CENTRO UNIVERSITÁRIO FEEVALE

MARCO AURÉLIO BRAUN

MODELO DE INTEGRAÇÃO ENTRE DIFERENTES SOFTWARES APLICATIVOS POR UM BARRAMENTO DE SERVIÇOS CORPORATIVOS BASEADO EM UMA ARQUITETURA ORIENTADA A SERVIÇOS

Novo Hamburgo, julho de 2009.

MARCO AURÉLIO BRAUN

MODELO DE INTEGRAÇÃO ENTRE DIFERENTES SOFTWARES APLICATIVOS POR UM BARRAMENTO DE SERVIÇOS CORPORATIVOS BASEADO EM UMA ARQUITETURA ORIENTADA A SERVIÇOS

Centro Universitário Feevale
Instituto de Ciências Exatas e Tecnológicas
Curso de Sistemas de Informação
Trabalho de Conclusão de Curso

Professor Orientador: Alexandre de Oliveira Zamberlam

Novo Hamburgo, julho de 2009.

AGRADECIMENTOS

Gostaria de agradecer a todos os que, de alguma maneira, contribuíram para a realização desse trabalho de conclusão, em especial:

À minha esposa e filhos, minha gratidão, pelo apoio emocional e compreensão pelos momentos de ausência - dedicados à realização deste trabalho.

RESUMO

Durante anos, as empresas desenvolvedoras de software geraram uma incontável quantidade de sistemas, baseados nas mais diversas linguagens de programação e arquiteturas. Esses sistemas são o resultado da demanda gerada a partir da evolução dos negócios corporativos. Muitas empresas ainda mantêm sistemas antigos em pleno funcionamento, convivendo com softwares aplicativos baseados em tecnologias mais modernas. Uma alternativa que pode ser aplicada na integração de todos esses softwares é a *Arquitetura Orientada a Serviços - Service Oriented Architecture* (SOA). SOA combina regras de negócios entre aplicações através de serviços intercambiáveis. Um componente ou ferramenta integrante de SOA é o Barramento Corporativo de Serviços – *Enterprise Service Bus* (ESB). ESB tem a capacidade de fazer com que sistemas heterogêneos comuniquem-se de forma transparente. Sendo assim, este trabalho tem o objetivo de estudar, definir o funcionamento e gerar um modelo de um ESB. Dessa forma, servir de referência às empresas desenvolvedoras de software, promovendo o reuso de componentes (linhas de código) e combinando recursos já existentes, no intuito de dar mais agilidade às mesmas, com objetivo de que atinjam uma vantagem estratégica.

Palavras-chave: SOA. ESB. Integração de Sistemas. Arquitetura de Software.

ABSTRACT

For many years, companies in software development created a countless number of systems, based in several programming languages and architectures. These systems are the result of demand generated from the development of corporate business. Many companies still have systems in full operation, living with software applications based on latest technology. An alternative that can be applied in the integration of all such software is the Service Oriented Architecture (SOA). SOA combines business rules between applications through services interchangeable. An integral component or tool of SOA is the Enterprise Service Bus (ESB). ESB has the ability to communicate with heterogeneous systems that are transparent manner. Thus, this work aims to study, to define the operation and to generate a ESB model. So, it serves the companies as a reference for companies in software development, promoting the reuse of components (lines of code) and combining existing resources in order to give more flexibility to them, with the aim of reaching a strategic advantage.
Key words: SOA. ESB. Systems Integration. Software Architecture.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

API	Application Programming Interface
BPM	Business Process Modeling
CMMI	Capability Maturity Model Integration
DLL	Dymanic Link Library
EAI	Enterprise Application Integration
ESB	Enterprise Service Bus
J2EE	Java 2 Enterprise Edition
MBA	Mestrado em Administração de Empresas
SOA	Service Oriented Architecture
SOAP	Simple Object Access Protocol
TI	Tecnologia da Informação
XML	Extensible Markup Language

SUMÁRIO

INTRODUÇÃO

Nos últimos anos, a tecnologia da informação sofreu fortes e rápidas mudanças. As empresas desenvolvedoras de software geraram uma infinidade de sistemas, a partir das mais variadas tecnologias (linguagens de programação, arquiteturas...). "O atual panorama corporativo é composto de sistemas que são o resultado da evolução dos negócios e da tecnologia com o passar dos anos." (CUMMINS, 2002, p.2).

Os sistemas legados[1] utilizam-se das mais variadas formas de integração, com o uso de tecnologias diferentes. Esse cenário é assim descrito:

> "Após muitos anos de desenvolvimento de aplicações de computador em um panorama tecnológico em evolução, a maioria das empresas está carregada de computadores e aplicações que estão interligados através de uma variedade de mecanismos específicos." (CUMMINS, 2002, p.XIII).

Essa diversidade acaba por determinar um forte acoplamento[2] entre as aplicações, com uma estrutura muito rígida de comunicação. Para cada interligação entre aplicações, existe uma tecnologia, forma ou meio diferentes. Isso acaba por tornar-se de difícil manutenção e depende do despendimento de recursos.

> "Os grandes sistemas utilizam plataformas diferentes, linguagens de programação diferentes (e paradigmas de programação) e até middleware[3] diferente. Eles são uma bagunça de mainframes, clientes SAP, bancos de dados, aplicações J2EE, pequenos motores de regras, e assim por diante. Em outras palavras, eles são heterogêneos." (JOSUTTIS, 2008, p. 13).

Além dos sistemas legados, desenvolvidos por uma mesma empresa, há também os casos em que empresas diferentes têm a necessidade de realizar a integração entre seus aplicativos, para atingir um objetivo em comum, que é a satisfação da necessidade de um cliente em comum. "SOA inclui práticas e processos que são baseados no fato de que redes de sistemas distribuídos não são controlados por um único proprietário. Equipes diferentes, departamentos diferentes ou até empresas diferentes podem gerenciar os sistemas diferentes." (JOSUTTIS, 2008, p. 13).

Dessa forma, uma empresa escreve seus softwares aplicativos alicerçados em uma determinada plataforma e em uma linguagem de programação específica, além de manter os dados dos mesmos em um banco de dados em particular. Na maioria dos casos, essas características não são reproduzidas em um todo pela outra empresa. Mesmo que haja

---

[1] "Termo normalmente utilizado pra fazer referência a hardware ou software antigos que não obedecem aos padrões ou níveis de desempenho mais recentes" (SAWAYA, 2002, p.261).
[2] Alta dependência entre softwares aplicativos integrados (JOSUTTIS, 2008).
[3] "O middleware gerencia a interação entre aplicativos por meio de plataformas de computadores heterogêneos [...]" (SAWAYA, 2002, p.296).

semelhanças de plataforma, linguagem ou banco de dados, as maneiras de como esses dados estão estruturados são diferentes. Inclusive, existem cadastros que são repetidos em dois ou mais aplicativos, não podendo assim ser compartilhados.

Uma alternativa estudada é a *Arquitetura Orientada a Serviços - Service Oriented Architecture* (SOA). SOA pode ser conceituada como "um estilo de arquitetura de sistemas de informação que habilita a criação de aplicações que são construídas com a combinação de acoplamento leve com serviços intercambiáveis." (SANTOS, 2007, p. 3).

Josuttis (2008) afirma que SOA é baseada em três conceitos técnicos principais: serviços, interoperabilidade através de um barramento corporativo de serviços (*Enterprise Service Bus* - ESB) e acoplamento fraco.

Finalmente, pretende-se com este trabalho, contribuir na forma de material teórico e prático sobre os temas abordados, que são de um referencial bibliográfico um tanto escasso. No meio acadêmico o assunto é novo e uma desafiadora fonte de pesquisa. Além do que, este estudo pode ser de relevante valia às empresas desenvolvedoras de software, ajudando a minimizar os investimentos e esforços para realizar a integração entre softwares legados. Pode ainda contribuir na promoção do reuso e compartilhamento de componentes, combinando recursos de maneira rápida, dando uma resposta ágil a uma exigência de mercado, possibilitando a elas a obtenção de uma vantagem estratégica. Dentro desse contexto, têm-se como exemplo, sistemas de folha de pagamento e controle de freqüência de funcionários (ponto eletrônico) que podem ser integrados de forma mais eficiente.

Assim, para facilitar a compreensão, o trabalho foi dividido em seis capítulos. O capítulo 1 aborda a tecnologia SOA. O capítulo 2 esmiúça as funcionalidades de um ESB, conceitua e descreve suas características. O objetivo do capítulo 3 é descrever algumas outras formas de integração entre softwares aplicativos, além de vantagens e desvantagens de cada uma delas. Já o capítulo 4 mostra estudos já realizados sobre SOA e ESB, relatando os resultados obtidos pelos mesmos, bem como os desafios e dificuldades encontrados durante as suas realizações. Como forma de identificar o principal problema enfrentado pela empresa escolhida, o capítulo 5 apresenta os processos atuais de integração utilizados por ela. Finalmente, o capítulo 6 promove um resumo dos demais capítulos, descreve dificuldades encontradas na realização deste trabalho e dá um encaminhamento para a realização do Trabalho de Conclusão II.

1 SERVICE ORIENTED ARCHITECTURE (SOA)

O presente capítulo tem como objetivo apresentar os principais conceitos associados à *Service Oriented Architecture* (SOA), suas características e importância para as organizações. Além disso, apontar em quais situações essa tecnologia pode ser empregada e relacioná-la com o contexto do problema apresentado anteriormente. E ainda, referir o seu uso por meio do *Enterprise Service Bus* (ESB). Cabe salientar que, devido à escassez de materiais formais e reconhecidos sobre o assunto, a obra (JOSUTTIS, 2008) foi a principal fonte de consulta.

1.1 CONCEITO

Josuttis (2008) escreve que encontrar uma definição exata do termo SOA é difícil. Isto porque existem muitas definições diferentes, porém todas concordam que SOA é um paradigma para melhorar a flexibilidade. A seguir são apresentados alguns conceitos de SOA.

Em (SANTOS, 2007, p.3), verifica-se que SOA pode ser conceituada como "um estilo de arquitetura de sistemas de informação que habilita a criação de aplicações que são construídas com a combinação de acoplamento leve com serviços intercambiáveis.".

Já Hurwitz (2007 apud BENEDETE JUNIOR, 2007, p.7) se refere a SOA, afirmando que

> "é uma arquitetura de software voltada para a construção de aplicações que implementam processos de negócio ou serviços utilizando um conjunto de componentes "caixa-preta", fracamente acoplados, e orquestrados para prover um nível de serviço bem definido".

Uma terceira definição encontrada é a dada por Bierberstein (2006 apud BENEDETE JUNIOR, 2007, p.7):

> "Arquitetura orientada a serviços é uma estrutura (framework) para integrar processos de negócio e a infra-estrutura de TI que o suporta, na forma de componentes seguros e padronizados – serviços – que podem ser reutilizados e combinados para endereçar as mudanças de prioridade do negócio".

