UNIVERSIDADE FEEVALE

FÁBIO LUIS FUSSIGER

ANÁLISE EXPERIMENTAL SOBRE ALTERNATIVAS DE INTEGRAÇÃO DE SISTEMAS DE INFORMAÇÃO HETEROGÊNEOS, COM FOCO EM BPEL

Novo Hamburgo
2011

FÁBIO LUIS FUSSIGER

ANÁLISE EXPERIMENTAL SOBRE ALTERNATIVAS DE INTEGRAÇÃO DE SISTEMAS DE INFORMAÇÃO HETEROGÊNEOS, COM FOCO EM BPEL

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do grau de Bacharel em Ciência da Computação pela Universidade Feevale

Orientador: Edvar Bergmann Araujo

Novo Hamburgo
2011

AGRADECIMENTOS

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

Aos amigos e às pessoas que convivem comigo diariamente, minha gratidão, pelo apoio emocional - nos períodos mais difíceis do trabalho.

RESUMO

Diante do cenário de mudanças tecnológicas e da crescente exigência do mercado, é cada vez mais comum encontrar empresas que usam vários sistemas para atender suas necessidades. A partir do momento em que vários sistemas são utilizados, surge a necessidade de que compartilhem dados ou informações. Seja nos casos em que uma informação gerada em um módulo ou sistema inicia a ação em outros, seja nos casos em que a mesma informação é utilizada em vários sistemas, é necessário que haja uma integração sistêmica segura. Para a construção de um modelo de integração de sistemas existem diversas formas técnicas. A escolha deve ser feita de forma criteriosa, pois alguns fatores podem influenciar a eficiência, segurança e funcionamento da solução criada. Deve-se, entre outras coisas, analisar os sistemas envolvidos, identificar o volume de dados trafegados, a periodicidade e se o processo de integração deve ser síncrono ou assíncrono. Neste contexto, surgem ferramentas como o BPEL (*Business Process Execution Language*), quem vêm ganhando visibilidade e ampliando sua utilização no mercado por unir em sua solução as melhores práticas de integração de dados e/ou sistemas. Este trabalho objetiva aprofundar o uso do BPEL como solução de integração, explorando quatro técnicas de integração mais utilizadas no mercado e comparando-as através do uso de um cenário. Este tipo de abordagem visa fornecer entendimento sobre onde, quando e qual tipo de técnica deve ser utilizada em relação ao contexto de integração que é apresentado.

Palavras-chave: BPEL; Integração de Sistemas; Sistemas Heterogêneos; Melhores Práticas.

ABSTRACT

Against the backdrop of technological changes and the increasing market demand, it is increasingly common to find companies that use multiple systems to meet your needs. From the moment when several systems are used, there arises the need to share information and data. Where is that information generated in a module or system initiates the action on others, whether in cases where the same information is used in many systems, there must be a safe systemic integration. For the construction of a model of integration of systems there are several technical ways. The choice must be done carefully, since some factors may influence the efficiency, safety and operation of the solution created. It should, among other things, analyze the systems involved, identify the volume of data traffic, the frequency and the integration process should be synchronous or asynchronous. In this context, there are tools such as BPEL (Business Process Execution Language), who are gaining visibility and expanding its use in the market by uniting in their solution best practices for data integration and / or systems. This paper aims to deepen the use of BPEL as the integration solution, exploring four more integration techniques used in the market and comparing them through the use of a scenario. This approach aims to provide understanding about where, when and what kind of technique must be used in relation to the context of integration is presented.

Keywords: BPEL; Systems Integration; Heterogeneous Systems; Best Practice.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

B2B	*Business to Business*
BAM	*Business Activity Monitoring*
BPEL	*Business Process Execution Language*
BPM	*Business Process Management*
BPMI	*Business Process Management Initiative*
BPML	*Business Process Modeling Language*
BPMN	*Business Process Modeling Notation*
BPMS	*Business Process Management System*
BRM	*Business Rules Management*
DLL	*Dynamic-link library*
EDI	*Intercâmbio Eletrônico de dados*
ERP	*Enterprise Resource Planning*
ETL	*Extract Transform Load*
HTTP	*Hypertext Transfer Protocol*
JAR	*Java Archive*
JDBC	*Java Database Connectivity*
ODI	*Oracle Data Integrator*
OHS	*Oracle Heterogeneous Services*
SGBD	*Sistema Gerenciador de Banco de Dados*
SOA	*Service Oriented Architecture*
SOAP	*Simple Object Access Protocol*
SQL	*Structured Query Language*
TI	*Tecnologia da Informação*
UML	*Unified Modeling Language*
URI	*Unique Resource Identifier*
URL	*Uniform Resource Locator*
VPN	*Virtual Private Network*
WSDL	*Web Service Definition Language*
XI	*Exchange Infrastructure*
XML	*Extensible Markup Language*

SUMÁRIO

INTRODUÇÃO

Diante das rápidas e constantes mudanças tecnológicas e da crescente exigência do mercado, é cada vez mais comum encontrar empresas que necessitem vários sistemas para atender suas necessidades. Muitas vezes os sistemas até então utilizados tornam-se ultrapassados, deixando de atender todas as necessidades. O custo de manutenção ou customização muito elevado e a necessidade de profissionais com conhecimento das melhores práticas do negócio tornam, na maioria das vezes, mais atrativa e rentável a aquisição de um novo sistema ou módulo especialista.

A partir do momento em que vários sistemas são utilizados, surge a necessidade de que compartilhem dados ou informações. Seja nos casos em que uma informação gerada em um módulo ou sistema inicia a ação em outros, seja nos casos em que a mesma informação é utilizada em vários sistemas, é necessário que haja uma integração sistêmica segura. Este processo deverá ser transparente aos usuários, ou seja, sem que haja uma interação humana, evitando assim, que ocorram erros operacionais ou de digitação.

> *Os usuários esperam ter acesso instantâneo a todas as funções de negócio que uma empresa pode oferecer, independentemente do sistema onde a funcionalidade é originalmente disponibilizada. Isso requer que diferentes aplicações possam se conectar através uma grande solução integrada (HOHPE, 2010, p. 1).*

No processo de análise para integração de sistemas a equipe se depara com uma série de desafios. O principal é a falta de preparo e qualificação da equipe técnica, que por vezes desconhece detalhes dos sistemas envolvidos, bem como as melhores soluções aplicadas no mercado. Sistemas adquiridos de diferentes fornecedores, com as mais diversas bases de dados e sistemas operacionais, são alguns dos fatores que levam as empresas a buscarem soluções de integração prontas ou a contratação de equipes de consultoria.

Em sintonia com as necessidades do mercado, empresas fornecedoras de sistemas corporativos já perceberam que não basta oferecer um bom produto, precisam oferecer também meios para que este possa se comunicar com os softwares já existentes. Empresas como a SAP já apresentam iniciativas, como por exemplo, a ferramenta de integração XI (*Exchange Infrastructure*) (SAP, 2010).

Os fornecedores de banco de dados também identificaram está demanda e passaram a disponibilizar ferramentas de integração entre banco de dados de diferentes fornecedores. Um

exemplo é a Oracle, que lançou no final do ano passado sua nova ferramenta de integração denominada Golden Gate (ORACLE, 2009).

Há ainda as técnicas e tecnologias usadas na concepção de soluções de integração sob medida. Uma técnica de integração já bem difundida que ainda hoje é muito utilizada para trafegar grande quantidades de informações é o EDI (Intercâmbio Eletrônico de Dados), que baseia-se na troca de informações através de arquivos estruturados. É comum encontrar este tipo de solução aplicada para automatizar a comunicação entre empresas, como também, em menor proporção, empregada para integrações entre sistemas internos.

Soluções de integração *open source* também vem ganhando espaço, como o SOA (*Service Oriented Architecture*). SOA é um paradigma de desenvolvimento de aplicações cujo objetivo é criar módulos funcionais chamados de serviços, com baixo acoplamento e permitindo a reutilização de código (SAMPAIO, 2006, p. 14). Estes serviços são programas menores não muito complexos que tem por objetivo executar ações específicas como consulta e/ou gravação de dados. São fragmentos de programas que atuam como apoio a sistemas maiores, podendo ser aplicado para os mais diversos fins como, por exemplo, integração de sistemas.

Neste universo variado de propostas de integração que vem aumentado muito nos últimos anos, as empresas ainda encontram barreiras quanto a utilização destas soluções, seja pela complexidade ou ainda pelo custo elevado para aquisição. Neste contexto, surge o BPEL, uma linguagem para orquestração de serviços que tem sido base para criação de ferramentas que visam facilitar a tarefa de integração de sistemas. Estas ferramentas caracterizam-se por serem produtivas e amigáveis e contemplarem as principais técnicas de integração utilizadas. É possível encontrar ferramentas *open source* e comerciais, muitas destas com custos acessíveis, como por exemplo, o JDeveloper da Oracle.

No decorrer deste trabalho serão exploradas quatro técnicas de integração, onde além de apresentar seus conceitos serão exploradas suas funcionalidades através do uso de um cenário. A proposta é trabalhar com um único cenário, onde uma determinada quantidade de dados deverá ser integrada entre sistemas heterogêneos, utilizando para isso quatro técnicas de integração: integração direta via banco de dados, *Web Service*, EDI e integração banco a banco, sendo que nos últimos três citados a implementação será realizada pelo BPEL. Para cada técnica utilizada será analisado qual melhor desempenha a atividade de integração para o cenário proposto.

Este trabalho está estruturado da seguinte forma: o primeiro capítulo trata dos conceitos sobre integração de sistemas e modelagem de processos, na sequência são apresentados conceitos e técnicas de integração mais utilizadas no mercado. Depois no capítulo 3 será detalhado o cenário de integração, nos capítulos seguintes do 4 a 7 são detalhados os procedimentos para construção dos cenários, aplicação e análise dos resultados e finaliza com a conclusão do trabalho.

1 PANORAMA TECNOLÓGICO

Em decorrência das inúmeras evoluções tecnológicas ocorridas nos últimos anos, as empresas perceberam que não era mais possível permanecer competitivas sem que melhorassem seus sistemas. Neste contexto, novas ferramentas surgiram, bem como pequenas empresas de software passaram a oferecer soluções a preços mais atrativos. Uma solução encontrada pelas empresas para fugir dos altos custos de manutenção/evolução dos sistemas atuais ou dos altos custos de aquisição, customização e implantação de uma solução de mercado completa, é a opção por sistemas especialistas (modulares) para executar determinadas ações não atendidas pelos sistemas atuais. Isto gerou por um lado um diferencial competitivo, mas por outro um problema de integração para área de tecnologia.

1.1 Integração de sistemas

O termo Integração de Sistemas é utilizado para definir a atividade de desenvolvimento necessária para que sistemas diferentes possam se comunicar. Esta atividade, além de bastante complexa, possui certas particularidades que variam de acordo com as características da empresa. A diversidade de softwares é um desafio para as equipes de TI (Tecnologia da Informação), principalmente pela falta *konw how* para garantir ou propor a melhor forma de integração. Muitas vezes, esta atividade é transferida para empresas especializadas neste novo tipo de negócio.

> *"A arquitetura de integração corporativa deve fornecer uma estrutura em que os sistemas legados eficientes possam continuar a funcionar quando novas tecnologias e mudanças de negócios correspondentes se integrarem a operação da empresa." (CUMMINS, 2002, p. 4).*

No cenário atual, onde mudanças tecnológicas são necessárias para que as empresas se mantenham competitivas, as empresas normalmente optam por: desenvolver ou customizar as aplicações para atender as novas necessidades ou adquirir sistemas especialistas que possam vir a agregar novas funcionalidades ao sistema existente. É possível encontrar em algumas empresas os dois cenários convivendo juntos, sendo cada um responsável por um conjunto de processos ou negócios.

A customização de sistemas, especialmente quando desenvolvidos especificamente para os processos da empresa, torna-se uma opção atraente, pois é altamente aderente. Por outro lado, surgem dificuldades como os altos custos e a falta de profissionais qualificados para manter um sistema com tecnologia algumas vezes ultrapassada, prejudicando a empresa

em termos de agilidade e competitividade, necessárias para acompanhar as evoluções do mercado.

