de Santa Catarina
Caixa Postal 476, 88040-900 Florianópolis, SC, Brasil
E-mails: piazza@das.ufsc.br, rabelo@das.ufsc.br

Abstract— Nowadays enterprises have to deal with a more competitive and changing market. To remain competitive, enterprises have intensified the investments on ICT in order to improve their business processes by means of using more enterprise systems, which are present since the shop-floor till management levels of a company. However, this is generating a very heterogeneous scenario, with distributed systems from different vendors, using different technologies and responsible for different levels of the enterprise, hence creating technological silos inside the company. Therefore, it is imperative to provide ways to integrate these different systems, in a standard, open and dynamic way. To accomplish that, it is necessary to apply an approach that follows a decentralized and loosely-coupled model, allowing different systems to achieve a transparent and interoperable collaboration. This paper proposes an approach to ensure that different systems can be integrated in a complete way, making use of the service oriented paradigm and using Web services technology.

Keywords — Enterprise systems, systems integration, interoperability, collaboration, Web services.

Resumo— Cada vez mais empresas têm enfrentado um ambiente de intensa concorrência e mudanças de mercado. Para se manterem competitivas, empresas têm intensificado o investimento na TIC para melhorar os processos de negócios, através do uso de sistemas empresariais, que vão desde o chão de fábrica até os níveis de gerência. Porém, isso cria um cenário de uma enorme quantidade de sistemas heterogêneos distribuídos de diversos fabricantes, que utilizam diferentes tecnologias e que são responsáveis por diferentes níveis da empresa, fazendo com que se criem silos tecnológicos. Portanto, é preciso que sejam providas formas de integrar esses diferentes sistemas, de maneira padronizada, aberta e dinâmica. Para tal, é necessário que se utilize uma abordagem que siga um modelo descentralizado e de baixo acoplamento, permitindo que os diferentes sistemas possam participar de maneira transparente e interoperável em processos colaborativos. Esse trabalho propõe uma abordagem para se garantir que diferentes sistemas possam ser integrados de maneira plena, se baseando no conceito de orientação a serviços, e utilizando a tecnologia de serviços Web.

Palavras-chave — Sistemas empresariais, integração de sistemas, interoperabilidade, colaboração, serviços Web.

1 Introdução

A Tecnologia da Informação e Comunicação (TIC) vem tendo um papel de extrema importância em diversas áreas da indústria. A TIC é responsável pela automatização da maioria dos processos das empresas atuais, sejam estes relacionados com manufatura, distribuição, vendas, ou qualquer outro tipo de processo de negócio (KRAFZIG *et al.*, 2004). Diferentes sistemas de TIC, como os sistemas MES (*Manufacturing Execution System*), ERP (*Enteprise Resource Planning*), CRM (*Customer Relashionship Management*) e SCM (*Supply Chain Management*) automatizam diferentes áreas de negócios, permitindo um acesso mais direto e ágil às informações, permitindo também um maior compartilhamento de recursos computacionais, aumentando a colaboração entre diferentes níveis de uma empresa.

Porém, com a velocidade com que novas tecnologias são lançadas e adotadas, surge o problema de sistemas e das infra-estruturas que acabam não mais atendendo aos requisitos necessários. Isto porque os processos de negócios das empresas têm sido alterados com cada vez maior freqüência, resultado tanto de melhorias e adaptações internas, como de requisitos de clientes e do reflexo de inovações. Isto engloba alterações nos sistemas existentes/legados, o que é difícil, custoso, moroso e as vezes até mesmo inviável, dependendo da qualidade e da flexibilidade do projeto original do sistema, e das TICs usadas nas suas implementações (BISBAL *et al.*, 1999).

Com isso, surge um grande problema que é a falta e interoperabilidade entre esses diferentes sistemas novos e legados, heterogêneos, resultando numa enorme dificuldade (técnica, custos e tempo) de automação dos processos dos diferentes setores dentro da empresa. Grande parte dessa integração acaba sendo feita de maneira manual, através de adaptações *ad-hoc* e interfaceamentos proprietários entre diferentes sistemas.

Visando minimizar esse problema, um esforço grande vem sendo feito no sentido de se criar novos conceitos e tecnologias para suportar essa necessidade de integração e colaboração entre os diferentes níveis dentro de uma organização, de maneira padronizada e universal. Os sistemas MES, ERP, SCM, CRM, por exemplo, antes aplicações independentes e isoladas, tendem a se tornar uma única infra-estrutura de serviços de alto nível, integrada e "sem costuras"[1]. Em muitas empresas isso é feito via Portais.