Inclusive, para Josuttis (2008), SOA não é uma arquitetura concreta: é algo que conduz a uma arquitetura concreta. Pode-se chamá-la de estilo, paradigma, conceito,

perspectiva, filosofia ou representação. SOA não é uma ferramenta ou *framework*[4] que se possa comprar. É uma abordagem, uma maneira de pensar, um sistema de valores que leva a certas decisões concretas quando se projeta uma concreta arquitetura de software.

1.2 CARACTERÍSTICAS

Santos (2008, p.3) descreve algumas características de SOA:

> "Em um ambiente SOA, recursos em uma rede são disponibilizados como serviços independentes que podem ser acessados sem o conhecimento de como foi implementado internamente. [...] A chave é a independência de serviços que definem interfaces que podem ser chamada para executar tarefas específicas por meio de um padrão, sem que o serviço tenha conhecimento da aplicação solicitante, e sem que a aplicação tenha conhecimento de como o serviço executará a tarefa".

Por outro lado, Josuttis (2008, p.12) cita a flexibilidade como principal característica esperada por SOA, quando afirma que "a principal razão para usar SOA é que ela deve ajudá-lo em seu negócio. Por exemplo, no caso de precisar de soluções de TI que guardem e gerenciem os seus dados e permitam automatizar os processos comuns que lidam com esses dados".

1.3 CONCEITOS TÉCNICOS

A fim de melhor compreender SOA, devem-se observar seus principais conceitos:

1.3.1 Serviços

Um dos principais objetivos de SOA é basear os sistemas na abstração das regras e funções de negócio.

Em (JOSUTTIS, 2008) tem-se que, na essência, um serviço é uma representação da TI de alguma funcionalidade do negócio. Ou seja, serviço é a representação de uma abstração.

[4] Conjunto de classes que colaboram para realizar uma responsabilidade para um domínio de um subsistema da aplicação (FAYAD, 1997).

O principal objetivo de um serviço é representar um passo natural da funcionalidade do negócio. Dependendo do domínio para qual ele é fornecido, deve representar uma funcionalidade independente que corresponde a uma atividade de negócio do mundo real, como por exemplo, ‘criar um cliente’, ‘obter contratos de um cliente’, ‘transferir dinheiro’.

Note que, nesse contexto, a tecnologia usada para prover o serviço (um software rodando como serviço – *service*- no sistema operacional), não faz parte dessa definição.

De acordo com Josuttis (2008), os serviços podem ser classificados em três categorias:

- Serviços básicos – primeiro estágio de expansão (SOA fundamental), que provê apenas serviços correspondentes a funcionalidades básicas de negócio;
- Serviços compostos – segundo estágio de expansão (SOA federativa), os quais representam serviços compostos por outros serviços. Na terminologia SOA, isso se chama orquestração;
- Serviços de processos – o terceiro estágio de expansão (SOA habilitada para processos) representa os *workflows* ou processos de negócio de longo prazo. Diferente dos serviços básicos e compostos, um serviço de processo tem um estado que se mantém estável durante múltiplas chamadas. Um exemplo disso é o carrinho de compras localizado nos sites de compras.

A Figura 1.1 ilustra essa classificação.

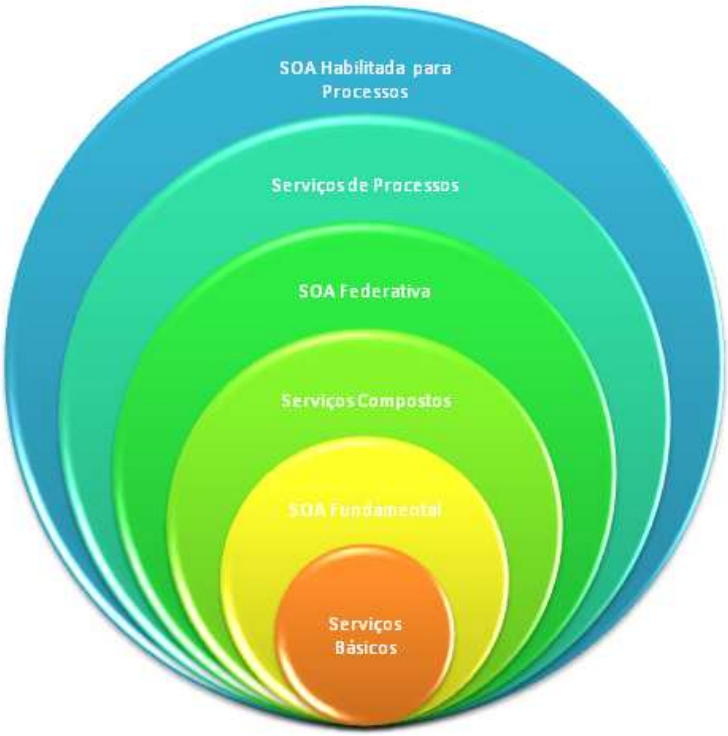

Figura 1.1 – Camadas básicas de serviços e estágios de expansão SOA
Fonte: JOSUTTIS, 2008, p.56 (adaptado pelo autor)

1.3.2 Alta interoperabilidade

Tem-se em (JOSUTTIS, 2008), que diferentes sistemas utilizam plataformas diferentes, linguagens e paradigmas de programação diferentes, assim como diferentes bancos de dados e motores de regras, ou seja, eles são heterogêneos. Uma das características mais fortes de SOA é a aceitação dessa heterogeneidade – ela lida e suporta essa propriedade.

Conforme Josuttis (2008), com os sistemas heterogêneos, o primeiro objetivo é ser capaz de conectar esses sistemas facilmente. Isso é normalmente chamado de alta interoperabilidade. Para SOA isso é o começo e não o fim. É a base a partir da qual as funcionalidades de negócio (serviços) são implementadas.

1.3.3 Acoplamento fraco

Acoplamento fraco é o conceito de minimizar as dependências. Quando as dependências estão minimizadas, as modificações têm os efeitos minimizados e os sistemas ainda executam quando partes deles estão quebradas ou indisponíveis. Minimizar as dependências contribui para tolerância às falhas e flexibilidade (JOSUTTIS, 2008).

A escalabilidade e a tolerância às falhas são as chaves para a manutenibilidade dos sistemas. Outro objetivo importante é minimizar o impacto das modificações e das falhas dentro do cenário do sistema como um todo. Dessa forma, o acoplamento fraco é um conceito-chave de SOA (JOSUTTIS, 2008).

1.4 IMPACTO NAS ORGANIZAÇÕES

Agilidade é um dos atributos mais importantes das empresas de hoje, pois possibilita aproveitar as oportunidades do mercado antes que a concorrência o faça. Para conseguir essa flexibilidade, uma possível abordagem é a empresa analisar sua operação de negócio como um conjunto de funções interconectadas (processos e serviços). As funções chaves para o negócio devem ser identificadas e trabalhadas, de forma a se tornarem um diferencial positivo no mercado (BENEDETE JUNIOR, 2007).

Segundo Benedete Junior (2007, p.17),

> "Através de sua habilidade de escalar e evoluir, SOA habilita um portfólio de TI que se adapta às várias necessidades de um domínio específico de problema ou arquitetura de processo. Portanto SOA pode prover uma sólida fundação para a agilidade e adaptabilidade do negócio".

Sendo assim, SOA permite que as empresas obtenham melhores resultados por torná-las focadas em suas competências chaves; ágeis em atender as demandas dos clientes; resilientes aos impactos das mudanças no negócio ou nas tecnologias; e integradas com seus clientes, parceiros e fornecedores (BENEDETE JUNIOR, 2007).

1.4.1 Departamentos

As estruturas dominantes das empresas são os departamentos, que mantêm sistemas específicos que cresceram ao longo dos anos (JOSUTTIS, 2008). Isso implica que, quando essas empresas necessitam de uma nova funcionalidade em algum desses sistemas, entregam a tarefa ao departamento correspondente. Por sua vez, esse departamento implementa a nova funcionalidade no seu sistema específico, a qual fica restrita ao mesmo.

Hoje em dia, a integração e a distribuição estão se tornando cada vez mais importantes. Então, embora as empresas ainda tenham estruturas monolíticas orientadas a departamentos, os departamentos e os sistemas estão começando a trabalhar juntos (JOSUTTIS, 2008).

1.4.2 Gerenciamento

Por impactar em diversos departamentos, unidades de negócio ou empresas, o desenvolvimento de qualquer nova funcionalidade deve ser tratado como um projeto. Por esse motivo, um papel fundamental em SOA é o de gerente da solução (ou do projeto). Esse gerente tem como competência inicial, criar uma modelagem de alto nível para determinar o impacto nos sistemas existentes (JOSUTTIS, 2008).

O suporte do gerente da solução é também necessário durante a realização da solução, porque na prática as coisas sempre se mostram diferentes do esperado. O perigo com

as soluções distribuídas também é que qualquer sistema pode quebrar toda a solução (JOSUTTIS, 2008).

1.4.3 Colaboração

Pelo fato de envolver diferentes departamentos (unidades de negócio ou empresas), a colaboração é apontada com um requisito chave. De acordo com Josuttis (2008, p.91),

> "desde a formulação de uma idéia até a manutenção da sua realização, os sistemas distribuídos requerem a colaboração. É claro que a colaboração é difícil nas organizações que consistem de departamentos que se comportam como territórios isolados. Nestes casos, mudar a cultura da empresa será o fator chave para o sucesso de SOA".

1.5 DESEMPENHO

Os sistemas de TI têm dois aspectos que repetidamente quebram planos, conceitos e modelagens: desempenho e segurança (JOSUTTIS, 2008).

Josuttis (2008, p.137) evidencia o a necessidade da observância do desempenho, quando afirma que "há diferentes lugares onde o desempenho entra em jogo no contexto de serviços e infra-estrutura de SOA. Se o desempenho se torna crítico, isso é normalmente por causa do tempo de execução de um serviço".

Uma vez que SOA é um conceito para cenários de sistemas heterogêneos, não se pode simplesmente enviar dados binários de um sistema a outro, pelo motivo que o código compilado de uma aplicação não combina com o de outra. Por este motivo, obrigatoriamente, é necessário o uso de um protocolo intermediário conhecido tanto pelo consumidor[5] quanto pelo fornecedor[6].

Um formato possível é o baseado em XML[7]. Porém, de acordo com (JOSUTTIS, 2008, p.138), esse formato pode ser um problema: "XML gera muita informação e o resultado

[5] Sistema que chama um serviço (usa um serviço fornecido) (JOSUTTIS, 2008).
[6] Sistema que implementa um serviço (uma funcionalidade de negócio) de tal forma que outros sistemas possam chamá-lo (JOSUTTIS, 2008).
[7] eXtensible Markup Language. Uma notação de propósito geral e de leitura fácil para humanos amplamente usada para a descrição e troca de dados (JOSUTTIS, 2008).

disso é que os dados essenciais podem ser aumentados de 4 a 20 vezes. Isto pode ter um impacto significativo na banda de rede em ambientes onde a mesma é limitada".