Quanto mais específicos e dependentes forem os sistemas de uma empresa em relação aos seus processos, maior é dificuldade de encontrar um sistema de mercado que possa atender as novas necessidades de forma totalmente aderente. Em contrapartida, o mercado de software está repleto de sistemas especialistas, que atendem as melhores práticas de mercado. Estes sistemas, na maioria das vezes, são comercializados em pacotes ou módulos, o que torna seu custo bastante atraente, fazendo com que muitas empresas adotem este modelo de solução. A contratação de diversos sistemas especialistas de diferentes fabricantes ou tecnologias, inicialmente atende, de forma mais rápida e ágil a necessidade da empresa. Contudo, após algum tempo, a necessidade de que estes sistemas compartilhem informações entre si, resulta em aumento de custos e tempo.

Segundo Cummins (2002, p. 22), a tecnologia evolui constantemente fazendo com que os sistemas avançados de hoje, sejam os sistemas legados de amanhã. As mudanças nas empresas continuarão ocorrendo em alta velocidade, orientadas em parte por mudanças na tecnologia e pela necessidade de explorar a tecnologia para permanecerem competitivas.

Todas as considerações feitas nesta seção reforçam uma necessidade importante das empresas atuais: a de integrar os sistemas de informação utilizados por elas de forma efetiva e eficiente.

1.2 Sistema legado

Sistema Legado é o termo utilizado em referência aos sistemas computacionais de uma organização que, apesar de serem bastante antigos, fornecem serviços essenciais. A permanência destes, que por vezes utilizam tecnologias já defasadas, se faz necessária por implementarem funcionalidades, regras de negócio e/ou disponibilizarem recursos que atendem a necessidades específicas da empresa, tornando os únicos em relação aos demais sistemas de mercado.

A partir do momento que outros sistemas passam a ser necessários, seja para agregar novas funcionalidades ou substituir funções já existentes, deve ser considerada sua compatibilidade bem como seu impacto sobre os sistemas legados. Para Cummins (2002, p. 2), não é possível compreender o impacto da nova tecnologia sem levar em conta os sistemas legados. O atual panorama corporativo é composto de sistemas que são o resultado da

evolução dos negócios e da tecnologia com o passar dos anos. Foram desenvolvidos sistemas para solucionar problemas específicos de alta prioridade ou para obter aprimoramentos específicos na produtividade.

Também segundo Cummins (2002, p. 2), a evolução da tecnologia também contribui para a fragmentação de sistemas, uma vez que quando novos sistemas são desenvolvidos é utilizada a tecnologia mais recente. Sistemas criados com tecnologias mais antigas não podem simplesmente ser atualizados à nova tecnologia, visto que isto exigiria um retrabalho ou redesenvolvimento substancial. Se a funcionalidade de negócios atual de um sistema for adequada, muitas vezes não haverá um grande incentivo para execução do retrabalho.

1.3 Sistemas heterogêneos

O termo sistemas heterogêneos é utilizado para referenciar aplicativos que são desenvolvidos em linguagem ou tecnologias distintas comparadas entre si. Também é possível ter sistemas heterogêneos desenvolvidos em tecnologias idênticas, porém, com dados conceitualmente divergentes.

Os sistemas heterogêneos em geral resultam de sistemas locais individuais que implementam suas próprias bases de dados e a integração é deixada para um estágio posterior. Em um sistema heterogêneo é necessária à tradução para permitir a comunicação entre os diferentes SGBDs (Sistema Gerenciador de Banco de Dados), uma vez que os dados são solicitados de locais com hardware e SGBDs diferentes (CAVALCANTI, 2001, p.15).

Existem vários aspectos que fazem de um conjunto de sistemas um corpo heterogêneo. Mesmo nos casos onde as bases de dados possuem o mesmo fornecedor é possível encontrar estas diferenças. De acordo com Sheth (1998, p. 8-9) dentro de um SGBD, o modelo utilizado para definir estruturas de dados, suas restrições, como também aspectos da linguagem pode levar a heterogeneidade. Estas diferenças classificam-se em:

1. **Diferenças de estrutura -** estas são entendidas como diferenças na concepção da modelagem dos dados. Por exemplo, o endereço pode ser representado por um único campo em uma entidade de banco de dados e em outra como um atributo composto (tipo de rua, logradouro e número);

2. **Diferenças nas restrições -** trata das diferentes formas de implementação de restrições aos dados. Como exemplo, é possível garantir a integridade referencial dos

dados a partir de recursos disponibilizados para este fim (*foreign key constraints*) ou ainda implementadas através do uso de *triggers*;

3. **Diferenças de linguagens de consulta -** podem acontecer em decorrência de diferentes versões de SQL (*Structured Query Language*) suportadas pelos SGBDs no decorrer do tempo (SQL-92, SQL: 1999 e SQL: 2003);

4. **Diferenças nos aspectos do sistema SGBD -** estas versam sobre diferenças primitivas de gerência de transação e técnicas (incluindo controle de concorrência), onde a aplicação destes métodos pode variar entre os fornecedores.

1.4 Interoperabilidade

A expansão da tecnologia computacional tem acontecido em ritmo acelerado desde o surgimento e a disseminação dos computadores nas grandes empresas, passando pela popularização dos computadores pessoais, até o crescimento da Internet. Em poucas décadas, chegou-se a um ponto no qual as organizações mantêm sistemas legados de grande porte, utilizando tecnologia ultrapassada, ao mesmo tempo em que desenvolvem e mantêm sistemas tecnologicamente atualizados. (FEDERAL COORDINATING COUNCIL FOR SCIENCE, 1994).

A possibilidade de que estes sistemas definidos como heterogêneos, possam trocar informações entre si é chamado de interoperabilidade. A interoperabilidade é definida como "a interconexão efetiva de diferentes sistemas de computador, bancos de dados ou redes com o fim de apoiar a computação distribuída e/ou o intercâmbio de dados". Apesar de ser uma característica desejada nos sistemas, a interoperabilidade plena dificilmente é atingida, mesmo quando se trata de sistemas relativamente atuais. (FEDERAL COORDINATING COUNCIL FOR SCIENCE, 1994).

A grande diversidade de sistemas disponíveis no mercado para os mais diferentes propósitos, baseadas nas mais diversas linguagens, tornam cada vez mais remota a possibilidade de sistemas heterogêneos interoperarem nativamente. No entanto, iniciativas neste sentido têm sido realizadas. Neste âmbito cabe dar destaque aos *Web Services*. O ponto central da interoperabilidade em *Web Services* repousa sobre a compatibilidade dos principais componentes da tecnologia XML(*Extensible Markup Language*). (NETO e LINS, 2006, p 169).

Existem ainda outras barreiras significativas para alcançar a interoperabilidade de forma efetiva e ampla. Essas barreiras podem ser classificadas como políticas, organizacionais, econômicas e técnicas (MESQUITA e BRETAS apud ANDERSEEN, 1991):

1. **Políticas –** dentre as barreiras políticas destacam-se: definição das diretrizes das políticas adotadas; conflitos nas definições dos níveis de privacidade nos acessos às informações; cultura organizacional predominante; ambigüidade da autoridade na coleta e uso das informações; descontinuidade administrativa;

2. **Organizacionais** – falta de experiência e ausência da predisposição de compartilhar; nível de qualificação do pessoal envolvido nos processos; cultura organizacional;

3. **Econômicas –** falta de recursos para disponibilização das informações para outros órgãos; forma de aquisição dos recursos (normalmente adquiridos pelo menor preço e não pelo melhor valor);

4. **Técnicas –** incompatibilidade de hardware e software adotados; direitos de propriedade; desconhecimento dos dados gerados e armazenados pelos sistemas; múltiplasdefinições de dados.

Em suma, nos últimos anos diversas ferramentas propõem-se a estabelecer a interoperabilidade entre sistemas, uma vez que o mercado já sinalizou que não quer um único padrão, ou seja, não quer aceitar uma solução imposta por único fornecedor. Para apoiar esta tendência, o mercado já dispõe de diversas tecnologias de integração que tem por objetivo resolver o problema de interoperabilidade entre os sistemas.

Uma vez que os sistemas utilizados pelas empresas são normalmente atrelados aos seus processos, bem como os processos de negócio tem sua qualidade e funcionalidade suportada pelos sistemas, a seguir discutir-se-á um pouco sobre modelagem de processos para depois tratar então de ferramentas que suportam tal modelagem.

1.5 Modelagem de processos

Para as empresas manterem-se competitivas, mais do que nunca, precisam conhecer a forma com que se organizam para execução de suas atividades. Para poder entender se as atividades (serviços automáticos ou humanos) estão contribuindo para o crescimento da empresa, não é possível analisar estas atividades de forma isolada e sim dentro do processo

em que está inserida. Além disto, é necessário entender como estes processos interagem para que empresa possa realizar o trabalho que se propôs a fazer.

Segundo Gonçalves (2000, p. 10), a análise dos processos nas empresas implica na identificação das diversas dimensões desses processos: fluxo (volume por unidade de tempo), sequência das atividades, esperas e duração do ciclo, dados e informações, pessoas envolvidas, relações e dependências entre as partes comprometidas no funcionamento do processo.

A necessidade de mapear e representar os processos fez com que surgisse o conceito de *workflow* que ainda hoje é muito aplicado. Também serviu como base para criação de ferramentas como a BPMS (*Business Process Management Software*) que possuem os recursos de modelagem, implementação e monitoramento de processos onde o conceito de *workflow* (modelagem) apenas representa uma fração destes recursos.

1.5.1 Workflow

Workflow pode ser definido como a forma gráfica de representação de um determinado conjunto de atividades necessárias para conclusão de uma tarefa. Este tipo de recurso ainda é muito utilizado pelas empresas para representar seus processos (manuais ou até mesmo automatizados). A necessidade por organizar de alguma forma a comunicação entre as pessoas, transferindo responsabilidades ao longo do processo era e até hoje é fundamental em qualquer tipo de empresa (REIS, 2008, p. 6).

Workflow, ou literalmente Fluxo de Trabalho, é a modelagem e gestão informática do conjunto das tarefas a realizar e dos diferentes atores implicados na realização de um processo do negócio (também chamado processo operacional). O termo *Workflow* poderia, por isso, ser traduzido em português por Gestão Eletrônica dos Processos do Negócio. Um processo operacional representa as interações sob a forma de troca de informações entre diversos atores, como: humanos, aplicações ou serviços, processos terceiros, etc. De maneira prática, um *Workflow* pode descrever: a sequência de validação, as tarefas a realizar entre os diferentes atores de um processo, os prazos a respeitar, os modos de validação, entre outros. (KIOSKEA, 2010).

Fornece, além disso, a cada um dos atores as informações necessárias para a realização da sua tarefa (KIOSKEA, 2010). Uma tarefa pode necessitar do envolvimento de um ator ou ser executada automaticamente por um sistema de informação. Entre as tarefas

existentes no *Workflow* podem circular documentos e informação diversas. Um sistema de gestão de *workflow* efetua leituras, escritas, processamento, automatização de tarefas e gestão de *Workflows*.