No que toca a integração com aplicações mais de chão-de-fábrica, abordagens iniciais para se

[1] Do inglês, *seamless*

resolver este problema basearam-se na de EAI[2], que possibilita a automatização e integração desses diferentes sistemas em uma única infra-estrutura centralizada (PUSCHMANN *et al.*, 2004).

Porém, os custos de implantação de soluções de EAI são altos e, além disso, são soluções majoritariamente monolíticas, pouco modulares, e, assim, pouco flexíveis para ágeis mudanças nos processos e novas integrações de sistemas. Desta forma, na prática, no modelo proposto pelas soluções de EAI, a dificuldade e custos são praticamente similares aos das integrações iniciais.

Para se atingir os benefícios reais de uma integração de maneira plena, é necessário que se utilize uma abordagem que reflita o dinamismo de uma empresa, onde seja possível integrar os sistemas existentes e também soluções de integração previamente implantadas, de acordo com novas necessidades do negócio e mudanças nos processos.

Uma empresa possui vários setores e estes precisam trabalhar cooperativamente para que ela possa funcionar de forma mais eficiente. Para tal, é preciso que a abordagem de integração siga um modelo descentralizado e de baixo acoplamento[3], permitindo que sistemas autônomos possam participar de maneira transparente e interoperável em processos colaborativos (internos ou externos à organização), independentemente de uma arquitetura centralizada e proprietária de suporte a tal colaboração.

Essa é a principal motivação desse trabalho, o de prover uma forma de garantir que diferentes sistemas possam ser integrados de maneira plena, provendo o máximo de transparência e flexibilidade para que sistemas de todos os níveis da empresa possam trabalhar cooperativamente sem a necessidade de intervenções manuais ou de construções de interfaces caso-a-caso.

Este artigo está organizado da seguinte maneira: A seção 2 apresenta uma visão geral do cenário o qual a abordagem é proposta, resumo das tecnologias, descrição do problema e a proposta do trabalho. A seção 3 apresenta o modelo conceitual, que apresenta todos os componentes relevantes, porém independentemente de tecnologias. A seção 4 apresenta detalhes da implementação do protótipo. A seção 5 apresenta as conclusões do trabalho e desafios existentes.

[2] EAI (*Enterprise Application Integration*) é o processo de interconexão entre diferentes sistemas dentro de uma empresa.

[3] Baixo acoplamento, do termo em inglês *loose coupling*, descreve uma relação entre dois ou mais sistemas que podem interagir entre si, porém de forma independente.

2 Cenário Geral

2.1 Arquitetura Orientada a Serviços

Quando se fala em integração e eficiência das empresas no mundo atual, não se pode mais desconsiderar a Internet. Soluções modernas e de maior ciclo de vida passam a ser cada vez mais concebidas como soluções baseadas na Web.

A integração plena entre sistemas e processos negócios se torna possível com a utilização da Internet como meio de comunicação, e com o auxílio de uma infra-estrutura de TIC que seja transparente, de fácil utilização, aberta e de baixo custo (CAMARINHA-MATOS, 2003). Esta infra-estrutura deve servir como um *middleware*, devendo ser implementada com a utilização de tecnologias e padrões abertos, de forma que a empresa não fique atada a um único fabricante. Essa infra-estrutura provê a interoperabilidade entre os vários sistemas heterogêneos, independente da linguagem, *framework* e sistema operacional deles (RABELO *et al.*, 2007).

O paradigma de orientação a serviços, que é uma evolução das abordagens baseadas em objetos, tem o potencial para tratar os problemas mencionados, principalmente no provimento de integração, reusabilidade e interoperabilidade entre os atuais ambientes computacionais heterogêneos das empresas (LI *et al.*, 2006). A Arquitetura Orientada a Serviços (*Service Oriented Architecture* - SOA) é uma abordagem de projeto e de integração de sistemas que permite uma interação entre diversas aplicações independente da plataforma, através da utilização de serviços genéricos e padronizados, tanto intra como inter-empresa (FEUERLICHT, 2006).