A adoção de um protocolo intermediário, baseado na serialização e desserialização, pode igualmente ser crítico, considerando-se o tempo que o processo de transformar uma sequência de bytes em dados complexos pode levar.

De qualquer maneira, seja qual for o protocolo utilizado, de acordo com Josuttis (2008, p.139), "é uma boa idéia colecionar as estatísticas sobre os tempos de execução de diferentes requisições de serviço e monitorar os números ao longo do tempo. Isso pode ajudá-lo a identificar os problemas mais imediatos e as tendências futuras".

1.5.1 Reusabilidade

A reusabilidade, apontada como virtude de SOA pode acabar por se tornar um problema de desempenho. Josuttis (2008) ilustra um exemplo de um sistema (Figuras 1.2 e 1.3) que fornecia um serviço responsável por retornar todos os dados do cliente - o que correspondia a centenas de atributos. Certa feita surgiu um novo sistema que necessitava de apenas alguns dados dos clientes. A fim de se atender ao requisito de usabilidade, optou-se pelo reuso do serviço existente (que retornava todos os dados do cliente).

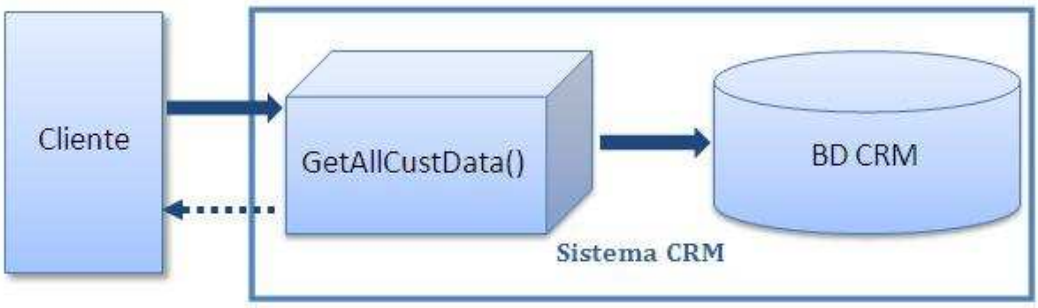

Figura 1.2 – Carregar todos os dados do cliente leva tempo
Fonte: JOSUTTIS, 2008, p.143

Ocorre que, imediatamente, o novo consumidor reclamou da demora no atendimento de sua solicitação. É claro que processar um resultado que retornava apenas os atributos desejados seria muito mais rápido do que a resposta usual que continha o portfólio completo do cliente.

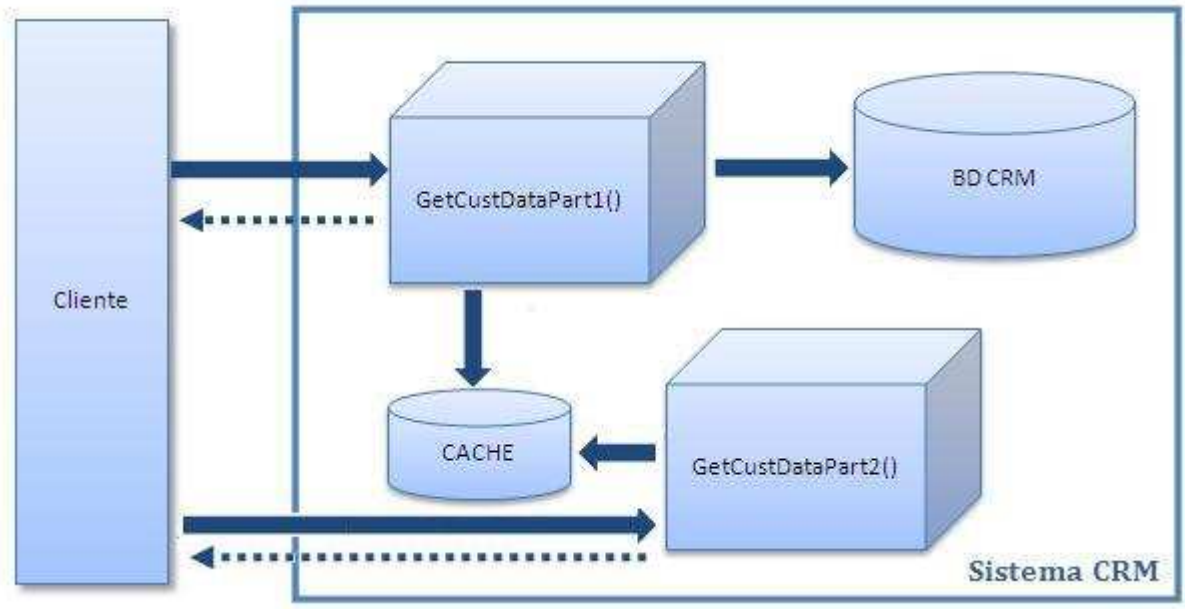

Figura 1.3 – Carregar dados de cliente em dois passos
Fonte: JOSUTTIS, 2008, p.144

Mas isso não significa que o objetivo de reusabilidade nunca possa ser atingido (JOSUTTIS, 2008).

1.5.2 Customização

O exemplo citado expõe o fato de que “sistemas diferentes freqüentemente precisam de visualizações diferentes dos mesmos dados, o que leva à criação de diferentes serviços quando o desempenho entra em jogo” (JOSUTTIS, 2008, p.143). Sendo assim, às vezes, é necessário criar serviços específicos para consumidores específicos, ou seja, customizar os serviços.

1.6 SEGURANÇA

Outro fator importante a ser considerado em uma implantação SOA é a segurança. Josuttis (2008) define a seguinte categorização para os aspectos de segurança para SOA:

- Autenticação – verificar identidades de usuários, dispositivos ou serviços externos;

- Autorização – controlar restrições e permissões para uma determinada identidade;
- Confidencialidade – assegurar que ninguém, além do fornecedor e consumidor, tenha acesso aos dados durante sua transferência;
- Integridade – garantir que os dados não sejam manipulados ou falsificados, ou que contenham erros;
- Disponibilidade – manter o sistema sempre disponível;
- Contabilização – monitorar chamadas de serviço para gerenciamento, planejamento ou outros propósitos;
- Auditoria – monitorar e rastrear todo o fluxo de dados relevantes para a segurança, seja para melhorar a sua confiabilidade, seja para atender a determinações legais.

1.6.1 Requisitos de segurança

A fim de atender os requisitos de autenticação e autorização “você normalmente precisa do conceito de ID de usuário e senha. Para IDs de usuário normalmente existe alguma forma indireta para que seja possível atribuir papéis a esses usuários freqüentemente chamados de fornecedor de identidade” (JOSUTTIS, 2008, p.150).

Para confidencialidade e integridade, os conceitos comuns como criptografia e assinaturas digitais são usados.

1.7 WEB SERVICES

Sampaio (2007, p.139), define um *Web Service* da seguinte forma:

> “um Web Service é um aplicativo servidor que disponibiliza um ou mais serviços para seus clientes de maneira fracamente acoplada. [...] expõe sua interface para os usuários usando um documento XML conhecido como Web Services Description Language ou WSDL”.

Embora, num primeiro momento, quando se ouve falar em SOA se faça uma relação direta com *Web Service*, deve-se entender que ele é apenas uma forma de implementação dessa arquitetura. Josuttis (2008, p.30), explica isso quando afirma que

> "o uso de *Web Services* não significa automaticamente que está implementando SOA. Isso pode ser apenas uma maneira para deixar alguns sistemas interagirem, sem todos os outros aspectos fornecidos e exigidos por SOA (JOSUTTIS, 2008, p.196)".

No capítulo 2, de título "*Enterprise Service Bus* – ESB", encontra-se uma justificativa para a não adoção do W*eb Service* como meio de implementação de SOA neste trabalho.

Outros conceitos ou características, citados em (JOSUTTIS, 2008) são governança e maturidade. Governança pode ser definida como uma maneira de "garantir que as pessoas façam o que é certo", ou ainda de "controlar o desenvolvimento e a operação de software". O autor justifica a aplicação da governança em SOA com a finalidade de evitar o risco de caos e de desperdício de dinheiro e recursos. Assim como acontece com o modelo de referência *Capability Maturity Model Integration* (CMMI), SOA também lida com níveis de maturidade. Em (JOSUTTIS, 2008), são descritos os cinco de níveis de maturidade SOA. Para maiores informações sobre esses conceitos, consulte a obra do referido autor.

Finalizando o capítulo, é importante lembrar, principalmente, dos conceitos de acoplamento fraco, heterogeneidade e alta interoperabilidade. Sendo assim, considerando que ESB é parte integrante e fundamental da arquitetura SOA e detentor desses conceitos, é fundamental dedicar um capítulo exclusivo a ele.

2 ENTERPRISE SERVICE BUS (ESB)

A finalidade deste capítulo é minudenciar as funcionalidades de um ESB, conceituar e expor suas características e desafios enfrentados. Assim como no capítulo anterior, este também está alicerçado, principalmente, na obra de Josuttis (2008).

Considerando que *Web Service* não é a única e nem a melhor forma de se implementar SOA, este trabalho segue o modelo de ESB sugerido por Josuttis (2008), que afirma que "para sinergia melhor, considere a implementação de um ESB orientado a API[8]". Para melhor entender esse modelo, leia a seção 2.1.1, intitulada "Tipos de ESB".

A Figura 2.1 mostra como um ESB é posicionado entre as aplicações, de forma a centralizar a comunicação entre as mesmas.

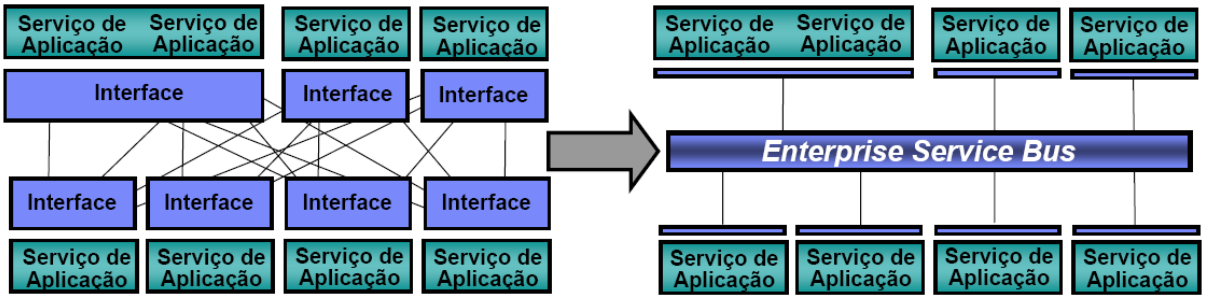

Figura 2.1 – Enterprise Service Bus - ESB
Fonte: VIOLA, 2006 apud BENETE JUNIOR, 2007, p. 33

2.1 CONCEITO

Uma parte de SOA é a infra-estrutura que lhe permite usar os serviços em um ambiente de sistemas produtivos. Isso é comumente chamado de Barramento de Serviços Corporativos (ESB – *Enterprise Service Bus*) (JOSUTTIS, 2008).