1.5.2 BPMN

> *BPMN (Business Process Modeling Notation) é uma notação visual para representação de fluxos de processos que pode ser mapeada para diversos formatos de execução, como BPM (Business Process Management) e BPEL. Em uma analogia com a orientação a objetos, o BPMN seria como UML (Unified Modeling Language), que define elementos gráficos para representar objetos e classes, como também sua interação.*
>
> *Da mesma forma que a UML, o BPMN não define formatos de armazenamento nem elementos programáticos relacionados à implementação, como por exemplo, as rotinas. Por outro lado, possui representação de fluxos de processo pelas áreas de negócio, com detalhamento bem próximo das complexidades de um ambiente real, além de ter elementos que se aproximam de mecanismos de execução, como pode ser observado na figura 1.1 (OLIVER, 2010, p. 220).*

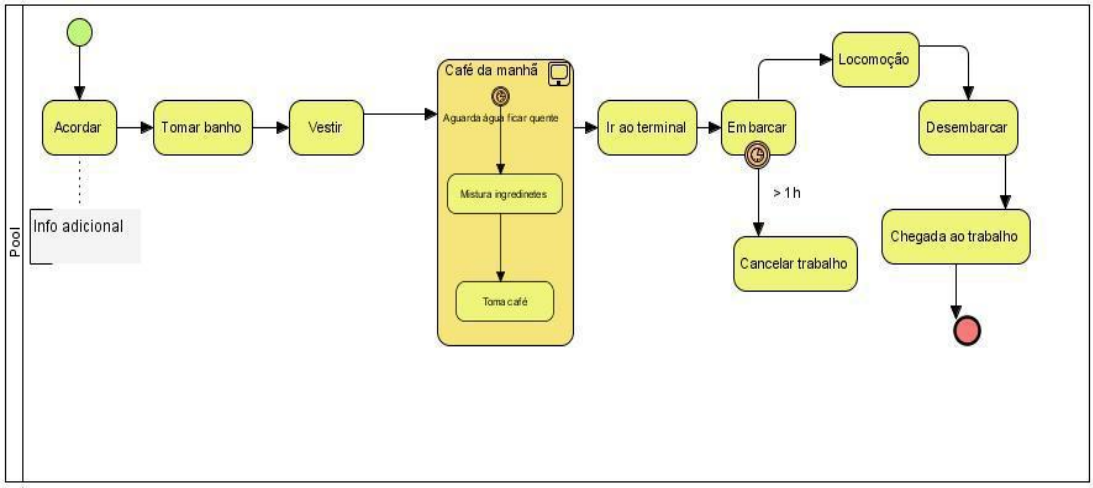

Figura 1.1 – Exemplo de diagrama de processo de negócio
Fonte: Rabelo, 2009

A sigla BPMN é utilizada quando se trata de modelagens e nada mais é que a linguagem que define como os fluxos devem ser representados. Segundo Bitencourt (2007, p. 1), a BPMN está se consolidando como o mais importante padrão de notação gráfica aberta para desenhar e modelar processos de negócios. Com ela é possível modelar os processos de negócio, capturando e documentando modelos atuais (AS-IS) em diagramas de fácil entendimento, projetar e descrever modelos ideais (TO-BE), estender detalhes técnicos,

monitorar e mensurar o negócio com indicadores de desempenho baseados nas atividades dos fluxos de processos automatizados.

O objetivo do desenho é ser de entendimento rápido por todos os usuários do negócio, de forma que permita aos analistas criarem seus primeiros esboços dos processos. Também deve permitir aos arquitetos de TI e desenvolvedores adaptar os processos a serem gerenciados e monitorados (BITENCOURT, 2007, p. 1). Mesmo com a padronização do modelo de representação, faltava ainda uma ferramenta que permitisse a execução destes processos. Foi quando surgiu o BPEL, uma linguagem para execução de processos de negócio compatível com a notação BPMN.

1.5.3 BPMS

Amaral (2006, p. 1) define BPMS como uma categoria de softwares que visa atender o ciclo completo da Gestão de Processos, composta por: modelagem, redesenho, implementação, monitoramento e otimização de processos. Ainda segundo o autor há vários estudos, de diferentes fontes respeitáveis, tais como Gartner, IDC, Forrester e BPMG, que enumeram os requisitos de um BPMS. Apesar de haver diferenças nas definições empregadas, existe uma crescente convergência, unificando cada vez mais o conceito de um BPMS. Segundo Amaral (2006, p. 1) atualmente espera-se que uma verdadeira solução de BPM:

1. Seja aderente aos padrões da área como BPMN, BPEL e/ou BPML (Business Process Modeling Language);

2. Permita modelar processos de negócio, podendo também simulá-los e documentá-los extensivamente;

3. Tenha componentes prontos para integração com sistemas heterogêneos. Integrações via *Web Services*, JMS e JCA são básicas, sendo esperados mecanismos prontos para conectar com ERP's (*Enterprise Resource Planning*) de fornecedores como SAP, Peoplesoft, Oracle E-Business Suite, etc;

4. Possua componente de BAM (*Business Activity Monitoring*) ou integre-se nativamente a um produto deste tipo. Uma solução de BAM monitora em tempo real os indicadores dos processos, e permite que os gestores tomem ações corretivas imediatamente;

5. Possua componente de BRM (*Business Rules Management*) ou integre-se nativamente a um produto deste tipo. Um BRM permite separar as regras dos

processos do código de aplicação, permitindo que usuários de negócios configurem estas regras de forma ágil e transparente.

Um recurso muito apreciado pelas áreas gerenciais são os *Dashboards*, que são gráficos gerenciais, gerados a partir dos processos em execução, e que permitem verificar visualmente quando algo não vai bem. Isto ajuda na tomada de decisão. Uma característica dos *Dashboards* é que eles tendem a ser muito dinâmicos. Portanto, deve ser fácil criá-los e alterá-los sem demandar grandes esforços da TI (REIS, 2008, p. 13).

1.6 Considerações finais

Com o passar dos anos ficou claro que o mercado deseja viabilizar seus negócios utilizando as tecnologias que possam lhe trazer um maior diferencial competitivo. O investimento em tecnologia está aos poucos deixando de ser visto pelas empresas, como um custo, ou mal necessário, mas sim como parte da estratégia da empresa em ganhar mercado.

A crescente oferta no mercado de pacotes de softwares especialistas, bem como de empresas que atuam somente neste segmento, tornou atraente esta opção. Esta opção permitiu mais agilidade às empresas, porém penalizou a infra-estrutura de sistemas.

A necessidade de integrar as aplicações internas com os pacotes especialistas adquiridos, fez com que surgisse um novo segmento no mercado, que são empresas de consultoria onde o foco dos serviços é ajudar as empresas a encontrar a melhor forma de integração para que seus sistemas possam interoperar. Aliado a isso já é possível encontrar ferramentas específicas desenvolvidas para atender este tipo de necessidade, ferramentas de fácil utilização e que permitem o gerenciamento de todo o fluxo de integração. É neste contexto que este trabalho pretende mostrar o quão necessário hoje em dia é a integração de sistemas, bem como mostrar algumas estratégias que podem ser aplicadas nas soluções de mercado.

2 TECNOLOGIAS DE INTEGRAÇÃO

A integração entre sistemas não é um fato recente. Grandes empresas já enfrentavam problemas dessa ordem. No entanto, ainda se vendia a idéia de que os ERP's podiam suprir todas as necessidades afastando a necessidade de integração, mas o tempo provou que estavam errados. Hoje a necessidade de integração é uma realidade e muitas das soluções utilizadas naquela época ainda são utilizadas nos dias de hoje. Muitas evoluções e novas ferramentas têm surgido neste contexto, tornando importante a decisão sobre qual alternativa de integração utilizar. Este capítulo trata de quatro alternativas de integração: integração direta via banco de dados, integração via banco de dados com BPEL, e integração com SOA e EDI implementadas via BPEL.

O aumento da procura por soluções completas que não somente realizem o papel de integração, mas que também disponibilizem meios para que se gerencie todo o processo passaram ganhar espaço. É neste contexto de possibilidades, que entender a utilidade de cada um destes recursos passa a ser fundamental para realização de qualquer atividade de integração.

2.1 SOA

SOA (*Service Oriented Architecture*) é um paradigma de desenvolvimento de aplicações cujo objetivo é criar módulos funcionais chamados de serviços, com baixo acoplamento e permitindo a reutilização de código (SAMPAIO, 2006, p. 14). Estes serviços são programas menores, não muito complexos, que tem por objetivo executar ações específicas como consulta e/ou gravação de dados. São fragmentos de programas que atuam como apoio a sistemas maiores, podendo ser aplicados para os mais diversos fins como, por exemplo, integração de sistemas. Em suma, SOA é um estilo de arquitetura, orientada a serviços que podem ser implementados de diversas formas, sendo uma delas através de *Web Services*.

A utilização de *Web Services* tem sido cada vez mais frequente no mercado, sendo vários os motivos que os tornam tão atrativos. Um dos motivos é que os *Web Services* vão ao encontro do movimento da tecnologia da informação em busca de ferramentas padronizadas que facilitem a integração de dados e comunicação entre sistemas. Diversas tecnologias ou linguagens de programação podem ser utilizadas para o desenvolvimento destes serviços. Além disso, eles podem ser integrados de forma transparente, e são reutilizáveis, ou seja, um

Web Service pode ser utilizado em diversos sistemas ou processos diferentes executando as mesmas funções.

Estas características fazem da utilização desta solução na integração de sistemas e na comunicação entre aplicações uma forte tendência. Torna-se uma das alternativas mais viáveis para a integração entre novas aplicações e sistemas já implantados, mesmo que sejam desenvolvidos em plataformas diferentes. *Web Services* é a tecnologia ideal para comunicação entre sistemas, sendo muito usado em aplicações B2B (*Business To Business*). A comunicação entre os serviços é padronizada possibilitando a independência de plataforma e de linguagem de programação (PAMPLONA, 2010, p. 1).

Segundo W3C (2007, p. 1), um *Web Service* é um sistema de software projetado para suportar a interoperabilidade entre máquinas sobre rede. Tem uma relação descritiva num formato *machine-processable*, especificamente WSDL (*Web Services Description Language*). Outros sistemas interagem com o *Web Service* usando as mensagens SOAP, tipicamente sobre HTTP (*Hypertext Transfer Protocol*) com XML na junção com outros *standards* da Web. Este protocolo é o responsável pela independência que o *Web Service* precisa. Atualmente já encontram-se várias implementações disponíveis em várias linguagens (PAMPLONA, 2010, p. 1).

Na figura 2.1 apresenta-se um esquema simples que pode ajudar a visualizar um *Web Service* e mostrar como aplicações remotas podem interagir umas com as outras usando este recurso.

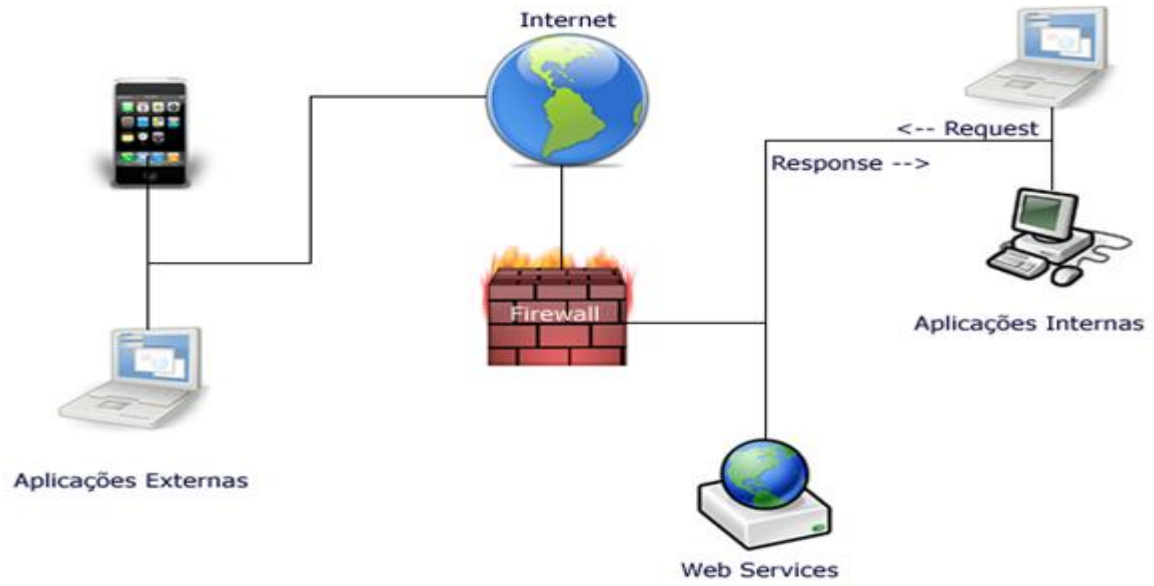

Figura 2.1 – Esquema de funcionamento de *Web Services*
Fonte: Oliveira, 2010

Segundo Cunha (2002, p. 1), os *Web Services* são identificados por um URI (*Unique Resource Identifier*), descritos e definidos usando XML. Um dos motivos que tornam os *Web Services* atrativos é o fato deste modelo ser baseado em tecnologias *standards*, em particular XML e HTTP. Os *Web Services* são utilizados para disponibilizar serviços interativos na Web, podendo ser acessados por outras aplicações usando, por exemplo, o protocolo SOAP.