Numa filosofia SOA, uma aplicação não é mais desenvolvida como um software monolítico, um "pacote", mas sim como uma *composição* de *serviços de software*, interligados de acordo com a lógica do processo. Pode-se fazer uma analogia com um "lego". Conforme a montagem feita das "peças" (serviços), diferentes "produtos" (aplicações) podem ser criados. E quando se deseja alterar alguma funcionalidade, basta procurar na "caixa", achar a(s) peça(s) desejada(s), trocar as necessárias, e todo o resto do sistema se manterá "acoplado" (integrado), pois os "pinos" (interfaces) são padronizados. O "tamanho da peça" (granularidade do serviço) pode ser muito variável.

2.2 Serviços Web

A forma que largamente mais tem sido utilizada para implementar a filosofia SOA é através da tecnologia de serviços Web (*web services*), que são fortemente amparados em padrões abertos.

Um serviço Web é uma aplicação modular, independente e reutilizável para ser empregado em um processo de negócios, e que possui uma

interface aberta, padrão e baseada na Internet. Essa interface pode ser descrita, publicada, descoberta e invocada utilizando protocolos padrão baseados na linguagem XML, como o WSDL (*Web Services Description Language*), SOAP (*Simple Object Access Protocol*) e UDDI (*Universal Description, Discovery and Integration*) (AUSTIN *et al.*, 2004).

Um aspecto importante da tecnologia de serviços Web é que toda a funcionalidade ou processo de negócio que uma organização deseja disponibilizar será exposto na forma de um serviço (FEUERLICHT, 2006). Nesse cenário, diversos serviços Web são definidos, desenvolvidos e gerenciados pela empresa (ou mesmo por outras), tornando-os uma tecnologia de baixo acoplamento, e, por utilizar padrões abertos, também permite uma grande interoperabilidade entre diferentes sistemas e organizações. Isso significa que a partir do momento que todas as aplicações (novas ou existentes) utilizarem interfaces padrão, elas poderão ser acessadas por qualquer outra aplicação, até mesmo por outras empresas de fora.

2.3 Descrição do Problema e Proposta

A comunicação sem costuras, transparente e dinâmica entre sistemas computacionais facilita a colaboração, tornando a execução dos processos de negócio mais ágeis. Os serviços Web foram criados com o intuito de atingir essa transparência, utilizando a Web como meio e repositório de serviços, que são disponibilizados de maneira padrão, e podem ser executados por qualquer organização (ANDRADE *et al.*, 2003).

Porém, apesar de estarem trabalhando com tecnologias padrão, quando as empresas desejam fazer seus *frameworks* e serviços (normalmente desenvolvidos em diferentes plataformas e sistemas de informação) se comunicarem plenamente entre si, diversos problemas de interoperabilidade surgem na prática (EGYEDI, 2006), dentre eles:

- Os diversos fabricantes de software e *frameworks* para serviços implementam as especificações dos serviços Web com alguns aspectos fora da norma.
- A necessidade da aplicação-cliente de conhecer previamente a localização do serviço para então se poder criar, de forma estática (em tempo de projeto da aplicação) e manual, os necessários *stubs* e *proxies* para a invocação do serviço.

Portanto, o problema é como se pode garantir uma integração plena e "sem costuras" entre os sistemas de uma empresa (ou entre empresas) de forma que esta possa atuar com mais agilidade e ao mesmo tempo com mais flexibilidade quando da mudança de processos.

3 Modelo Conceitual

Como dito no capítulo 1 e embasado no capítulo 2, este artigo propõe uma abordagem para uma integração plena de sistemas de uma empresa, através da localização e invocação dinâmica e interoperável de serviços Web em um ambiente heterogêneo, independentemente da plataforma computacional, do sistema operacional, das tecnologias e linguagens de implementação, e da localização geográfica do serviço. Essa seção tem o objetivo de detalhar o modelo conceitual proposto. A arquitetura proposta é baseada no conceito de orientação a serviços, mantendo assim as entidades envolvidas, i.e., aplicações-cliente, provedores e registros desacoplados e independentes uns dos outros. A Figura 1 apresenta o cenário geral e a arquitetura proposta pelo trabalho, vista como um *middleware* de suporte a integração e colaboração.