É interessante visualizar um conceito atribuído ao ESB à luz de outra área de conhecimento – Recursos Humanos. Silva (2009, p.58) escreve que "o barramento corporativo é uma camada que admite regras especiais, lógica específica e uma grande quantidade de softwares (programas de computador) integrados em padrões pertinentes".

De acordo com Josuttis,

[8] "Application Program Interface - Interface de Programas Aplicativos: conjunto de rotinas que os programas aplicativos utilizam para requisitar e obter os serviços de nível mais baixo oferecidos pelo sistema operacional do computador" (SAWAYA, 2002, p.26).

> “A infra-estrutura é a parte técnica de SOA que possibilita a alta interoperalidade. A infra-estrutura de um cenário SOA é chamada de barramento corporativo de serviços (ESB). Este termo foi tomado de integração das aplicações corporativas onde era chamado de barramento EAI ou simplesmente de barramento corporativo” (JOSUTTIS, 2008, p.17).

Dessa forma, torna-se evidente a clara relação entre SOA e ESB, uma vez que este é o responsável por possibilitar a chamada de serviços entre os sistemas heterogêneos, contribuindo de forma fundamental para a implementação dessa arquitetura.

Comparando as Figuras 2.2 e 2.3, pode-se perceber como a implementação de um ESB pode melhorar a comunicação entre diferentes softwares aplicativos.

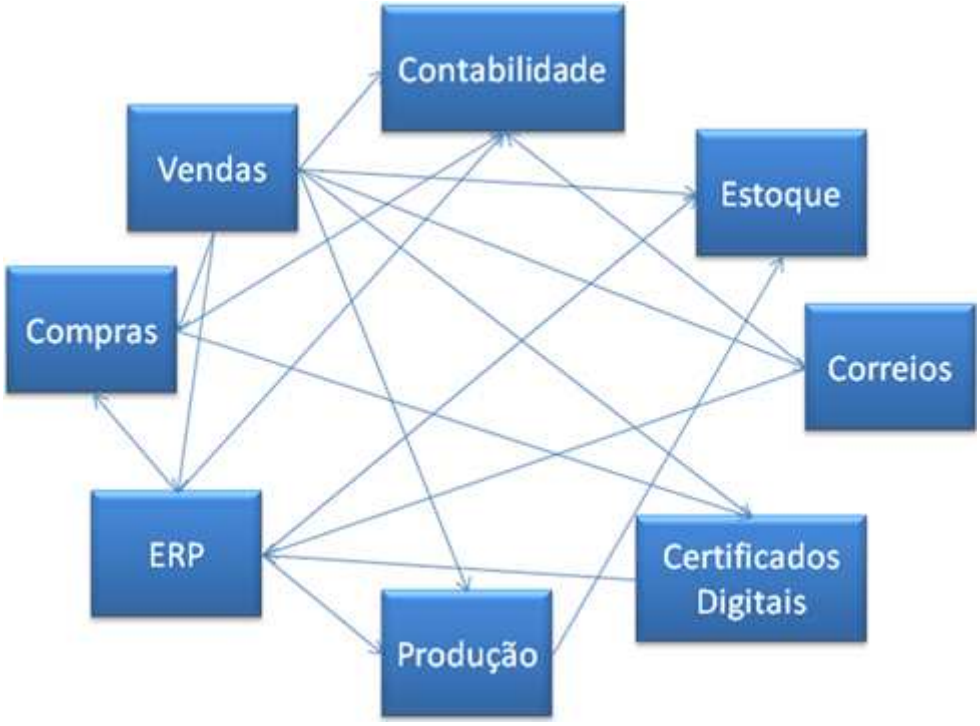

Figura 2.2 – Caos de comunicação
Fonte: MOREIRA, 2008

A Figura 2.2 demonstra que sem a adoção de um ESB, cada um dos softwares aplicativos realiza a troca de dados diretamente conectado ao outro, através de uma forma específica. E, na maioria dos casos, quando um desses softwares deseja se comunicar um terceiro, uma nova forma é adotada.

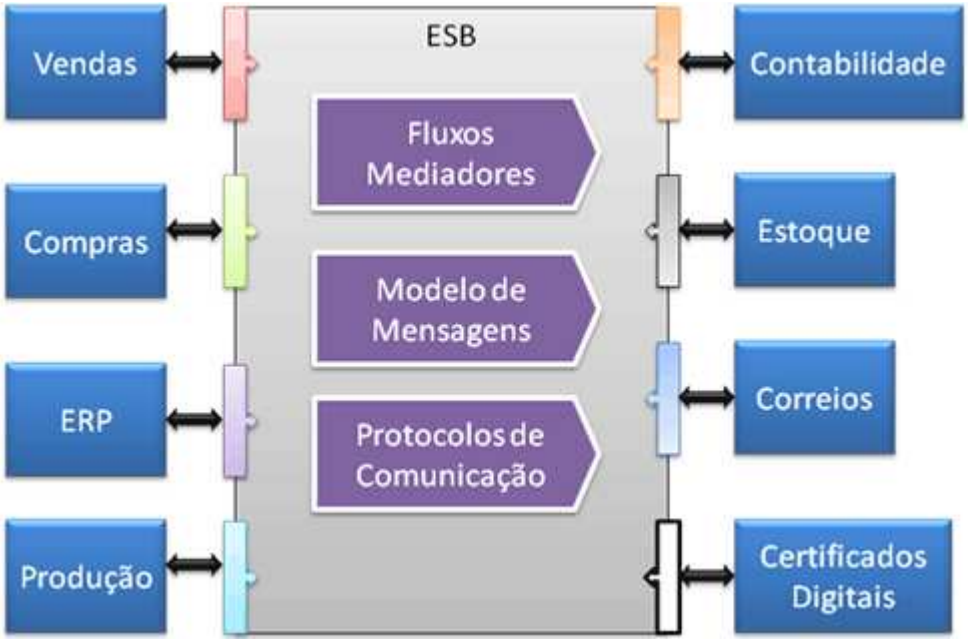

Figura 2.3 – Simplificação na comunicação através de um ESB
Fonte: http://tec.brq.com/conheca-um-pouco-mais-sobre-esb-enterprise-service-bus-e-as-ferramentas-ibm/

Já a Figura 2.3 demonstra como a comunicação pode ser simplificada quando centralizada pelo barramento, podendo ser realizada de forma transparente no que diz respeito a protocolo. Isso pode ser mais bem entendido na sequência do texto.

2.1.1 Tipos de ESB

Para Josuttis

> "Conceitualmente, duas abordagens diferentes são possíveis considerando-se onde a responsabilidade de um ESB começa do ponto de vista dos consumidores e dos fornecedores. [...] abordagem orientada a protocolo[...] abordagem orientada a API [...]" (JOSUTTIS, 2008, p.48).

Na abordagem orientada a protocolo, consumidores e fornecedores enviam e recebem as mensagens de acordo com o protocolo estabelecido pelo ESB. Um exemplo disso são os *Web Services* que requerem um protocolo SOAP[9] (JOSUTTIS, 2008). Cada participante pode escolher as ferramentas que vai utilizar para atender o protocolo que foi definido.

Já na abordagem orientada a API, o ESB prevê que podem ser utilizadas as APIs específicas de cada plataforma, para que os consumidores e fornecedores implementem as

[9] O *Simple Object Access Protocol* (SOAP) é uma tecnologia que permite distribuir objetos pela Internet (DEITEL, 2003).

chamadas de serviço (JOSUTTIS, 2008). Diferentemente da abordagem anterior, a equipe que mantêm o ESB é responsável pela disponibilidade de formas de envio e recebimento das mensagens, baseadas nas APIs da plataforma utilizada.

2.2 FUNCIONALIDADES

A principal responsabilidade ou funcionalidade de um ESB é possibilitar aos consumidores chamarem os serviços oferecidos pelos fornecedores (JOSUTTIS, 2008). Além disso, é possível listar as funcionalidades a seguir:

2.2.1 Conectividade

Um ESB deve ser capaz, de forma síncrona ou assíncrona, prover a conectividade entre os softwares aplicativos (fornecedor e consumidor) envolvidos.

2.2.2 Repositório

Manter arquivos ou dados em um diretório comum (unidade de disco local ou de rede) ao consumidor e fornecedor, é outra tarefa que deve ser oferecida por um ESB.

2.2.3 Transformação de dados

Talvez a mais importante das funcionalidades de um ESB seja a transformação de dados. Segundo Chappell,

> “A transformação de dados é inerentemente parte do barramento em uma distribuição ESB. Os serviços de transformação que são especializados para as necessidades das aplicações individuais ligadas no barramento podem ser localizados em qualquer lugar e acessíveis de qualquer lugar do barramento. Pelo fato de a transformação de dados ser uma parte integrante de ESB, um ESB pode ser

> imaginado como sendo uma resolução de desencontros de impedância entre as aplicações" (2004 apud Josuttis, 2008, p.43).

2.2.4 Roteamento inteligente

Outra tarefa fundamental de ESB é o roteamento. Deve existir uma forma para que o consumidor envie uma mensagem ao fornecedor e receba uma resposta (JOSUTTIS, 2008).

De acordo com o conteúdo das mensagens, o ESB deve ter a capacidade de proporcionar que sejam atribuídas prioridades diferentes a cada uma delas, ou ainda, que se tomem diferentes atitudes (JOSUTTIS, 2008).

2.3 DESAFIOS

Durante a implementação de um barramento de serviços, dentro de uma abordagem SOA, alguns desafios precisam ser superados:

Deve haver um entendimento de que SOA é muito mais do que apenas uma infra-estrutura - uma cultura organizacional de colaboração deve ser criada; suporte gerencial é fundamental para o sucesso da implementação; deve-se introduzir SOA de maneira gradativa, garantindo um aprendizado.

Além desses, ainda destacam-se outros três desafios: complexidade, segurança e confidencialidade. Qualquer forma de acoplamento fraco aumenta a complexidade. Além do que, o esforço requerido para depuração e testes de sistemas distribuídos também é bem maior (JOSUTTIS, 2008). Em SOA, substitui-se as soluções individuais de conexão de cada par de sistemas por um ESB comum, afim de que cada sistema conectado ao ESB esteja conectado a todos os outros sistemas, por padrão esses sistemas estão desprotegidos (JOSUTTIS, 2008). A fim de garantir a confidencialidade dos dados transferidos, pode-e aplicar recursos de criptografia, tanto na camada de transporte, quanto na camada de mensagens.

A fim de justificar a escolha pela integração através de um ESB ao invés dos métodos convencionais, o próximo capítulo explora outras formas de se realizar a integração de softwares.

3 FORMAS ALTERNATIVAS DE INTEGRAÇÃO ENTRE SOFTWARES APLICATIVOS

Como existem outras formas de integração de softwares aplicativos, neste capítulo são arroladas algumas delas, a fim de se poder relacionar com o modo utilizado atualmente, pela empresa escolhida para este trabalho. E ainda são elencadas vantagens e desvantagens desses modos de integração, comparadas às utilizadas pelo ESB.