Os *Web Services* são sistemas disponibilizados para uso na internet ou intranet e não possuem uma interface para uso final, são geralmente acessados por outros serviços ou por sistemas. São utilizados para integrações síncronas e podem ser acessados tanto por aplicações internas como externas.

2.2 EDI

EDI é uma técnica muito difundida entre as empresas que se baseia na troca de informações através de arquivos estruturados. É utilizado principalmente em integrações onde precisam trafegar grandes quantidades de dados. É comum encontrar este tipo de solução para automatizar a comunicação entre empresas, como também, em menor proporção, empregada para integrações entre sistemas internos.

Segundo Colcher e Valler (2000, p. 34), o EDI é uma ferramenta estratégica em alguns segmentos do mercado, onde a rapidez e segurança na comunicação entre fornecedores e clientes são fatores críticos desse relacionamento. O EDI não é, entretanto, uma ferramenta nova. Os princípios básicos dessas tecnologias já orientavam o desenvolvimento de alguns sistemas que datam de meados da década de 60. Embora a tecnologia básica do EDI esteja disponível há vários anos e também não seja necessariamente sofisticada, o processo de difusão amplo no mercado só foi deflagrado a partir do amadurecimento das tecnologias de informação e comunicação, dos avanços na busca de um padrão universal de mensagem e das transformações no padrão de concorrência do mercado.

Ao implementar o EDI, é preciso levar em conta questões como sua integração com os processos internos da empresa e a maneira de trocar os dados de acordo com as necessidades dos parceiros. Para que os documentos eletrônicos e os dados fluam harmoniosamente entre as empresas e sejam corretamente interpretados, é preciso que sejam respeitadas certas regras. Essas regras definem o conteúdo da informação, isto é, os dados dos documentos, e a forma como eles são transmitidos (EAN, 2010, p. 1).

A estrutura de integração deste tipo de solução é composta basicamente de um sistema de origem (onde os dados são extraídos e formatados em um arquivo), um canal de comunicação (local onde o arquivo será disponibilizado) e o sistema de destino (responsável por processar os dados, importando as informações para sua base de dados), como pode ser observado na figura 2.2.

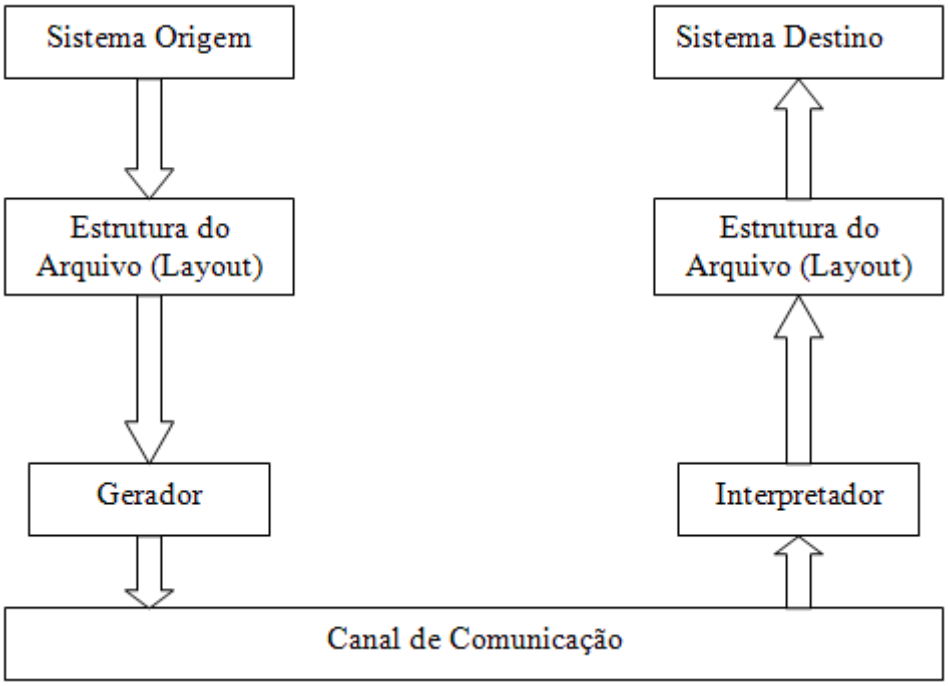

Figura 2.2 – Fluxo básico de troca de arquivos
Fonte: Próprio autor

De modo geral, não há uma solução única para desenvolver um sistema de EDI, nem um ambiente de processamento de dados específico. A solução de EDI comporta uma grande variedade de opções em termos de equipamentos, sistema operacional, software básico, aplicativos, protocolos de comunicação e de interface com os usuários. Tal diversidade implica graus variados na relação custo/desempenho referente aos diferentes sistemas de EDI utilizados. Entretanto, a lógica da operação do serviço é basicamente a mesma. (COLCHER e VALLER, 2000, p. 34).

A solução EDI automatiza operações que antes eram realizadas manualmente, através da digitação. Garante que a informação gerada como saída de um sistema seja a mesma processada como entrada no outro, não havendo alteração em seu conteúdo. Contudo, não é uma solução que possa ser aplicada a qualquer tipo de integração. Alguns fatores devem ser levados em consideração, como, por exemplo, a velocidade que se deseja que esta informação trafegue até seu destino. Neste modelo de integração as informações são geradas e

processadas em momentos diferentes, não há uma sincronia entre a geração dos dados e o seu processamento.

2.3 BPEL

Em 2002, numa iniciativa da BPMI (*Business Process Management Initiative*), a mesma organização que anos antes havia consolidado no mercado o padrão BPM, surge a BPML, linguagem para execução de processos de negócio. Mas, ao contrário do BPM, não foi bem aceito no mercado, recebendo muitas críticas. Segundo Bortolini (2006, p. 1), em paralelo a tudo isso, as empresas Microsoft, IBM, Siebel, BEA e SAP resolveram se unir e propor (ou impor) um novo modelo, mais completo e universal, denominado BPEL. O BPEL surgiu da união de outros padrões, mais especificamente XLANG da Microsoft e o WSFL da IBM. Depois de concluído, seu meta-modelo e especificação inicial, passaram o controle do padrão para a organização internacional Oasis. Rapidamente, esta nova especificação começou a chamar a atenção dos principais *players* do mercado de gestão de processos. Amparado pelas grandes empresas de TI do mundo, ficou claro para muitos que o BPEL seria o padrão mais largamente aceito em curto espaço de tempo.

O BPEL é uma linguagem que é descrita em XML, basicamente utilizada para desenhar processo de negócio seguindo como base o BPM. Destaca-se pela simplicidade da criação de um processo e pela amigável interface, proporcionando uma sensação agradável no momento do *design*. O utilizador de BPEL não necessita de conhecimentos em linguagem de programação, porém, um programador poderá obter mais recursos da ferramenta (SILVA, 2010, p. 1).

As ferramentas de BPM permitem modelar os processos de negócio e as mais recentes, suportam a especificação do BPEL, ou seja, é possível transportar o modelo de negócio para uma ferramenta BPEL sem a necessidade de reescrever o processo. As ferramentas de BPEL funcionam como um *Workflow* e dispõem de vários recursos de integração.

O BPEL pode realizar integrações via *Web Services*, seja como cliente (consumindo outro *Web Service*) ou ainda disponibilizar *Web Services* para serem utilizados por outros sistemas. Permite executar scripts SQL acessando uma grande variedade de banco de dados, que facilita em muito os processos de integração, pois pode receber um número menor de parâmetros e acessar as demais informações diretas no banco de dados. Outro recurso também

muito utilizado nas integrações é via troca de arquivos, que permite gerar arquivos em um determinado layout, bem como também realizar a leitura de arquivos texto.

Existem hoje no mercado várias ferramentas para desenvolvimento de aplicações em BPEL, sendo uma delas a ferramenta da Oracle, denominada JDeveloper. O JDeveloper é uma ferramenta visual e completa que contempla em sua solução os principais recursos de integração utilizados no mercado, como já mencionado anteriormente. A ferramenta possui uma interface onde é possível modelar o fluxo de execução do processo de forma fácil e rápida. Para apresentar o BPEL, será descrito, de forma sintética, nos parágrafos a seguir, os passos para criação de um projeto em BPEL.

O processo inicia com criação de um projeto BPEL (figura 2.3), onde é possível definir de que forma será seu comportamento, síncrono, assíncrono ou livre. Para este exemplo será utilizado um modelo livre.

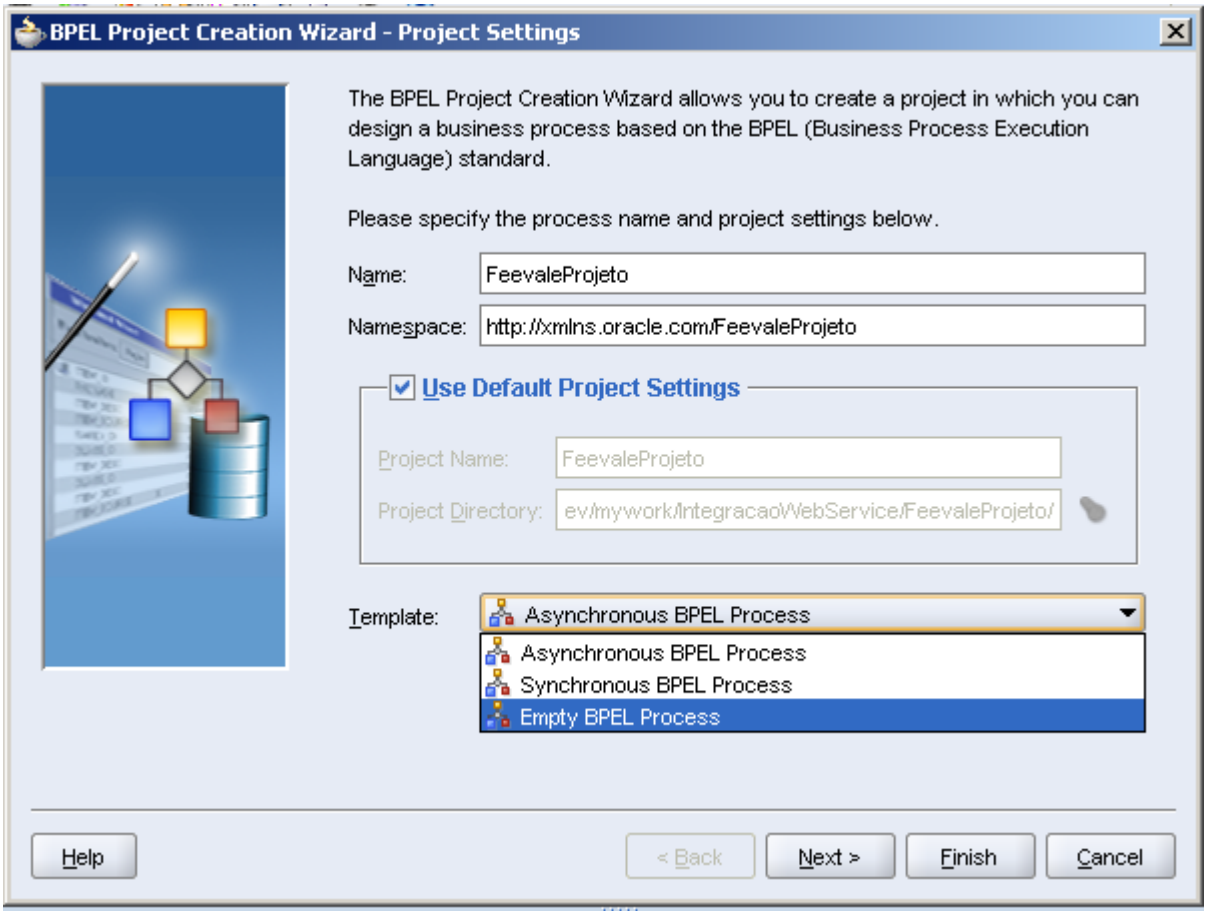

Figura 2.3 – Criação de Projeto em BPEL
Fonte: Próprio autor

Os próximos passos são: realizar a configuração com o banco de dados (figura 2.4) e o desenho do fluxo de execução do processo (figura 2.5), que poderá acessar banco

de dados, *Web Services*, etc. Finalizado o fluxo do projeto, o mesmo está pronto para ser disponibilizado para o uso. Maiores detalhes sobre o desenvolvimento e uso da ferramenta podem ser obtidos no site da Oracle[1].