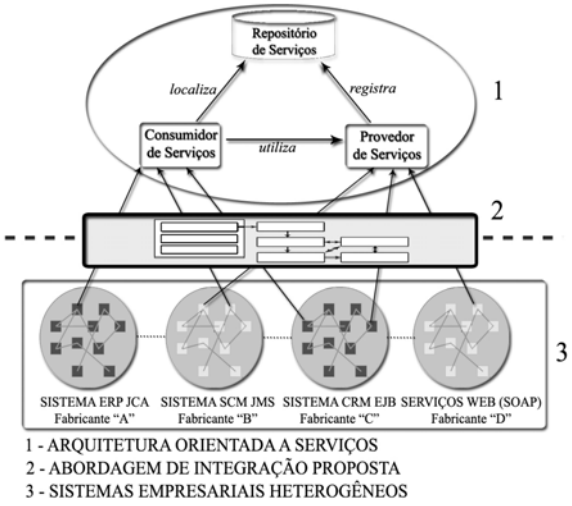

Figura 1. Posicionamento da Abordagem Proposta

3.1 Entidades envolvidas

É no *Repositório de serviços* que a empresa publica todas as operações e aplicações de seus diferentes sistemas, tornando-os disponíveis para que outros serviços (internos ou externos à empresa) possam localizá-los e executá-los, de acordo a lógica do processo.

O *Provedor de serviços* dentro do contexto da abordagem proposta, pode ser entendido como qualquer entidade ou sistema interno ou externo a uma empresa e que provê algum tipo de serviço, para a execução de processos de negócio, que são publicados no Repositório.

O *Consumidor de serviços* (ou *Aplicação-Cliente*), dentro do contexto da abordagem proposta, é a entidade que, ao contrário do provedor, utiliza os serviços que são disponibilizados no Repositório. Para isso, o Consumidor faz uma consulta por serviços no Repositório, que, ao ser localizado, retorna as instruções de como utilizar esse serviço. Após isto,

cabe ao Consumidor processar essas instruções e criar sua lógica para acesso às funcionalidades oferecidas pelo provedor.

3.2 Módulos da abordagem proposta

A Figura 1 apresenta uma visão geral da arquitetura, onde na camada 2 é ilustrada a abordagem proposta por este trabalho. Esta camada engloba oito diferentes módulos, que são responsáveis por tarefas específicas dentro do modelo que, essencialmente, atuam como uma meta-interface entre sistemas-clientes existentes da empresa e o(s) repositório(s) de serviços desta, tornando completamente transparente o acesso dos serviços pelos sistemas.

Os módulos *Publicador*, *Removedor* e *Localizador de Serviços* são os responsáveis pela interação com o Repositório de serviços, e podem ser utilizados tanto pelo Consumidor quanto pelo Provedor de serviços.

Os módulos *Processador de interface* e *Seletor do tipo de serviço* são responsáveis pela extração de dados relevantes para a invocação de um dado processo de negócio. O módulo de *Suporte a invocações* é responsável pela invocação "multi-serviços". Neste trabalho este módulo é provido pelo WSIF (ver seção 4.2).

Os módulos *Construtor* e *Invocador* são os responsáveis pela construção da chamada e invocação propriamente dita do serviço.

4 Implementação

Essa seção apresenta as entidades e componentes genéricos apresentados na seção anterior, instanciando-os com as tecnologias associadas aos serviços Web que foram usadas na implementação.

Em um cenário empresarial, que faz uso dos serviços Web para a integração de suas aplicações, existem diferentes entidades que se comunicam e colaboram entre si, através do uso de protocolos abertos e padronizados, de forma a garantir a interoperabilidade entre os diferentes sistemas presentes tanto dentro quanto fora da empresa. Essas entidades, que foram apresentadas de maneira genérica na seção anterior, são: repositório, provedor e aplicação cliente.

Nos serviços Web, o repositório é implementado utilizando as especificações da UDDI, e é a entidade responsável pelo gerenciamento do armazenamento, publicação e localização interna de serviços e processos de negócio, e também das instruções de utilização dos serviços (aqui chamadas de interfaces).

O provedor de serviços é o responsável pelo provimento de uma funcionalidade, e do desenvolvimento da lógica que irá expor essa funcionalidade como um serviço Web. Isso inclui a criação da aplicação, ou da utilização de uma aplicação existente (como um sistema ERP, SCM, etc), e a sua publicação na forma de um serviço Web. Cabe também ao provedor a publicação no repositório UDDI de detalhes sobre o serviço ou processo de negócio, e também a publicação da interface que é utilizada pelos consumidores (clientes) de serviços para a criação de seus procedimentos para a utilização do sistema.