3.1 INTEGRAÇÃO PONTO A PONTO

No início, os aplicativos eram independentes, e não tinham nenhuma forma de comunicação com outros aplicativos. Logo as empresas sentiram a dificuldade de atualizar ou substituir esses aplicativos, e a capacidade de compartilhar passou a ser desejável (SPACKMAN e SPEAKER, 2006).

A principal forma de integração de dados, realizada ponto a ponto, dá-se através da troca de informações por arquivos texto.

> “A simplicidade dos arquivos texto fez deles uma escolha popular em diversas situações. De fato, um arquivo texto freqüentemente é a estrutura subjacente usada na implementação de arquivos seqüenciais mais elaborados, como um arquivo de funcionários” (BROOKSHEAR, 2005, p.321).

Um exemplo de integração de dados através da troca de arquivos texto pode ser visto na Figura 3.1, onde arquivos são utilizados para integrar diferentes aplicativos, desde a indústria, passando pela empresa distribuidora, até o varejo.

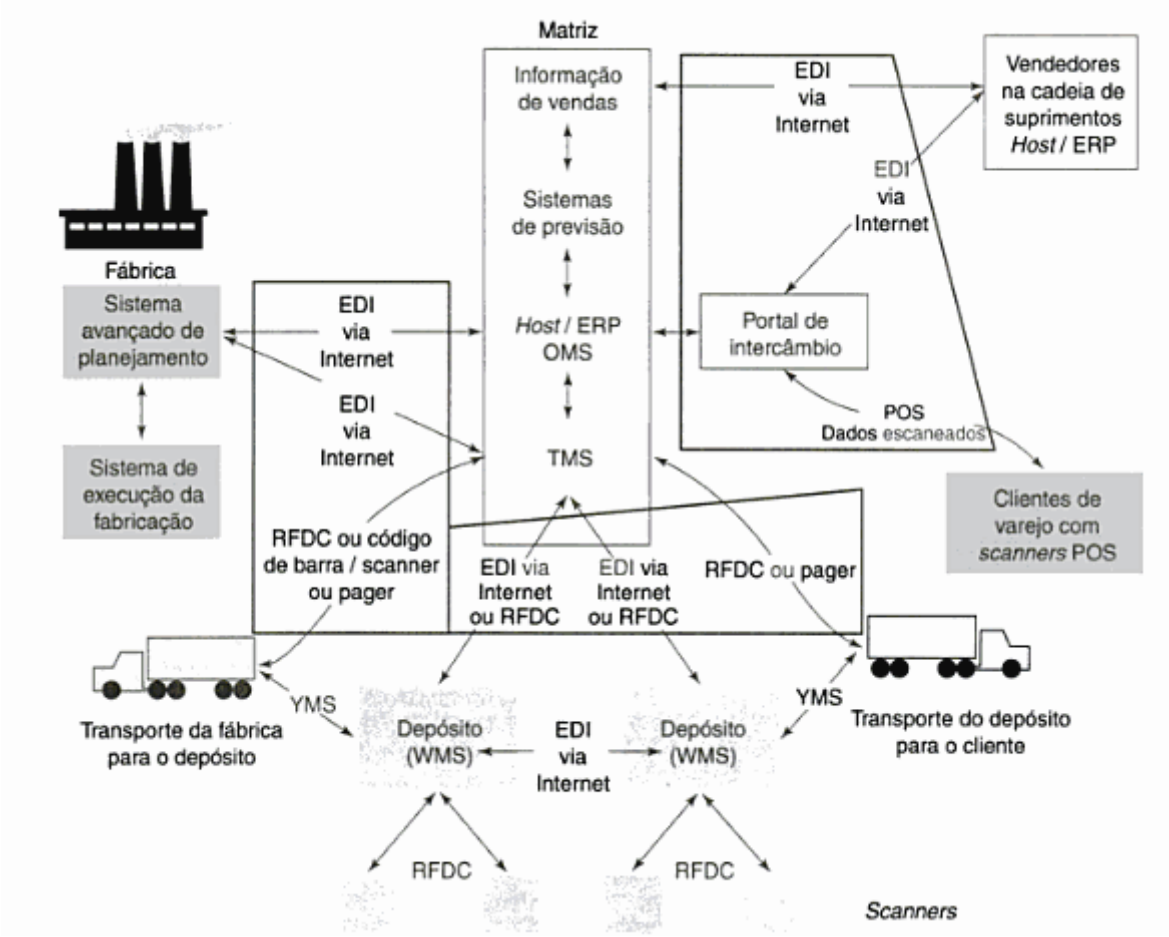

Figura 3.1 – Troca de dados por arquivo texto
Fonte: BOWERSOX et al, 2006, p.176

Como alternativa ao arquivo texto, tem-se utilizado arquivos XML. Cummins (2002, p. 10) conceitua um arquivo XML da seguinte forma:

> "Fundamentalmente, a XML possui um formato de dados com tags. Um documento XML contém vários elementos, cada qual com uma tag descritiva e um valor associado, todos expressos como caracteres. Caracteres especiais delimitam os elementos, de modo que os documentos e seus nomes de elemento e valores têm tamanho variável".

3.2 SOLUÇÕES DE PONTO CENTRAL

Seguindo a evolução da integração, o próximo passo foi a adoção da técnica de ponto central – "uma versão modificada da integração ponto a ponto, na qual um ponto central era responsável por direcionar os dados que estavam sendo passadas entre os aplicativos" (SPACKMAN e SPEAKER, 2006, p.29).

3.2.1 Troca de mensagens

Uma forma utilizada para realizar a integração entre sistemas, através de um ponto central, é oferecida pela Integração de Aplicações Corporativas – *Enterprise Application Integration* (EAI) e apresentada na obra de Cummins (2002).

A proposta da EAI, apesar de ser muito parecida com a de um ESB, não é herdada de do mesmo conceito. Enquanto ESB é parte inerente de SOA, EAI não está contextualizada dentro de uma arquitetura específica, ou seja, é um modo isolado de se realizar uma integração entre sistemas.

Na abordagem da EAI, basicamente, um emissor envia uma mensagem para uma fila para transmissão, onde o gerente da fila a encaminha ao destino. No destino, essa mensagem é retida até que o destinatário esteja pronto para recebê-la (CUMMINS, 2002).

Assim como em um ESB, essa conexão é realizada por meio de um acoplamento fraco, uma vez que o destinatário, necessariamente, não precisa estar apto a receber a mensagem no momento em que ela é gerada pelo emissor.

A Figura 3.2 ilustra a ação de um intermediário de mensagens, utilizado pela EAI.

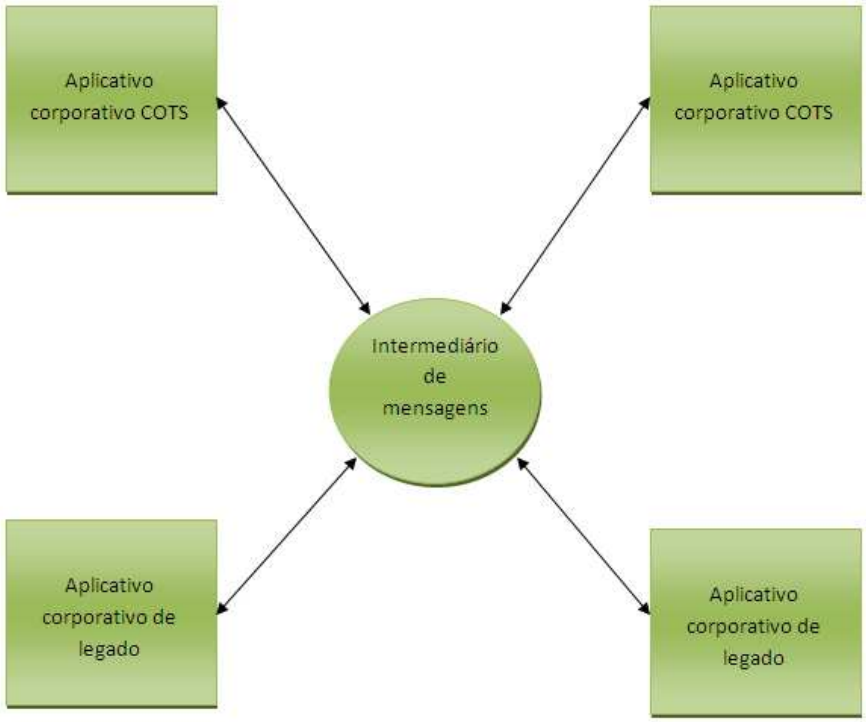

Figura 3.2 – Integração de aplicativos corporativos
Fonte: CUMMINS, 2002, p.7

3.2.2 Armazenamento compartilhado

A necessidade de integração levou, primeiramente, ao compartilhamento dos dados dos aplicativos através de um banco de dados em comum. De acordo com Spackman e Speaker (2006, p.28),

> "Sempre que uma empresa precisava compartilhar dados entre aplicativos, ela simplesmente adicionava um novo trecho de código ou um modo de visualização de bancos de dados, até que o número de aplicativos e pontos de integração finalmente tornou gerenciamento da integração praticamente impossível. Os aplicativos se tornaram tão fortemente integrados que uma pequena alteração em um trecho de código levava a uma cadeia complexa de alterações adicionais, apenas para preservar a integração, tornando o processo como um todo extremamente frágil e dispendioso".

Em (BROOKSHEAR, 2005, p. 339) verifica-se que, "historicamente, os bancos de dados evoluíram como forma de integrar sistemas de armazenamento de dados [...] os sistemas de banco de dados emergiram como meios de consolidar a informação armazenada e mantida por uma organização", conforme ilustrado na Figura 3.3.

Figura 3.3 – Organização de um banco de dados
Fonte: BROOKSHEAR, 2005, p.339

Além do banco de dados, pode-se ainda, realizar uma integração, mantendo os dados em um *Data Warehouse*[10]. Em relação aos dados guardados em um *Data Wahrehouse*, Rosini e Palmisano (2003, p. 34) especificam que "devem ser padronizados quanto ao nome que terão nesse ambiente - é o conceito de Dicionário de Dados -, em que cada entidade na empresa é conhecida por um único nome e esse será padrão para todos os sistemas". A Figura 3.4 mostra isso.

[10] "Uma coleção de dados orientada por assuntos, integrada, variante no tempo, e não volátil, cujo objetivo é dar suporte aos processos de tomada de decisão" (INMON, 1992 apud ROSINI e PALMISANO, 2003).