Os testes realizados permitem constatar que a ferramenta facilita o trabalho de integração, mas o domínio do ambiente BPEL é necessário, para o desenvolvimento da solução de integração.

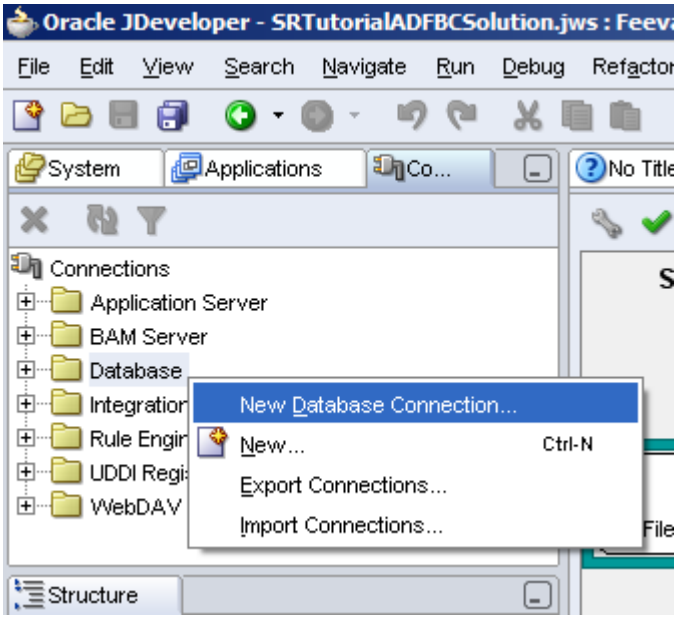

Figura 2.4 – Configuração do banco de dados
Fonte: Próprio autor

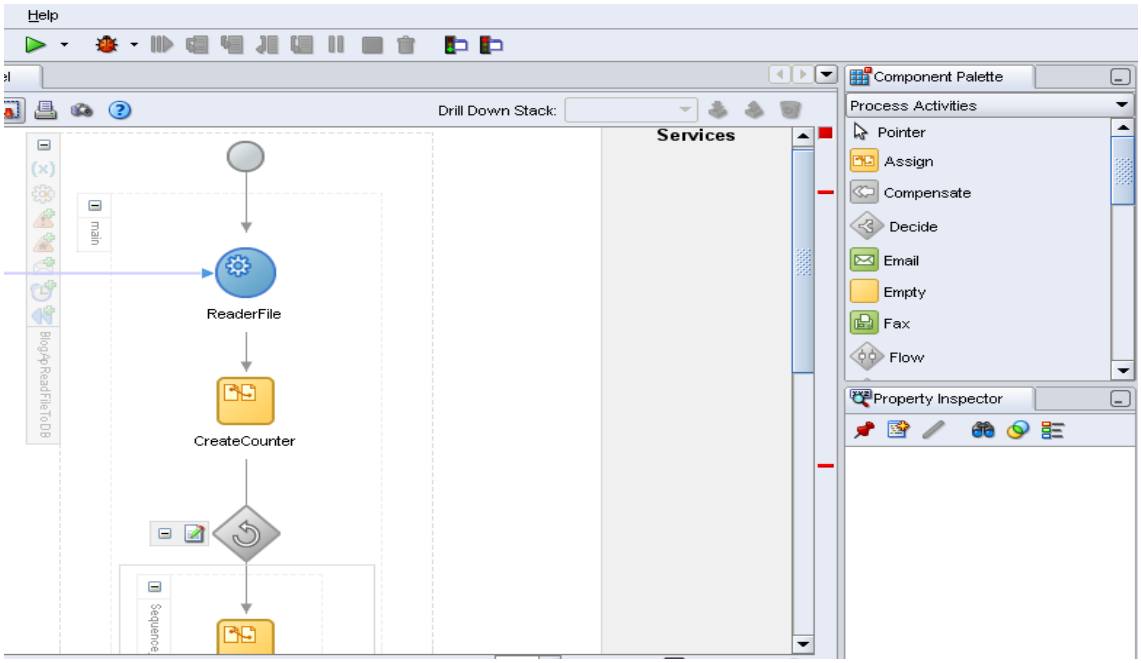

Figura 2.5 – Desenho do fluxo de execução do processo
Fonte: Próprio autor

[1] O site da Oracle com informações de BPEL fica em http://otn.oracle.com/bpel.

2.4 Integração entre banco de dados

A utilização de bancos de dados em diferentes servidores é uma situação comum em uma organização. Vários são os motivos, seja pela necessidade de garantir alto desempenho a consulta de informações, por requisitos do negócio ou até mesmo para garantir a continuidade da operação em caso de falhas de um servidor (contingenciamento). Em todos os exemplos citados, é necessário que exista uma integração para que possam enviar e receber informações. Em se tratando de um mesmo fornecedor, a configuração da integração tende a ocorrer naturalmente. Mas em muitos casos, banco de dados heterogêneos precisam se comunicar para obter ou gerar informações.

Para suprir estas necessidades de integração, empresas fornecedoras de banco de dados, tem investindo continuamente na melhoria dos seus mecanismos de integração entre bases homogêneas como também heterogêneas. O resultado deste trabalho são recursos mais confiáveis e simples de serem configurados, dentre os quais é possível citar o *Heterogeneous Services* da Oracle e o *Linked Server do* SQL Server.

O Oracle *Heterogeneous Services*, denominado simplesmente OHS, é um recurso que estende o uso de *database links*, fornecendo um meio mais simples de acessar bancos de dados de diferentes fabricantes a partir de um banco de dados Oracle. Além de ser usado como uma solução para processos de ETL (Extração, Transformação e Carga de Dados) entre sistemas diferentes, ele é particularmente útil para acessar bancos de dados legados com custos baixos de implementação (CARNEIRO, 2011, p.1).

O *Linked Server* do SQL Server permite estabelecer um canal entre outros bancos de dados, utilizando para isso dois objetos: um provedor OLE DB e uma fonte de dados OLE DB. O provedor OLE DB é uma biblioteca de vinculo dinâmico (DLL) que é capaz de gerenciar e interagir com uma fonte de dados específica. Já a fonte de dados OLE DB é responsável por identificar o banco de dados específico que poderá ser acessado através do OLE DB. (MICROSOFT, 2011, p. 1).

Estes recursos são muito utilizados em ambientes distribuídos, onde os *links* entre banco de dados em servidores separados é um requisito essencial para organização. Segundo Melo (2010, p. 32) a implementação de um banco de dados distribuído pode envolver diversos servidores de banco de dados corporativos que executam operações que sejam capazes de replicarem dados entre os diversos SGBD’s de uma determinada empresa, que podem ou não estarem em lugares distintos.

Também segundo Melo (2010, p. 32), todo banco de dados suporta aplicações clientes locais, assim como também tem a capacidade de se comunicar com outros bancos de dados que façam parte da rede. Se um dos servidores envia a solicitação de banco de dados para outro servidor, o servidor que envia a solicitação funciona como um cliente, e o servidor que recebe deverá ser capaz de executar as instruções SQL que recebeu para depois retornar o resultado positivo ou as condições de erro ao emissor.

2.5 Considerações finais

A evolução das ferramentas de integração estão diretamente ligadas a própria evolução dos softwares e de sua complexidade, mas ainda assim existe espaço para utilização de meios de integração mais antigos mas que atendem a necessidade para determinado contexto de integração. O conhecimento das funcionalidades, capacidades e restrições de cada uma das técnicas é o componente fundamental para contribuir com a escolha da tecnologia, mas não é o único, as necessidades do negócio tem um peso ainda maior nesta definição, ou seja, a urgência da informação pode influenciar no tipo de integração a ser utilizada.

As tecnologias disponíveis no mercado quando utilizadas em sua essência podem resolver problemas de integração, mas o que se percebeu é que a falta de um controle automatizado e até mesmo a ausência de ferramentas de gerenciamento destes processos são riscos que não podem ser mais mitigados. Desta forma, o surgimento de ferramentas desenvolvidas especificamente para integração, reunindo as melhores práticas e disponibilizando recursos que permitam gerenciar toda a cadeia de integrações passaram a ser fundamentais dentro de uma empresa.

3 AMBIENTE DE INTEGRAÇÃO

Durante este trabalho foram apresentadas as técnicas de integração mais utilizadas no mercado, seus conceitos e alguns exemplos de aplicação. Contudo, analisadas isoladamente cada uma atende com excelência determinado contexto de integração, o que pode influenciar sua utilização em integrações que outras tecnologias fossem mais adequadas. A tarefa de definir a integração que mais se ajusta ao contexto é uma atividade difícil. Logo, este trabalho objetiva, além de apresentar algumas técnicas empregadas para solucionar problemas de integração, demonstrar na prática o funcionamento e aplicabilidade destas técnicas, contribuindo com mais informações e, consequentemente, facilitando a tarefa de definição da melhor tecnologia a ser utilizada.

Para tornar o estudo mais interessante, será proposto um cenário de integração que se assemelha com situações reais, mas que seja de fácil entendimento, deixando a complexidade somente para o desenvolvimento das integrações. O cenário proposto consiste de uma empresa do ramo de vendas ao varejo incorporou uma empresa do mesmo ramo. A empresa incorporadora pretende manter o sistema atual da empresa incorporada. Entretanto, a diretoria definiu que os preços a serem praticados pela empresa incorporada deve ser os mesmos praticados pelo grupo. Como o cadastro de produtos compreende cerca de 40 mil itens, a única forma de garantir que não haja divergência entre os cadastros é de que os mesmos sejam integrados sistemicamente.

A empresa incorporada perde a autonomia ao cadastro - incluir e alterar dados - pois os mesmos sempre serão replicados da incorporadora. Os dados sempre que alterados na incorporadora devem ser integrados o mais rápido possível.

3.1 Características do cenário

As informações a serem integradas devem ser enviadas em único sentido, da empresa incorporadora (matriz) para a empresa incorporada (filial), a qual deverá receber as informações e gravar em seu banco de dados. Os servidores de banco de dados envolvidos no processo não estão localizados em uma mesma rede local. Para que possam se comunicar, será utilizada uma VPN (*Virtual Private Network*) existente entre as duas empresas. Em relação aos softwares instalados, a matriz possui em seu servidor de banco de dados o sistema operacional Windows XP e banco de dados SQL Server, já o servidor da empresa filial

possui, sistema operacional Windows 7 e banco de dados Oracle. A figura 3.1 representa o cenário descrito acima.