O consumidor de serviços representa, em uma empresa, qualquer entidade que deseja utilizar e invocar operações de outro sistema ou serviço, esteja este dentro ou fora das fronteiras da empresa. É a entidade responsável por localizar a descrição do serviço (interface WSDL), executar o processamento desta interface, extrair dados relevantes, para então criar os procedimentos de invocação do serviço Web.

4.1 Invocação Dinâmica e Stubless

O objetivo da invocação dinâmica e sem a necessidade de *proxies* ou *stubs* (*stubless*) é o de permitir que uma aplicação cliente possa efetivamente utilizar qualquer serviço provido pela empresa e que esteja disponível em um repositório UDDI, de forma transparente e automática, independentemente da tecnologia e ambiente que esse serviço foi desenvolvido e disponibilizado.

A Figura 2 apresenta a forma de invocação *stubless* e independente de tecnologia/ protocolos que é proposta pelo modelo da abordagem, e para fins de comparação, ilustra abaixo a forma tradicional de invocação de serviços Web.

Baseando-se nos requisitos de suporte ao cenário almejado, onde diferentes sistemas interoperam efetivamente de maneira independente e aberta, este trabalho faz uso de um conjunto de APIs que podem ser utilizadas de forma que seja possível atingir uma invocação automática e dinâmica de serviços Web, independentemente de detalhes técnicos e tecnologias (SOAP, EJB, JMS, etc.) envolvidos.

4.2 Utilização do WSIF

As APIs do Apache WSIF foram escolhidas para o desenvolvimento do invocador dinâmico e *stubless*, que irá prover a transparência à aplicação na invocação dos diferentes serviços, independente dos mecanismos de acesso envolvidos.

A principal vantagem da utilização do WSIF é que, através deste, é possível visualizar a Arquitetura Orientada a Serviços como algo mais extensível que apenas serviços e ligações SOAP. Existe atualmente uma grande quantidade de tecnologias para sistemas distribuídos e de diferentes protocolos que podem ser encontrados nos diferentes níveis de uma empresa, o que torna essa flexibilidade de uso de diferentes protocolos uma característica mandatória para uma integração plena de sistemas heterogêneos.

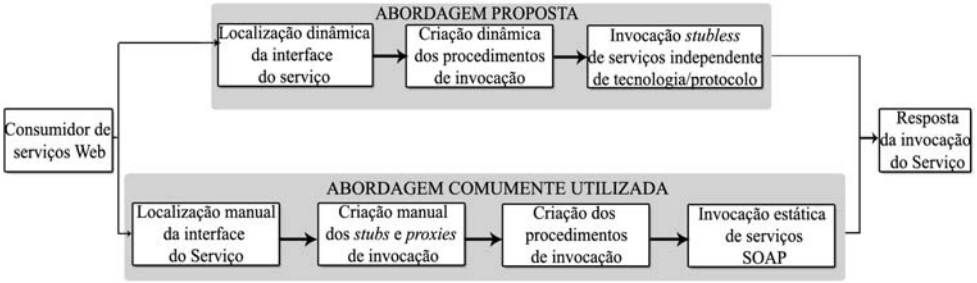

Figura 2. Modelo proposto.

Outras vantagens providas pelo invocador *stubless* proposto são:

- Possibilidade de se ter um mecanismo independente de ligações (*bindings*) na invocação de serviços Web.
- Invocação automática e dinâmica de serviços, sem a criação prévia de *stubs*.
- Maior interoperabilidade, independentemente do tipo de sistema ou serviço.

Com base nas vantagens descritas acima, é possível verificar um claro aumento na agilidade e eficiência dos processos de negócio empresariais. Porém, algumas limitações e desvantagens também surgem, como por exemplo o aumento do processamento na invocação de serviços e a dependência da linguagem Java no código cliente. No entanto, em comparação com outras abordagens de interoperabilidade, a proposta deste artigo ainda se torna vantajosas devido ao fato da localização e invocação de serviços ser altamente dinâmica, além do suporte a invocação de outros tipos de serviços, como o *document/literal*, por exemplo.