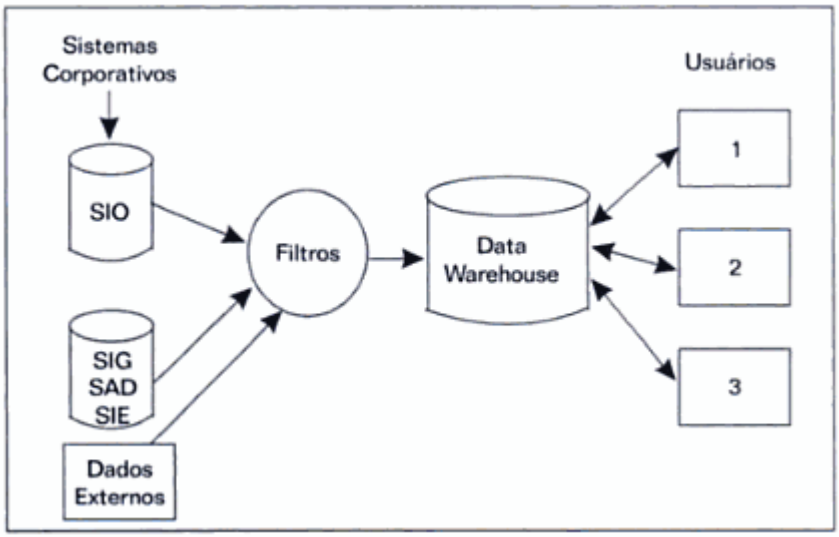

Figura 3.4 – Arquitetura simplificada de um *Data Warehouse*
Fonte: ROSINI e PALMISANO, 2003, p.35

3.3 OUTRAS FORMAS DE INTEGRAÇÃO

Oportunamente, compete ainda neste estudo citar outras formas, mesmo que ainda não tão difundidas, de proceder-se uma integração de sistemas.

Três diferentes formas de integração são relacionadas em (CUMMINS, 2002):

- Gerenciamento de fluxo de trabalho – um processo de fluxo de trabalho gerencia a execução de atividades, que podem realizar diretamente operações limitadas ou invocar outros aplicativos. Pode variar do processamento *batch* ao processamento baseado em eventos, onde as transações comerciais são processadas à medida que ocorrem;
- Objetos distribuídos – essa tecnologia oferece suporte ao desenvolvimento de sistemas através de componentes distribuídos, os quais interagem como objetos, trocando mensagens em uma rede. Esses objetos podem ser serviços ou objetos compartilhados de uma aplicação de negócio;
- Tecnologia de agentes – programa autônomo que capta o ambiente e diante disso reage de acordo com regras internas de operação. Diferentemente de uma aplicação convencional, o próprio agente determina quando e de que maneira executará uma função. Podem ser usados para direcionar fluxos de trabalho.

O Quadro 3.1, baseado nas formas de integração relacionadas neste estudo, identifica vantagens e desvantagens na aplicação de cada uma delas.

Forma	Vantagens	Desvantagens
Arquivos Texto	Simplicidade	Falta de segurança da informação
Arquivos XML	Identificação dos elementos	Tamanho do conteúdo, em virtude da duplicidade de texto
Troca de Mensagens	Acoplamento fraco	Necessidade de um centralizador de mensagens
Banco de Dados	Alta disponibilidade e performance	Entidades e atributos devem ser conhecidos pelas aplicações
Data Warehouse	Segurança	Necessidade de um dicionário de dados
Gerenciamento de Fluxo de Dados	Mapeamento dos processos	Falta de interoperabilidade
Objetos Distribuídos	Escalonabilidade	Homogeneização de plataforma
Tecnologia de Agentes	Autonomia	Difícil implementação

Quadro 3.1 – Comparativo entre as formas de integração
Fonte: do autor

Com a finalidade de se verificar o estudo e a aplicação dessas formas de integração, bem como daquela que está sendo proposta, faz-se necessária uma pesquisa sobre trabalhos correlatos, objeto do próximo capítulo.

4 TRABALHOS CORRELATOS

No intuito de auxiliar no estabelecimento de parâmetros para a modelagem do ESB, são estudados trabalhos já realizados sobre os assuntos expostos. Dessa forma, são analisados os resultados obtidos, além de desafios e dificuldades encontrados durante as suas realizações.

Primeiramente, foi realizada uma pesquisa sobre trabalhos acadêmicos, tanto em nível de graduação quanto de pós-graduação (mestrado e doutorado), que tivessem uma proposta similar a deste trabalho. A pesquisa não obteve êxito. Como alternativa, optou-se então, por realizar uma pesquisa sobre trabalhos que tivessem uma fundamentação parecida.

Foram localizados trabalhos em diferentes níveis acadêmicos, fundamentados na arquitetura SOA e que, diferentemente deste trabalho, não implementaram, mas utilizaram um ESB, ou em determinado momento deram ênfase a essa tecnologia.

Comercialmente, foram localizadas diferentes implementações de ESB, tanto em softwares proprietários, como em softwares de código aberto.

A seguir, encontra-se uma breve exposição de alguns dos resultados encontrados.

4.1 NÍVEL ACADÊMICO

Como resultados da pesquisa, a nível acadêmico, nesta seção são expostos trabalhos em diferentes graus (graduação e pós-graduação).

4.1.1 Graduação

Foi localizado um trabalho apresentado à Universidade Federal de Itajubá – UNIFEI, de Itajubá – MG (GARCIA et al, 2008) que tem como justificativa a rápida evolução das tecnologias utilizadas para o desenvolvimento de software e a falta de padronização nesse desenvolvimento, especialmente nos sistemas utilizados na área médica. Como proposta, foi apresentada uma solução para a integração de diferentes softwares médicos, que trabalham com laudos descritos em forma de texto e com imagens obtidas em exames médicos como tomografias e ressonâncias magnéticas.

Na implementação, foram utilizados *Web Services* como barramento, para atender aos serviços de consulta de pacientes e exames.

Como resultado, foi publicado que, com a organização das funcionalidades dos sistemas médicos como serviços, é alcançada a robustez que dá suporte à escalabilidade dos mesmos. E ainda, essa organização permite que se reúnam dados do paciente que estão distribuídos em sistemas distintos para obtenção de um prontuário completo.

Sobre SOA, o estudo concluiu que "é uma tecnologia adequada para a integração de ambientes heterogêneos como os ambientes da área médica", e que ela "introduz diretrizes para guiar o desenvolvimento de futuras integrações entre outros sistemas médicos".

Ainda na graduação, foi encontrado um trabalho que está em fase de conclusão na Universidade Federal de Santa Catarina (MENEZES e GALOPPINI, 2009). Ele tem como objetivo "construir um suporte para integração de sistemas utilizando Enterprise Service Bus (ESB), que permita a interação de sistemas legados com provedores de serviços construídos com base em uma arquitetura orientada a serviços (SOA), provendo mecanismos para transformação de dados, conectividade e tratamento de erros". Apesar de tratar sobre a integração de sistemas através de um ESB, ele não propõe a modelagem e/ou desenvolvimento de um barramento, mas sim de um *framework* para aplicação de um protocolo específico, através do uso de um barramento já existente.

4.1.2 Especialização

Benedete Junior (2007) apresentou uma monografia à Escola Politécnica da Universidade de São Paulo, a fim de obter o título de MBA[11] em Tecnologia da Informação. O título do trabalho é *Roteiro para definição de uma arquitetura SOA utilizando BPM*[12].

A motivação do trabalho está baseada na necessidade das melhorias de processos de negócio e da comunicação com a área de tecnologia da informação. Ele propõe a utilização de SOA e BPM a fim de suprir tais necessidades.

[11] Mestrado em Administração de Empresas (Especialização ou Pós-graduação Lato Sensu).

[12] Gestão de Processos de Negócio (BPM), "visto como uma disciplina de gestão, é a habilidade de continuamente otimizar aqueles processos operacionais que são mais diretamente relacionados à obtenção dos objetivos da corporação" (BENEDETE JUNIOR, 2007, p.13).

Apesar de apenas fazer referência ao ESB, o trabalho explora a aplicação de SOA de uma forma geral, em conjunto com o mapeamento dos processos de negócio, tendo sido de grande valia para este estudo, servindo inclusive como referência bibliográfica.

O autor, no capítulo 2, demonstra em uma tabela os desafios apresentados e como SOA pode superá-los. Além do que, em todo o trabalho ele procurou reforçar o potencial de SOA no alinhamento entre as áreas de TI e de negócios. Benedete Junior (2007, p.53), concluiu ainda que “a transformação da empresa, de uma estrutura departamental para uma com foco em processos corporativos, pode ser essa oportunidade de mobilização, de construção de um futuro melhor para a empresa e seus colaboradores.”.

4.1.3 Mestrado

SOA também foi apresentada e defendida em uma dissertação para obtenção de grau de Mestre em Engenharia Informática e de Computadores da Universidade Técnica de Lisboa (Portugal). Pardal (2006), motivado pela resposta à necessidade que as empresas têm em agilizar seus sistemas de informação, enfrentando o desafio de adaptação aos requisitos de negócio.

O principal objetivo da dissertação foi a avaliação de normas de segurança para *Web Services*, através de um estudo de caso e ensaios em protótipos.

Em sua conclusão, o autor identificou mecanismos necessários para implementar a segurança de uma plataforma de serviços.

4.2 NÍVEL COMERCIAL

Comercialmente, existem produtos disponíveis que se propõem a realizar as funcionalidades de um ESB, tanto em forma de software proprietário, quanto na forma de software livre.

4.2.1 Código proprietário

A Microsoft comercializa o software Microsoft BizTalk Server, atualmente na quinta versão (2006). De acordo com o *DataSheet*[13] do produto, "o BizTalk Server 2006 R2 tem como alicerce o BPM e as capacidades de SOA e ESB de versões anteriores para ajudar as organizações a ampliar ainda mais as tecnologias de gerenciamento de processos". Uma simplificação do funcionamento do BizTalk é apresentada por Cambiucci (2008), através da Figura 4.1.

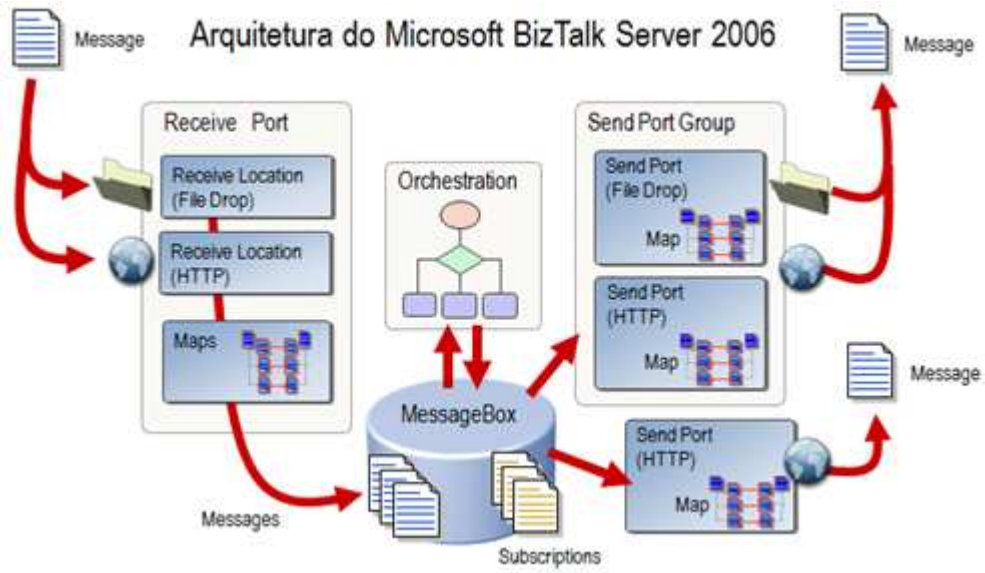

Figura 4.1 – Arquitetura do Microsoft BizTalk Server 2006
Fonte: CAMBIUCCI, 2008

O BizTalk Server 2006 R2 tem diversas edições de licenciamento pagas e uma edição sem custo, denominada de *Developer edition*, "restrita a soluções de criação, desenvolvimento e testes" (BizTalk 2006 DataSheet, p.2).