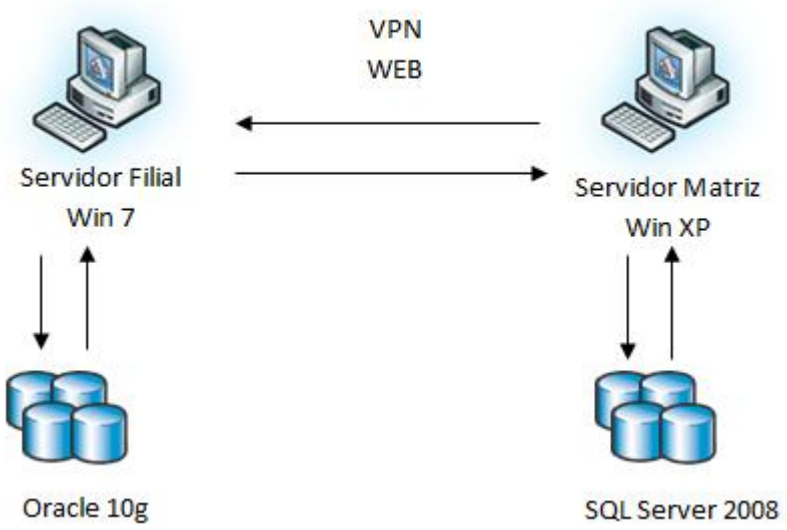

Figura 3.1 – Topologia da rede
Fonte: Próprio autor

Cabe destacar que este cenário prevê a utilização de sistemas operacionais e SGBD's diferentes, como também, conexão de rede diferente de uma rede local. A idéia foi propor um cenário que represente um ambiente heterogêneo.

3.2 Estrutura dos dados a serem integrados

A segunda etapa da definição do cenário é propor estruturas diferentes para os objetos de banco de dados responsáveis por armazenar as informações dos produtos em cada uma das empresas (incorporadora e incorporada ou ainda matriz e filial). Partindo da premissa de que as empresas usam sistemas distintos, serão propostas tabelas e campos distintos entre os dois sistemas.

As entidades responsáveis por manter as informações sobre os produtos são próprias de cada uma das empresas, possuindo nomenclatura e ainda significado diferentes. Desta forma, não há como mapear as informações de uma tabela a outra levando em consideração somente o nome de seus campos, sendo preciso conhecer que informações estão armazenadas em suas estruturas. Para que não ocorram erros ao definir os dados que farão parte da integração, o dicionário de dados tem um papel fundamental, como também a posterior

análise dos dados. A tabela 3.1 apresenta a estrutura da tabela **Produto**, existente no servidor da matriz.

SGBD	SQL SERVER 2008		
SCHEMA	MATRIZ		
TABELA	PRODUTO		
CAMPOS	**TIPO DE DADO**	**TAMANHO**	**DESCRIÇÃO**
COD_PRODUTO	INTEGER		CÓDIGO DO PRODUTO
COD_PRODUTO_FORNECEDOR	NUMERIC	(15, 0)	CÓDIGO DO PRODUTO NO FORNECEDOR (CÓDIGO DE BARRAS)
DES_PRODUTO_FORNECEDOR	VARCHAR	100	DESCRIÇÃO DO PRODUTO NO FORNECEDOR
DES_PRODUTO_CAIXA	VARCHAR	100	DESCRIÇÃO DO PRODUTO NOS TERMINAIS
VLR_VENDA	NUMERIC	(12, 2)	VALOR DE VENDA AO CONSUMIDOR
DT_CADASTRO	DATETIME		DATA DO CADASTRO DO PRODUTO

Tabela 3.1 – Dicionário de Dados da Tabela Produto (SQL Server)
Fonte: Próprio autor

A estrutura da tabela da filial equivalente à tabela **Produto** da matriz é apresentada na tabela 3.2. No servidor da filial a entidade responsável por manter as informações de produto recebe o nome de **Material**.

SGDB	ORACLE 10G		
SCHEMA	ESTOQUE		
TABELA	MATERIAL		
CAMPOS	**TIPO DE DADO**	**TAMANHO**	**DESCRIÇÃO**
CD_MATERIAL	VARCHAR2	20	CÓDIGO DO MATERIAL NO FORNECEDOR (CÓDIGO DE BARRAS)
DS_MATERIAL_FORNECEDOR	VARCHAR2	80	DESCRIÇÃO DO PRODUTO NO FORNECEDOR
VL_VENDA	NUMERIC	(12, 2)	VALOR DE VENDA AO CONSUMIDOR
DT_INCLUSAO	DATE		DATA DO CADASTRO DO MATERIAL
DT_MODIFICAÇÃO	DATE		DATA DE MODIFICAÇÃO

Tabela 3.2 – Dicionário de Dados da Tabela Material (Oracle)
Fonte: Próprio autor

3.3 Definição dos dados a serem integrados

As informações a serem integradas são três:

1. **Identificador do produto:** corresponde ao código do fornecedor (produto.cod_produto_fornecedor). A escolha do campo levou em consideração a existência da mesma informação na tabela Material (cd_material), como também de que o campo único e obrigatório nas duas estruturas;

2. **Valor do material:** corresponde ao valor de venda (vlr_venda);

3. **Descrição do produto:** corresponde a descrição do produto no fornecedor (produto.des_produto_fornecedor).

Durante a definição dos campos da integração se observa que apesar do universo de dados existentes nas estruturas serem equivalentes, o tipo de dados definidos para armazenar as informações em cada SGBD é diferente. Para que a integração possa ser realizada é necessária à conversão dos dados no momento do transporte para que o receptor tenha condição de gravá-los. Neste processo de conversão é importante não haver perda de informações e caso venha a ocorrer, avaliar se estas não comprometem o resultado da integração.

O resultado da análise de domínios como também o método de conversão a ser aplicado foi compilado em um documento denominado dicionário de integração de dados, tabela 3.3.

Emissor - SQL Server			Método de conversão	Cliente Oracle		
Descrição do campo	Tipo de dado	Tamanho		Descrição do campo	Tipo de dado	Tamanho
COD_PRODUTO_FORNECEDOR	NUMERIC	15	Durante o transporte dos dados o valor será convertido de numérico para texto. Este procedimento poderá ser realizado pelas funções de conversão nativas do banco emissor ou até mesmo por funções da aplicação de integração.	CD_MATERIAL	VARCHAR	20
DES_PRODUTO_FORNECEDOR	VARCHAR	100	O tamanho do campo destino é menor que o campo origem, desta forma as últimas 20 posições serão descartadas durante o transporte dos dados. Após análise foi entendido aceitável visto que não há no cadastro nenhum registro com descrição superior a 80 caracteres. Este procedimento poderá ser realizado pelas funções de manipulação de texto nativas do banco emissor ou até mesmo por funções da aplicação de integração.	DS_MATERIAL_FORNECEDOR	VARCHAR	80
VLR_VENDA	NUMERIC	(12,2)	Os dados podem ser transportados sem manipulações, isto porque possuem o mesmo tipo de dado.	VL_VENDA	NUMERIC	(12,2)

Tabela 3.3 – Dicionário de integração de dados (SQL Server x Oracle)
Fonte: Próprio autor

3.4 Alternativas de integração

Este cenário será considerado para avaliar as características de cada uma das tecnologias de integração propostas para serem analisadas neste trabalho. Desta forma, será possível avaliar as alternativas que cada tecnologia dispõe para resolver o mesmo problema de integração. Para explorar as potencialidades de cada uma das tecnologias, nos capítulos posteriores serão detalhados: de que forma se pretende atender a necessidade de integração, o passo a passo para desenvolver a solução, seus resultados e por fim a análise dos pontos de atenção ou riscos identificados durante a execução da integração.

O desenvolvimento dos cenários será iniciado pela integração direta entre bancos de dados, que será apresentada no capítulo 4. Este recurso nativo do SGBD pode ser utilizado para integração de dados entre bancos heterogêneos, que neste caso serão enviadas do SQL Server para Oracle. Na sequência, no capítulo 5 a proposta é integrar as informações utilizando recursos do banco de dados com o apoio do BPEL, que estabelecerá um canal de integração entre os dois SGBD's, transportando as informações de um servidor para outro.

No capítulo 6 será aplicada a integração por intermédio de arquivos digitais também conhecida como EDI. Neste caso, as informações existentes no servidor da matriz, serão gravadas em arquivos e transmitidas para o servidor da filial, que realizará atualização dos dados a partir do processamento do arquivo, sendo todo este procedimento viabilizado pelo BPEL. Por fim, no capítulo 7 a integração será realizada através de um *Web Service* que será disponibilizado na filial. Neste contexto as informações existentes na matriz serão identificadas pela rotina desenvolvida no BPEL, que comunicará com *Web Service* da filial, transmitindo todas as informações que precisam ser integradas. Finalizando este capitulo será possível avaliar os resultados obtidos por cada tecnologia, permitindo que sejam identificadas virtudes, deficiências e potencialidades de cada uma das tecnologias para o cenário proposto.

4 INTEGRAÇÃO DIRETA ENTRE BANCO DE DADOS

A integração direta entre bancos de dados é um recurso nativo dos SGBD's envolvidos nesta integração, onde, a partir da configuração de *link's* é possível executar consultas, inclusões e/ou alterações de dados de um SGBD no outro banco de dados, como se os objetos do outro banco de dados estivessem disponíveis no próprio servidor. Para demonstrar o funcionamento deste recurso, nesta integração a proposta será criar uma *trigger* na tabela de Produto da matriz, que durante sua execução enviará ao banco de dados da filial todos os registros incluídos ou alterados, estabelecendo desta forma uma integração *on-line* de dados. Maiores detalhes deste procedimento são apresentados neste capítulo.

4.1 Detalhamento da proposta de integração

O desenvolvimento do processo de integração utilizando banco de dados terá como base uma conexão estabelecida entre os dois bancos, de tal forma que sempre que houver alterações em informações do cadastro de produtos estas só serão confirmadas para a matriz caso tenham sido atualizadas também no servidor da filial. Para garantir este resultado, será criada no banco da matriz uma *trigger* vinculada a entidade Produto. Durante a execução desta *trigger* serão enviadas estas alterações para o banco da filial. Caso ocorra algum erro durante o processo de atualização, as alterações na matriz não serão gravadas.

A figura 4.1 mostra a representação gráfica deste fluxo de integração. As alterações realizadas no sistema utilizado na matriz são registradas na tabela de Produto, disparando a *trigger*. A *trigger* ao ser executada converterá os dados a serem integrados no formato da tabela de destino e por intermédio de uma *function* existente no banco Oracle, atualizará as informações no cadastro da filial. Havendo falha na atualização, que poderá ser identificada ao avaliar o retorno da *function*, a *trigger* realizará o cancelamento de todas as alterações. Neste caso, será gerada uma exceção que poderá ser percebida pela aplicação, informando ao usuário que os dados não foram salvos. De outra forma exibirá uma mensagem confirmando a alteração.