4.3 Arquitetura do WSIF

A idéia por trás do uso do WSIF é a sua simplicidade. Além da possibilidade de se utilizar múltiplas ligações (*bindings*), o fato de depender apenas da descrição da interface (WSDL) do serviço para a utilização de serviços providos por diferentes sistemas se torna muito atrativo em um cenário heterogêneo como o de uma empresa, ou até mesmo em uma integração e utilização de sistemas e serviços de diferentes empresas.

A API do WSIF permite que clientes invoquem serviços focando na descrição WSDL abstrata do serviço, sem se referir aos protocolos e tecnologias reais. Essa invocação abstrata é possível graças aos módulos que são providos pelo WSIF, que são específicos para cada tipo de protocolo, chamados de *Provedor*. Um Provedor no WSIF é responsável por lidar com as mensagens que serão trocadas, de acordo com as especificações de cada protocolo.

A Figura 3 ilustra a arquitetura de integração proposta em maiores detalhes. No nível mais superior (1) estão os diferentes sistemas empresariais heterogêneos/legados (ERP, SCM, etc), cada um utilizando uma tecnologia diferente (JCA, JMS, etc), porém, agora com uma interface WSDL expondo de forma padronizada detalhes da forma como utilizar os serviços providos por esses sistemas. No nível mais inferior (3) está a aplicação cliente propriamente dita, que deseja fazer uso, de maneira transparente, integrada e plena, de um sistema empresarial, independente de qual tecnologia esteja sendo utilizada por esse sistema. No nível (2) é onde estão localizados os módulos responsáveis pela invocação do serviço nos sistemas. Essa invocação independente de protocolo é gerenciada pela *WSIF Engine*, que se baseia no tipo de serviço/tecnologia utilizado pelo sistema, para então utilizar um *Provedor* específico para tal.

4.4 Detalhes da Implementação

Devido ao fato do WSIF ser desenvolvido totalmente na linguagem Java, a linguagem padrão utilizada para o desenvolvimento do protótipo – tanto as ferramentas de invocação quanto as de UDDI que serão descritas abaixo – é também o Java.

Um provedor de serviços normalmente torna seus serviços disponíveis através de sua publicação em um repositório UDDI. Quando uma aplicação-cliente precisa invocar um serviço, é necessário que se localize o serviço e sua descrição WSDL. Os módulos criados para tratar de todas as operações referentes à UDDI incluem publicação e atualização

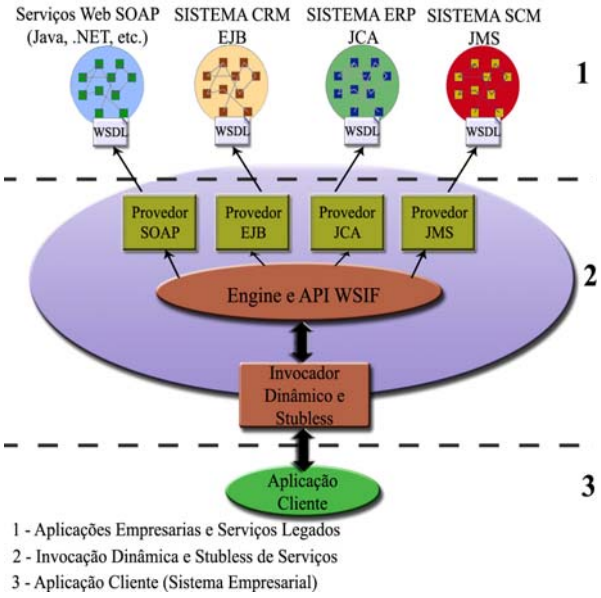

Figura 3. Integração de sistemas heterogêneos via serviços Web

de serviços, e também a localização e retorno da URL contendo a descrição do serviço a ser invocado.

Após isso, os módulos encarregados da invocação dinâmica e *stubless* de serviços irão ler o documento WSDL extraindo as informações necessárias para que os procedimentos de invocação sejam criados automaticamente e o serviço seja consumido, sem a necessidade de qualquer intervenção manual e trabalhos adicionais, i.e., nenhuma geração de *stubs* ou *proxies*. Com isso, a invocação dos serviços se torna transparente e dinâmica para a aplicação cliente, bastando apenas a instanciação da classe do invocador dinâmico, com a WSDL e a operação a ser executada.