Estão disponíveis no BizTalk Server 2006 R2, além dos adaptadores comuns (FTP, HTTP, SMTP, POP3, SQL Server...), opções para conexão com bancos de dados IBM DB2 e Oracle, além de softwares da SAP e PeopleSoft.

Conforme tabela[14], o preço do BizTalk Server 2006 R2, pode variar de 499 dólares - versão *developer* para usuários não cadastrados -, até 34.990 dólares.

Para maiores informações, consulte o site da Microsoft[15].

[13] Quadro de características. Especificações dos parâmetros de um dispositivo (hardware ou software), definidas pelo fabricante (SAWAYA, 2002).
[14] Tabela de preços publicada pela Microsoft em 28 de fevereiro de 2008 (http://www.microsoft.com/brasil/servidores/biztalk/howtobuy/default.mspx, acessado em 11/06/2009).
[15] http://www.microsoft.com/brasil/servidores/biztalk/default.mspx.

Outro produto disponível é o IBM WebSphere Enterprise Service Bus (WSESB). Segundo Moreira (2008), esse produto é parte integrante de outro produto da IBM, o WebSphere Application Server (WAS). Como vantagem, Moreira (2008) aponta a facilidade na implantação e o preço reduzido. Para informações adicionais, consulte o site da IBM[16].

4.2.2 Código aberto

O WSO2 ESB é disponibilizado gratuitamente no site da empresa WSO2[17]. A WSO2, com sede no Sri Lanka, foi fundada por Sanjiva Weerawarana, ex-funcionário da IBM, em agosto de 2005. Uma ilustração do seu funcionamento pode ser encontrado no *DataSheet* do produto, reproduzida na Figura 4.2.

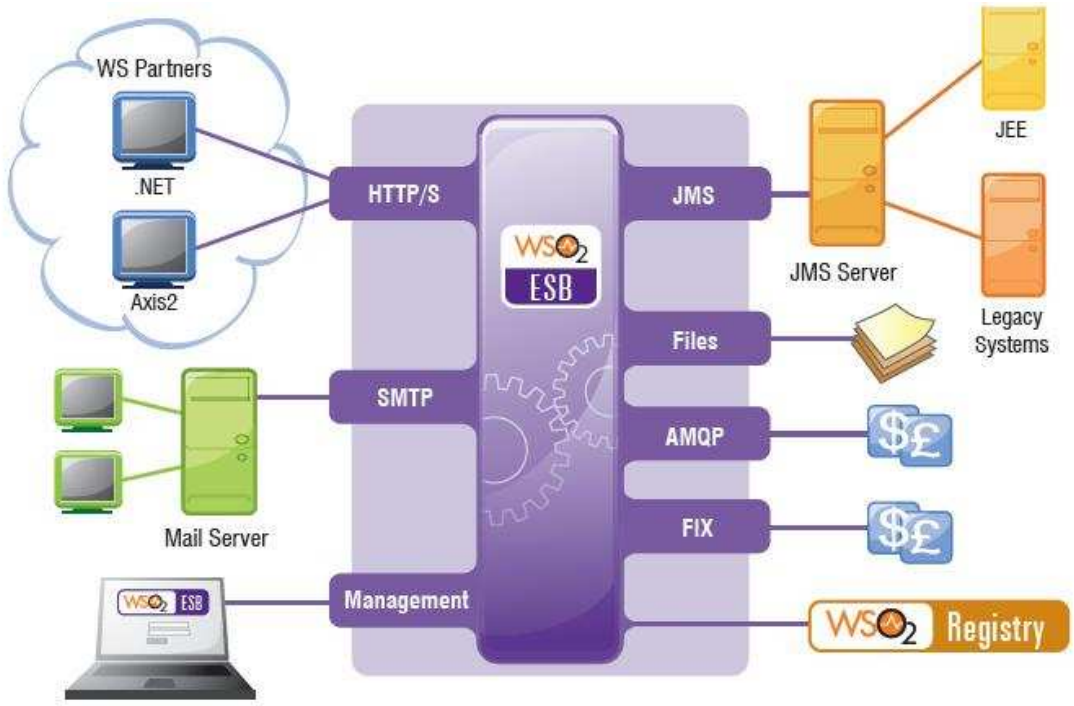

Figura 4.2 – ESB da WSO2
Fonte: WSO2 ESB DataSheet

De acordo com o *DataSheet* do produto, ele é capaz de conectar-se a qualquer banco de dados suportado por Java, além de ser multiprotocolo (FTP, HTTP, POP3, SMTP), com suporte a qualquer tipo de XML.

Um dos mais conhecidos ESB *open-source* é o Mule. Desenvolvido por uma comunidade, é disponibilizado no site http://www.mulesource.org/display/MULE/Home.

[16] http://www-01.ibm.com/software/br/websphere.
[17] http://wso2.com/products/connect/wso2-enterprise-service-bus.

Além disso, no site é possível encontrar farta documentação. A Figura 4.3 ilustra, resumidamente, o funcionamento do Mule e alguns conectores.

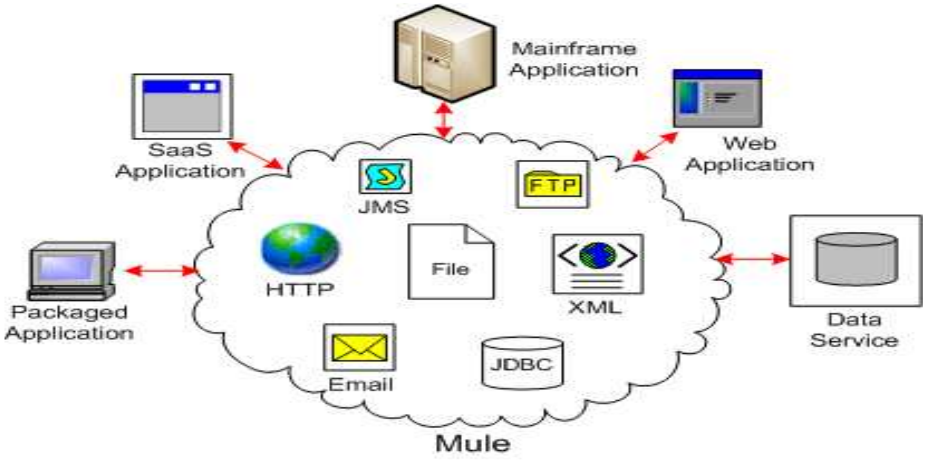

Figura 4.3 – Mule ESB
Fonte: http://www.mulesource.org/display/MULE2INTRO/What+is+Mule

Santos (2007, p.18) apresenta o Mule ESB, destacando suas características:

> "Mule é um framework Enterprise Service Bus (ESB) para mensagens. É escalável e funciona como um object broker distribuído que gerencia interações com services e aplicações utilizando tecnologias de transporte e mensagem. Mule foi desenhado de forma a ser leve e facilmente embutível em aplicações Java e servidores de aplicação ou rodar como um servidor stand alone".

O Quadro 4.1 resume e destaca as principais características e diferenciais de cada ESB mencionado.

ESB	Tipo	Características
MS BizTalk	Código Proprietário	Conectores para DB2 e Oracle; Valores de licença escalonáveis; Versão grátis para desenvolve-dores.
IBM WebSphere	Código Proprietário	Mais barato se já possuir IBM WAS; Conectores para SQL Server e Oracle.
WSOS ESB	Código Aberto	Gratuidade; Conexão com qualquer banco de dados suportado por Java.
Mule	Código Aberto	Gratuidade; Farta documentação.

Quadro 4.1 – Características e diferençais dos ESBs
Fonte: do autor

Ao final deste capítulo, vale salientar uma característica encontrada em cada ESB apresentado – a disponibilidade de conexão por diversos meios, chamados de conectores. É através dos conectores que as aplicações interagem com o ESB, estando ele alocado em um mesmo computador, em uma rede local ou não. Característica esta, que deverá ser observada durante o desenvolvimento do Trabalho de Conclusão II.

5 PROPOSTA

O foco deste capítulo é, inicialmente, apresentar aspectos estruturais e funcionais dos processos da empresa escolhida para a realização deste estudo. Dessa forma, por meio dessa especificação é possível indicar precisamente onde ocorre o principal problema a ser resolvido com este trabalho.

5.1 A EMPRESA ESCOLHIDA

A Sanvitron Controle e Automação Ltda foi fundada em 1997 na cidade de Igrejinha-RS, por dois formandos do curso técnico em eletrônica do Colégio CIMOL de Taquara.

O primeiro projeto da empresa foi o desenvolvimento de um relógio ponto eletrônico para um cliente específico, tendo sido estendido para diversos outros clientes. A empresa desenvolveu e mantém um software para gerenciamento dos registros desses equipamentos. Além disso, o software faz a apuração de horas normais, adicional noturno, repouso remunerado, banco de horas e todos os eventos necessários para o cálculo da folha de pagamento. Esses foram os principais produtos da empresa por diversos anos.

Atualmente, além desses produtos, a Sanvitron desenvolve e produz hardware e software para o controle de acesso através de catracas e cancelas, bem como controle e automação de diversos processos realizados por seus clientes.

É preocupação constante da empresa oferecer sempre produtos de altíssima qualidade, alicerçados em tecnologia de ponta e inovação.

A Sanvitron é estruturada na forma de departamentos independentes, como mostrado na Figura 5.1.

Figura 5.1 – Modelo de estruturação da Sanvitron
Fonte: http://www.sanvitron.com.br/site/content/empresa/estruturacao.php

5.2 MOTIVAÇÃO PARA ESCOLHA

E escolha dessa empresa se deu pelo fato de o autor deste trabalho ser colaborador da Sanvitron desde o ano de 2003, tendo trabalhado em diversos projetos durante este período na função de analista e desenvolvedor de software. Por esse motivo, o autor pode verificar pessoalmente a necessidade da melhoria nos processos de integração dos aplicativos da Sanvitron com as aplicações de folha de pagamento utilizadas pelos seus clientes.