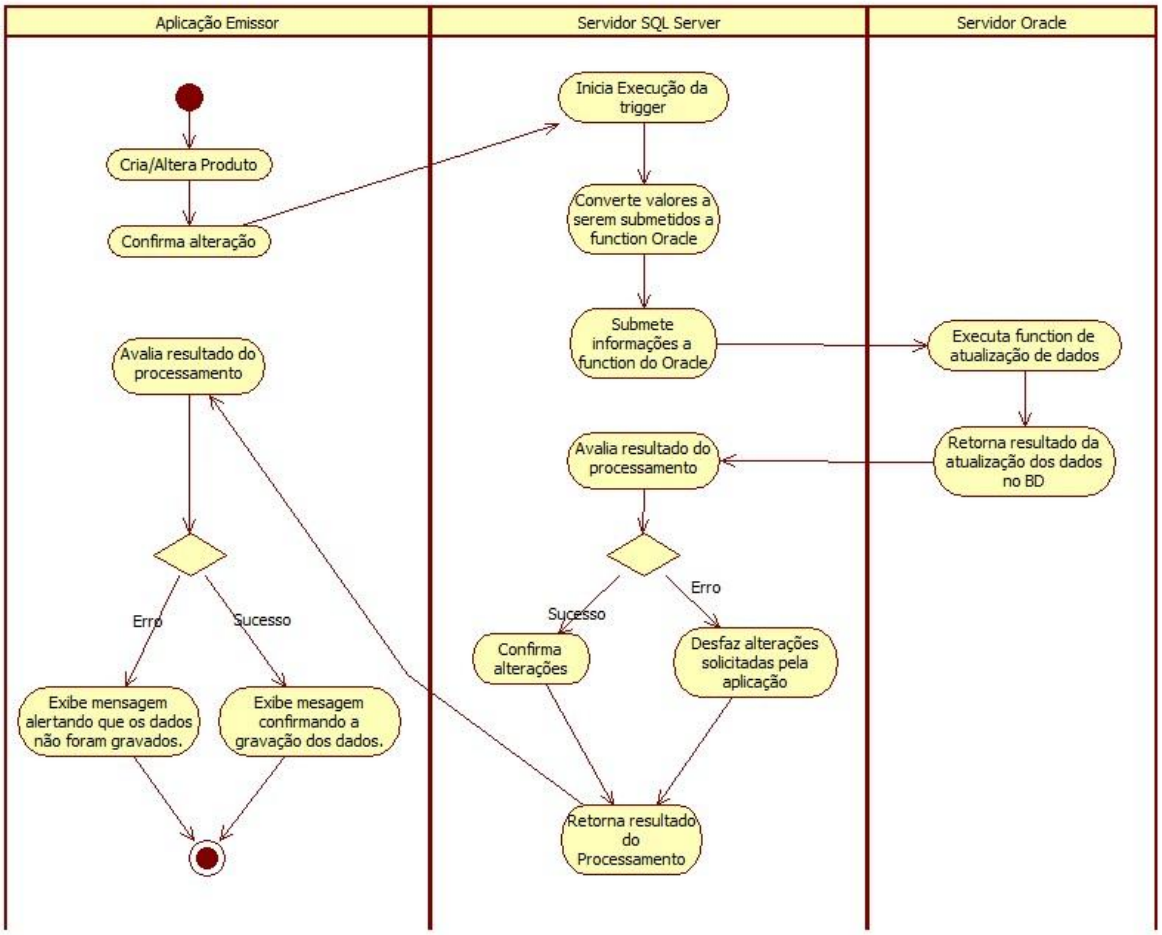

Figura 4.1 – Fluxo de Integração SQL Server x Oracle
Fonte: Próprio autor

4.2 Desenvolvimento

A configuração do ambiente de integração começa pela instalação do *client* do Oracle no servidor emissor[2]. Após sua instalação é necessário configurar o arquivo TNSnames.ora, adicionando as informações para conexão no servidor cliente, como pode ser observado na figura 4.2.

```
1 XE =
2   (DESCRIPTION =
3     (ADDRESS = (PROTOCOL = TCP)(HOST = 192.168.1.2)(PORT = 1521))
4     (CONNECT_DATA =
5       (SERVER = DEDICATED)
6       (SERVICE_NAME = XE)
7     )
8   )
```

Figura 4.2 – Visão parcial do arquivo TnsNames.ora (Oracle)
Fonte: Próprio autor

[2] O download do *client* pode ser realizado diretamente no site da Oracle em http://download.oracle.com/otn/nt/oracle10g/xe/10201/OracleXEUniv.exe

Dando continuidade a configuração do ambiente, no servidor emissor será necessário criar os *links* de acesso ao servidor Oracle, procedimento este executado no SQL Server através de duas *stored procedures*. *A stored procedure* sp_addlinkedserver é responsável por criar o vínculo entre outro servidor de banco, permitindo a realização de consultas e atualizações. A *stored procedure* possui sete parâmetros. Porém, para este trabalho serão utilizados somente quatro que são: @server: nome do servidor a ser vinculado, @srvproduct: nome do produto da fonte de dados, @provider: provedor OLE DB correspondente a fonte da dados remota e o por fim o @datasrc: identificador de banco de dados. O resultado pode ser observado na figura 4.3, onde é apresentado o script a ser submetido ao banco de dados SQL Server.

```
exec sp_addlinkedserver
    @server = 'Oracle',
    @srvproduct = 'Oracle',
    @provider = 'MSDAORA',
    @datasrc = 'XE'
```

Figura 4.3 – SQL para criação do *link* (SQL Server)
Fonte: Próprio autor

Na *stored procedure* sp_addlinkedsrvlogin será detalhado de que modo será feito o acesso ao servidor remoto. Esta *stored procedure* possui cinco parâmetros que são: @rmtsrname: corresponde ao parâmetro @server da *stored procedure sp*_addlinkedserver, @useself: determina se a conexão será realizada com a utilização de *logons* locais ou envio explícito de *logon* e senha, @locallogin: nome do usuário utilizado no servidor local, ou seja no servidor SQL Server, @rmtuser: usuário para *logon* no servidor remoto e por fim @rmtpassword: senha do usuário especificado no parâmetro @rmtuser. O resultado pode ser observado na figura 4.4, onde é apresentado o script a ser submetido ao banco de dados SQL Server.

```
exec sp_addlinkedsrvlogin
    @rmtsrvname = 'Oracle',
    @useself = 'false',
    @locallogin = 'micro1\administrador',
    @rmtuser = 'integracao',
    @rmtpassword = 'dba123'
```

Figura 4.4 – SQL para criação do *login* (SQL Server)
Fonte: Próprio autor

Realizadas as configurações entre os servidores, o próximo passo é criar os objetos de banco que permitirão a comunicação de dados entre os servidores. Para este fim será

desenvolvida uma *function* para manutenção das informações de produtos no servidor Oracle. A *function* recebera três parâmetros, o código do material, a descrição do material e o valor de venda. Ao final de sua execução deverá retornar um valor numérico onde 0 (zero) corresponde a sucesso na operação e -1 erro durante a execução. Esta *function* será consumida durante a execução da *trigger* no servidor SQL Server.

Os procedimentos a serem realizados pela *function* iniciam pela verificação se o material já existe no cadastro da filial. Caso não seja encontrado, as informações recebidas serão incluídas gerando um novo produto. De outra forma, atualizará a descrição do material e o valor de venda. O resultado deste procedimento será retornando ao final de sua execução como já mencionado. O *script* para criação da *function* no banco Oracle, bem como seus procedimentos podem ser observados na figura 4.5.

```
CREATE OR REPLACE FUNCTION "ESTOQUE"."F_CADASTRO_MATERIAL" (p_cd_material material.cd_material%type,
                                                            p_ds_material material.ds_material_fornecedor%type,
                                                            p_vl_venda material.vl_venda%type) return integer
is
  --Desenvolvido por Fabio Fussiger
  --Data: 02/03/2011
  --================================
  i_retorno integer;
  i_aux     integer;
begin

  i_retorno := 0;

  begin

    select count(*)
    into i_aux
    from estoque.MATERIAL
    WHERE cd_material = p_cd_material;

  EXCEPTION WHEN OTHERS THEN
    return -1;
  end;

  if i_aux <= 0 then

    begin

      insert into estoque.MATERIAL (cd_material, ds_material_fornecedor, vl_venda, dt_inclusao, dt_modificacao)
      values(p_cd_material, p_ds_material, p_vl_venda, sysdate, sysdate);

    EXCEPTION WHEN OTHERS THEN
      return -1;
    end;

  else

    begin

      update MATERIAL
      SET ds_material_fornecedor = p_ds_material,
      vl_venda = p_vl_venda,
      dt_modificacao = sysdate
      WHERE CD_MATERIAL = p_cd_material;

    EXCEPTION WHEN OTHERS THEN
      return -1;
    end;

  end if;

  commit;

  RETURN i_retorno;

end;
/
```

Figura 4.5 – Função para atualização de informações do Material (Oracle)
Fonte: Próprio autor

A *trigger* mencionada anteriormente ficará associada à tabela Produto e será configurada para ser disparada somente nos eventos de inclusão e alteração de dados. Durante sua execução realizará as seguintes ações: a primeira delas é avaliar se o valor de venda ou a descrição do produto foi alterada e somente nestas situações realizar a atualização dos dados para filial. Ao identificar que houve alteração no campo valor de venda ou descrição do produto, listará todos os registros alterados e um a um submeterá ao servidor da filial por intermédio da *function* supracitada. Não havendo erros na atualização dos dados na filial as informações serão confirmadas na tabela de Produto. De outra forma serão revertidas. Na figura 4.6 é possível observar o script de criação da *trigger* como também todos os seus procedimentos.

```
USE [matriz]
GO

CREATE TRIGGER [dbo].[TI_PRODUTO] ON  [dbo].[PRODUTO] AFTER INSERT,UPDATE
AS
BEGIN

    SET NOCOUNT ON;

    declare @errno   int
    declare @ll_status int
    declare @ds_msg  varchar(255)
    declare @cd_material varchar(20)
    declare @ds_material varchar(80)
    declare @vl_venda varchar(20)

    declare fila_cursor cursor  for
        select
            convert(varchar(20), i.cod_produto_fornecedor),
            substring(i.des_produto_fornecedor, 1, 80),
            convert(varchar(20), i.vlr_venda)
        from
            inserted i left join deleted d on (i.cod_produto = d.cod_produto)
        where i.vlr_venda <> d.vlr_venda or
        i.des_produto_fornecedor <> d.des_produto_fornecedor
    for READ ONLY

    open fila_cursor

    fetch fila_cursor into @cd_material, @ds_material, @vl_venda
    set @ll_status = @@FETCH_STATUS

    while @ll_status = 0
    begin

        exec (' begin
            ?:= estoque.f_cadastro_material('+@cd_material+', '''+@ds_material+''', '+@vl_venda+');
            END;',@errno OUTPUT) at Oracle;
        select @errno;

        if @errno < 0
        begin
            set @ds_msg = 'Não foi possível gravar as informações do Produto. #1'
            goto   error
        end

        fetch fila_cursor into @cd_material, @ds_material, @vl_venda
        set @ll_status = @@FETCH_STATUS

    end

    close fila_cursor
    deallocate fila_cursor

    return

    error:
        rollback transaction
        raiserror @errno @ds_msg

END
```

Figura 4.6 – *Trigger* da tabela de Produto no Oracle
Fonte: Próprio autor

4.3 Resultados

Para avaliar o cenário de integração, os 40 mil itens existentes no servidor SQL Server foram alterados, gerando por sua vez uma demanda de inclusão/atualização de dados para o servidor Oracle. Os tempos foram medidos a partir do momento que as alterações foram submetidas ao SQL Server, contemplando a chamada da *function* do Oracle e o retorno da sua execução.

As alterações foram realizadas em três momentos distintos e os resultados obtidos foram os seguintes:

1ª. **Execução:** as 40 mil linhas foram integradas com sucesso em 5 minutos e 11 segundos. Levando em média 0,007775 segundos por registro.

2ª. **Execução:** as 40 mil linhas foram integradas com sucesso em 4 minutos e 26 segundos. Levando em média 0,00665 segundos por registro.

3ª. **Execução:** as 40 mil linhas foram integradas com sucesso em 4 minutos e 11 segundos. Levando em média 0,006275 segundos por registro.

4.4 Análise do resultado

Todas as integrações foram realizadas com sucesso, com variações mínimas entre as execuções. O modelo de integração atendeu a necessidade. Porém, dois pontos de atenção devem ser considerados como:

1. **Disponibilidade de rede:** por se tratar de uma integração *on-line*, intermitência ou indisponibilidade impedem a atualização dos dados;

2. **Gerenciamento do processamento:** o baixo ou alto desempenho no atendimento das requisições pode interferir diretamente no processamento do outro banco de dados, podendo causar contenção, ou aumento de processamento.

Caso os riscos levantados sejam aceitáveis pode ser optar por este modelo.

5 INTEGRAÇÃO ENTRE BANCOS DE DADOS COM BPEL (SQL)

Este capítulo apresenta uma segunda proposta de integração, na qual a integração das informações ocorrerá por intermédio da ferramenta BPEL, que terá o papel de orquestrar a coleta de informações no servidor emissor e sua atualização no servidor destino. Ou seja, nesta proposta o acesso a recursos dos bancos de dados envolvidos e a execução dos *scripts* será feita pela ferramenta de BPEL e não de forma direta como na proposta anterior. Como conseqüência, a integração dos dados se dará em um momento posterior a gravação dos dados, diferente da forma *on-line* como ocorria na integração direta banco a banco.