4.5 Validação do protótipo

A validação do protótipo foi feita através de testes de invocação em diferentes serviços Web. Estes serviços foram criados e disponibilizados em diferentes plataformas heterogêneas e distribuídas de forma a contemplar a proposta de interoperabilidade e independência de plataforma do presente trabalho.

A invocação destes serviços foi avaliada com base em ferramentas disponibilizadas pela WS-I, que é um órgão de referência dentro da área de serviços Web e interoperabilidade. Estas ferramentas foram utilizadas, gerando relatórios de conformidade, que mostraram que todos os serviços puderam se invocar uns aos outros com plena interoperabilidade.

5 Conclusão

A integração de sistemas empresariais contribui efetivamente para o aumento da produtividade, qualidade e dinamicidade dos processos de negócio de uma empresa, sejam estes internos ou externos. Ao tornar acessíveis publicamente funcionalidades que antes eram isoladas em um único sistema, são abertas novas possibilidades de negócio, além de uma maior agilidade e facilidade no acesso a informações e na execução de processos de negócio por parte de membros da empresa.

Esse artigo apresentou uma abordagem para a integração plena de sistemas empresariais, através da localização e invocação dinâmica de serviços, independente de qual sistema esse serviço tenha sido desenvolvido. O desenvolvimento foi baseado no conceito de orientação a serviços, utilizando essencialmente a tecnologia de serviços Web para o provimento de uma forma de se invocar operações de diferentes sistemas de maneira transparente, dinâmica e *stubless*, sem a necessidade de intervenções manuais por parte da aplicação cliente. Com isso, cria-se um nível mais elevado de interoperabilidade entre os sistemas / processos de das empresas, uma vez que os serviços ficam acessíveis independentemente de quais plataformas e tecnologias foram usadas. Em uma escala maior, este trabalho cria uma importante base para uma integração e interoperabilidade plena entre sistemas e empresas, dentro de um cenário onde se prevê a criação e existência de federações de serviços.

Agradecimentos

Este trabalho foi suportado e desenvolvido no escopo do projeto brasileiro IFM (www.ifm.org.br) e do europeu ECOLEAD (www.ecolead.org).

Referências Bibliográficas

ANDRADE ALMEIDA, J. P.; FERREIRA PIRES, L.; VAN SINDEREN, M. J.; 2003. Web services and seamless interoperability. In: First European Workshop on Object-Orientation and Web Services (ECOOP 2003) (Darmstadt, Germany).

AUSTIN, D.; BARBIR, A.; FERRIS, C.; GARG, S. (2004). "Web Services Architecture Requirements." Acessado: 12 de Março, 2007, em http://www.w3.org/TR/wsa-reqs.

BISBAL, J.; LAWLESS, D.; WU, B.; GRIMSON, J.; 1999. Legacy Information System Migration: A Brief Review of Problems, Solutions and Research Issues. *IEEE Software*, v. 16, n. 5.

CAMARINHA-MATOS, L. M.; 2003. Infrastructures for virtual organizations - where we are. In: Emerging Technologies and Factory Automation, 2003. Proceedings. ETFA '03. IEEE Conference (Lisboa, Portugal). *Proceedings*.

EGYEDI, T. M.; 2006. Standard-Compliant, but Incompatible?! In: I-ESA'06 - Interoperability for Enterprise Software and Applications Conference (Bordeaux, França). *Proceedings*.

FEUERLICHT, G.; 2006. Enterprise SOA: What are the benefits and challenges? In: International Conference Systems Integration 2006 (Prague, Czech Republic). *Proceedings*.

KRAFZIG, D.; BANKE, K.; SLAMA, D.; 2004. *Enterprise SOA: Service-Oriented Architecture Best Practices*. Upper Saddle River, NJ, USA: Prentice Hall.

LI, M.-S.; CABRAL, R.; DOUMEINGTS, G.; POPPLEWELL, K.; 2006. Enterprise Interoperability - Research Roadmap (Final Version - v4.0) - Information Society Technologies - European Commission.

PUSCHMANN, T.; ALT, R.; 2004. Enterprise application integration systems and architecture – the case of the Robert Bosch Group. *The Journal of Enterprise Information Management*, v. 17, n. 2.

RABELO, R. J.; GUSMEROLI, S.; 2007. A Service-Oriented Platform for Collaborative Networked Organizations. In: 8th IFAC Symposium on Cost Oriented Automation Affordable Automation Systems (Cuba). *Proceedings*.