5.3 SITUAÇÃO ATUAL

A empresa integra o resultado do processamento dos registros de ponto realizados pelos funcionários dos seus clientes, combinado com as regras de cálculo estipuladas por eles, com as mais diversas folhas de pagamento do mercado. Essa integração acontece pela configuração de um *layout* estabelecido por cada uma dessas folhas de pagamento. Quando existe uma característica no *layout* que não pode ser atendida pela ferramenta de

configuração, há a necessidade de adequação do código fonte do sistema, para suprir essa necessidade.

Além da integração da apuração dos eventos, há também a integração na entrada de dados. Esses dados são constituídos de informações pertinentes aos funcionários, como código, nome, data de admissão, horários em que deve trabalhar, cargo e função atuais, dentre outros. Da mesma forma, existem diferentes formas de se fazer essa integração:

- Arquivo texto – é realizada a exportação dos dados na folha de pagamento, através da geração de um arquivo texto, depositado em um local previamente combinado entre as aplicações. O software da Sanvitron, denominado STSWin (*Sanvitron Time System for Windows*), ao localizar o arquivo, realiza a leitura do seu conteúdo, importando os dados que ali constam para a sua base de dados. Como cada software de folha de pagamento dispõe os dados de uma forma diferente, esse *layout* também pode ser configurado;
- Bibliotecas – através de bibliotecas de ligação dinâmica (DLL), pode ser realizada a integração com aplicativos desenvolvidos em linguagem de programação que as suportem;
- Banco de dados – menos usual, é a integração através de acesso direto à base de dados, seja do STSWin pela folha de pagamento, seja da folha de pagamento pelo STSWin.

Conforme observado, o processo de integração atual pode ser, superficialmente, exemplificado através da Figura 5.2.

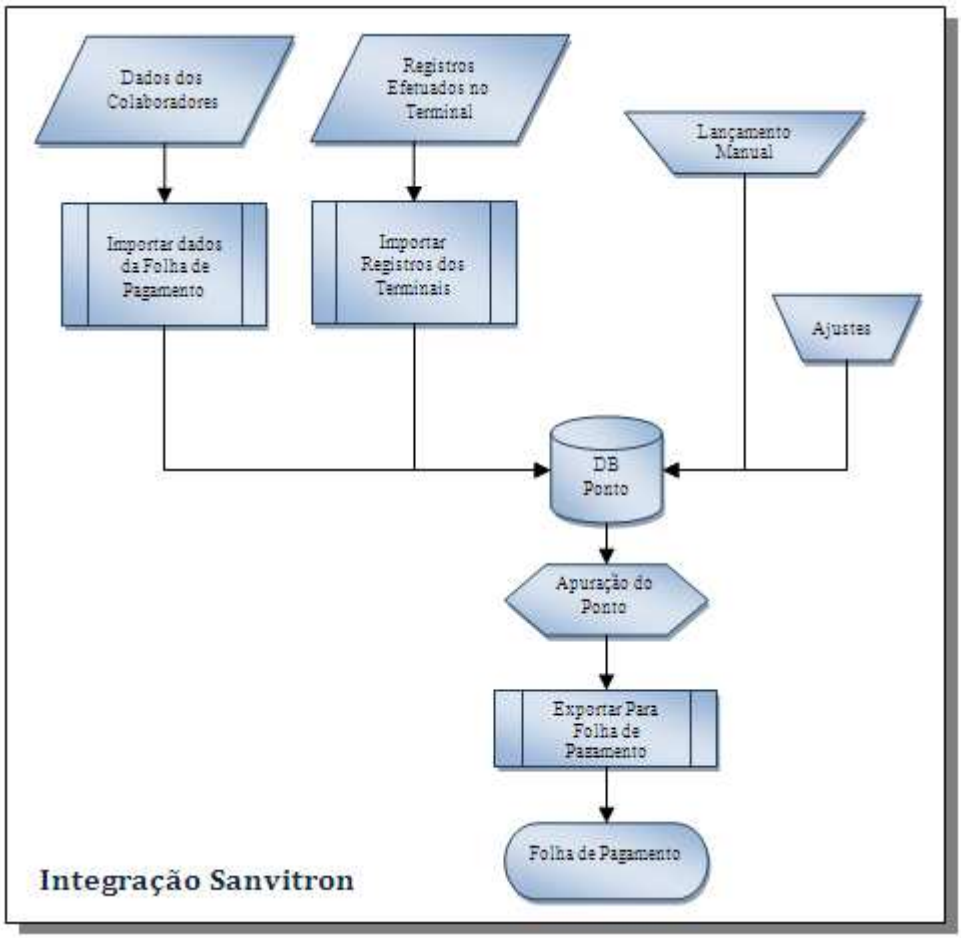

Figura 5.2 – Fluxo do processo de integração
Fonte: Do autor

Os problemas dessa integração são os já citados anteriormente, como o estabelecimento de um forte acoplamento entre as aplicações, em virtude da necessidade das mesmas saberem da existência uma da outra e trabalharem com um padrão comum.

Embora a manutenção do código fonte para adequação a necessidade de uma nova integração seja de fácil realização, sempre há o despendimento de recursos de tempo e pessoal, além do custo financeiro.

No sentido de minimizar a aplicação desses recursos no atendimento de uma nova integração, surge a proposta deste trabalho. Através da arquitetura SOA, especificamente pelo uso de um ESB, acredita-se que o processo de integração possa ser melhorado, de forma a garantir à empresa uma economia – de tempo e financeira.

Finalmente, é importante lembrar que a modelagem completa da proposta será realizada e apresentada no Trabalho de Conclusão II.

6 CONSIDERAÇÕES FINAIS

Este trabalho traz uma revisão bibliográfica sobre a Arquitetura Orientada a Serviços, detalhando conceitos e características, como serviços, interoperabilidade e acoplamento fraco. Além disso, foram estudadas as relações de SOA com desempenho e segurança e o seu impacto sobre as organizações. O Barramento de Serviços Corporativos teve destaque, com um capítulo dedicado ao mesmo. Nele foram detalhados conceitos e funcionalidades – as quais deverão ser observadas durante o desenvolvimento do Trabalho de Conclusão II. A fim de enriquecer o estabelecimento de parâmetros para a continuidade deste trabalho, foram pesquisadas outras formas de se realizar a integração entre softwares aplicativos, bem como estudos já realizados sobre SOA e ESB nos níveis acadêmico e comercial. Da mesma forma, realizou-se uma abordagem inicial sobre a empresa escolhida para aplicação da proposta a ser apresentada, e as maneiras de integração utilizadas atualmente pela mesma.

Cabe enfatizar que, dentre as dificuldades enfrentadas durante o desenvolvimento da presente pesquisa, a principal foi a pouca disponibilidade de material. Motivo pelo qual, em cada capítulo foi fundamentado em uma obra específica, e em complemento, outras publicações foram pesquisadas e referenciadas.

Na próxima fase, Trabalho de Conclusão II, será realizado um maior detalhamento das formas de integração empregadas atualmente pela empresa escolhida, através da descrição da arquitetura hoje utilizada. Como forma de contribuir no aprimoramento dos processos de integração dessa empresa, realizar-se-á o projeto, através da modelagem (ferramentada pela UML) de aspectos estruturais e funcionais de um ESB. Como forma de avaliação do modelo proposto, proceder-se-á na confecção de um protótipo, alicerçado nesse projeto. Feito isso, espera-se dar à empresa escolhida, condições efetivas de competitividade, ora pelo ganho de desempenho proporcionado por essa tecnologia, ora pela redução de custos, através do emprego da reusabilidade e agilidade no desenvolvimento.

REFERÊNCIAS BIBLIOGRÁFICAS

BENEDETE JUNIOR, Antonio Carlos. **Roteiro para a definição de uma arquitetura SOA utilizando BPM**. Monografia apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de MBA em Tecnologia da Informação. São Paulo, 2007.

BOWERSOX, Donald J. ET al. **Gestão Logística de Cadeias de Suprimentos**. Bookman, 2006.

BROOKSHEAR, J. Glenn. **Ciência da Computação: Uma visão abrangente**. 7. ed. Bookman, 2005.

CAMBIUCCI, Waldemir. **Microsoft BizTalk Server 2006 R2 e o modelo ESB – Enterprise Service Bus.** 2008. Disponível em < http://blogs.msdn.com/wcamb/archive/2008/10/04/ microsoft-biztalk-server-2006-r2-e-o-modelo-esb-enterprise-service-bus.aspx>. Acesso em 25 de maio de 2009.

CARVALHO, Davi. **ESB: Mitos e Principais Funcionalidades**. Disponível em <http://soasimples.com/blog/?cat=11>. Acesso em 27 de maio de 2009.

CUMMINS, Fred A. Integração de sistemas: EAI (Enterprise application integration): arquiteturas para integração de sistemas e aplicações corporativas. Rio de Janeiro: Campus, 2002.

DEITEL, Harvey M. **XML: como programar**. Bookman, 2003.

FAYAD, Mohamed; Schmidt, Douglas. **Object-Oriented Application Frameworks**. Communications of the ACM, New York, v. 40, n. 10, p. 32-38, Oct. 1997.

GARCIA et al. **Suporte da Arquitetura Orientada a Serviços na Integração de Sistemas Médicos**. Dissertação apresentada como requisito à obtenção do título de Bacharel em Ciência da Computação pela Universidade Federal de Itajubá. Itajubá, 2008.

MENEZES, Gabriel Nunes; GALOPPINI, Pedro Castello Branco. **Onix. Framework para Integração de Sistemas com Enterprise Service Bus**. Trabalho de Conclusão de Curso submetido à Universidade Federal de Santa Catarina como parte dos requisitos para obtenção do grau de Bacharel em Sistemas de Informação. Florianópolis, 2009.

MOREIRA, Mike. **Conheça um pouco mais sobre ESB (Enterprise Service Bus) e as Ferramentas IBM**, 2009. Disponível em <http://tec.brq.com/conheca-um-pouco-mais-sobre-esb-enterprise-service-bus-e-as-ferramentas-ibm/>. Acesso em 27 de maio de 2009.

PARDAL, Miguel Filipe Leitão. **Segurança de aplicações empresariais em arquitecturas de serviços**. Dissertação para obtenção do Grau de Mestre em Engenharia Informática e de Computadores. Lisboa, 2006.

ROSINI, Alessandro; PALMISANO, Angelo. **Administração de Sistemas de Informação e a Gestão do Conhecimento**. São Paulo: Cengage Learning Editores, 2003.

SAMPAIO, Cleuton. Guia do Java Enterprise Edition 5 - Desenvolvendo aplicações corporativas. Rio de Janeiro: Brasport, 2007.

SANTOS, Alfredo Luiz dos. **Integração de Sistemas com Java**. Rio de Janeiro: Brasport, 2007.

SAWAYA, Márcia Regina. Dicionário de informática & Internet: inglês/português. Nobel, 2002.

SILVA, Roberto Ferreira Lima. **e-RH em um ambiente global e multicultural**. Distrito Federal: Senac, 2009.

SPACKMAN, Devin; SPEAKER, Mark. **Soluções de Integração Empresarial**. Tradução de João Tortello. São Paulo: Bookman, 2006.