5.1 Detalhamento da proposta de integração

Para o desenvolvimento do processo de integração, será necessário criar algumas estruturas de controle no banco de dados do servidor de origem. Estas estruturas terão o objetivo de sinalizar e armazenar os registros criados ou modificados, como também garantir que nenhuma informação seja perdida. Para sinalizar as informações a serem integradas será necessário criar uma *trigger* e uma tabela de controle. A *trigger* será vinculada à tabela Produto e disparada sempre que houver criação ou manutenção de dados, sendo responsável por registrar as informações relevantes para integração na tabela de controle. Abaixo segue a representação gráfica do processo detalhado acima (figura 5.1).

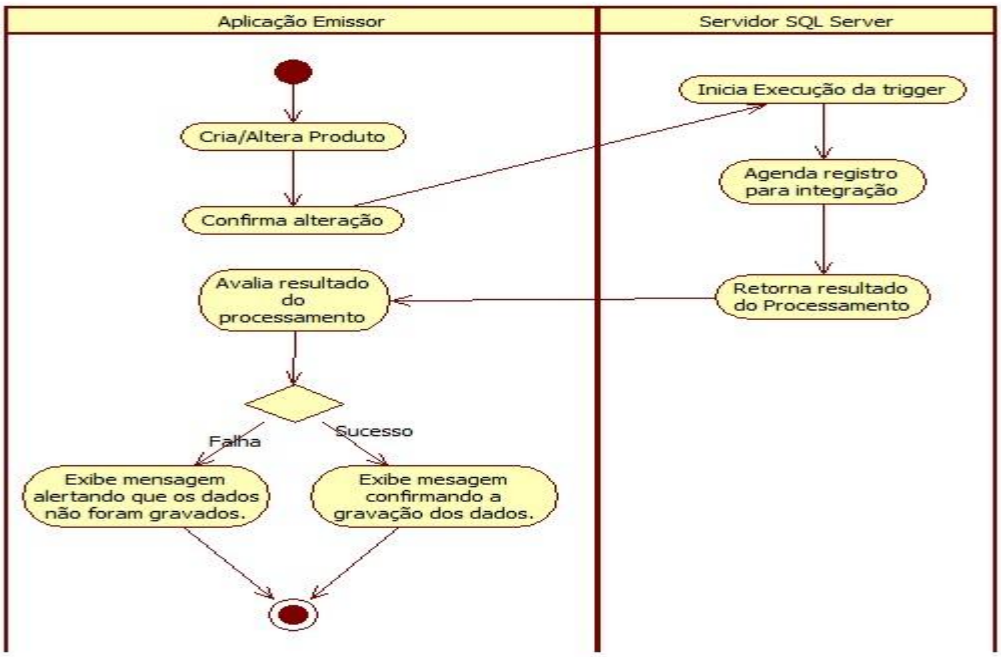

Figura 5.1 – Fluxo para registro de informações a serem integradas (SQL Server)
Fonte: Próprio autor

As informações agendadas na tabela de controle aguardarão a execução do processo de integração que terá o papel de coletar as informações, enviar ao servidor destino e após finalizar seu processamento, atualizar o status no servidor de origem. A recuperação dos dados a serem integrados será realizada em duas etapas. A primeira etapa é identificar todos os registros a serem integrados e alterar seus status na tabela de controle para "Em processamento". A segunda etapa é recuperar da base de dados todos os registros com o status citado anteriormente. Após coletadas as informações, uma a uma será atualizada no servidor destino, ação que será realizada por uma *function* disponibilizada para este fim. Após cada execução, o resultado do processamento será atualizado no servidor origem. A figura 5.2 mostra a representação gráfica de todo o fluxo de atualização dos dados.

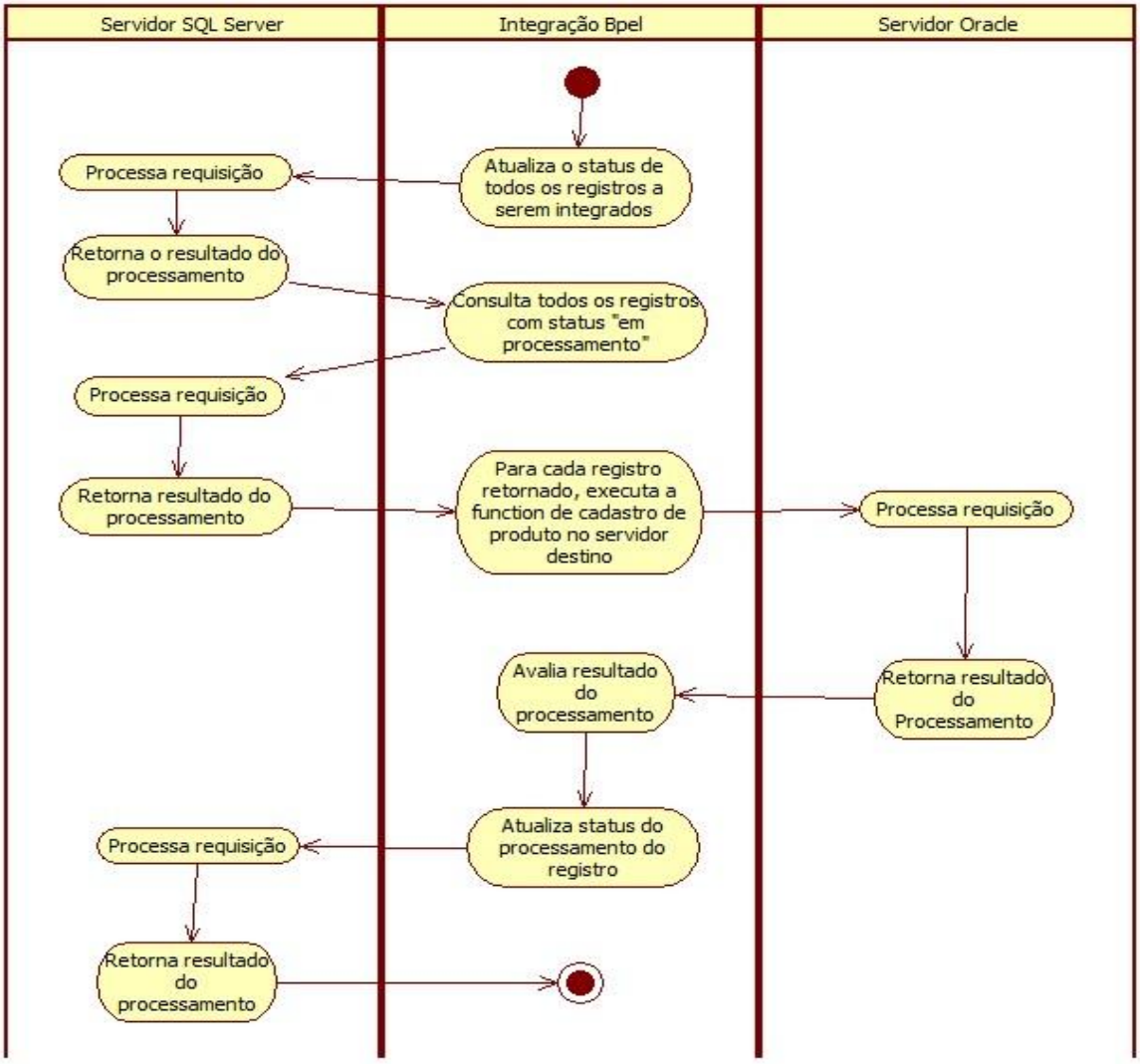

Figura 5.2 – Fluxo de integração de dados com BPEL (SQL)
Fonte: Próprio autor

5.2 Desenvolvimento

A construção desta proposta de integração envolve alterações na estrutura de banco de dados nas duas empresas. A criação dos controles para sinalização das informações a serem integradas foram realizadas no servidor de origem. Para tanto foram criadas duas estruturas para gerenciar os registros a serem integrados. A primeira nomeada como STATUS_PROCESSAMENTO, manterá os status possíveis que possam ocorrer durante todo o processo de integração. A segunda estrutura, nomeada como PRODUTO_CONTROLE, armazenará os registros que sofreram alteração e que devem ser integrados.

A tabela de STATUS_PROCESSAMENTO será composta por três atributos: o código do status de processamento (cod_status_processamento), que será um número sequencial e único, a descrição do status (des_status) e data de cadastro das informações (dt_cadastro). Na tabela 5.1 é possível observar todas as informações relevantes da tabela de STATUS_PROCESSAMENTO.

SGBD	SQL SERVER 2008		
SCHEMA	MATRIZ		
TABELA	STATUS_PROCESSAMENTO		
CAMPOS	TIPO DE DADO	TAMANHO	DESCRIÇÃO
COD_STATUS_PROCESSAMENTO	INTEGER		Indentificador único
DES_STATUS	VARCHAR	50	Descrição do status
DT_CADASTRO	DATETIME		Data de cadastro

Tabela 5.1 – Dicionário de dados da tabela STATUS_PROCESSAMENTO
Fonte: Próprio autor

A tabela PRODUTO_CONTROLE será composta por quatro atributos: o código do produto da tabela Produto (cod_produto), a data de agendamento do registro para integração (dt_inclusao), o status de processamento do registro na integração (cod_status_processamento) e a data de processamento, que será preenchida depois de realizada a integração com o servidor de destino (dt_processamento). Na tabela 5.2 é possível observar todas as informações relevantes da tabela de PRODUTO_CONTROLE.

SGBD	SQL SERVER 2008		
SCHEMA	MATRIZ		
TABELA	PRODUTO_CONTROLE		
CAMPOS	TIPO DE DADO	TAMANHO	DESCRIÇÃO
COD_PRODUTO	INTEGER		Corresponde a chave da tabela PRODUTO
DT_INCLUSAO	DATETIME		Data do agendamento das informações
COD_STATUS_PROCESSAMENTO	INTEGER		Status do processamento dos dados
DT_PROCESSAMENTO	DATETIME		Data em que os dados foram processados

Tabela 5.2 – Dicionário de dados da tabela PRODUTO_CONTROLE
Fonte: Próprio autor

Os scripts gerados para criação das tabelas STATUS_PROCESSAMENTO e PRODUTO_CONTROLE no servidor da matriz são detalhado na figura 5.3. Já na figura 5.4 é possível observar script de carga da tabela STATUS_PROCESSAMENTO, com a lista de status que serão utilizadas neste projeto.

```
1  CREATE TABLE STATUS_PROCESSAMENTO(
2  cod_status_processamento int NOT NULL,
3  des_status varchar(50) NOT NULL,
4  dt_cadastro datetime not null,
5  primary key(cod_status_processamento))
6  go
7
8  CREATE TABLE PRODUTO_CONTROLE(
9  cod_produto int NOT NULL,
10 dt_inclusao datetime NOT NULL,
11 cod_status_processamento integer not null,
12 dt_processamento datetime null,
13 CONSTRAINT FK_produto_staus FOREIGN KEY(cod_status_processamento)
14 REFERENCES STATUS_PROCESSAMENTO(cod_status_processamento))
15 go
16
17 create index IDX1_PRODUTO_CONTROLE on PRODUTO_CONTROLE(cod_produto)
18 go
19
20 create index IDX2_PRODUTO_CONTROLE on PRODUTO_CONTROLE(cod_status_processamento)
21 go
```

Figura 5.3 – *Script* de criação das estruturas de gerenciamento de integração (Oracle)
Fonte: Próprio autor

```
1 insert into STATUS_PROCESSAMENTO values(1, 'Pendente', GETDATE())
2 insert into STATUS_PROCESSAMENTO values(2, 'Em processamento', GETDATE())
3 insert into STATUS_PROCESSAMENTO values(3, 'Processado com Sucesso', GETDATE())
4 insert into STATUS_PROCESSAMENTO values(4, 'Processado com Erro', GETDATE())
```

Figura 5.4 – *Script* de carga de status de processamento no Oracle
Fonte: Próprio autor

O outro objeto a ser criado é a *trigger* que tem por objetivo garantir que nenhuma informação seja perdida. A *trigger* será vinculada a tabela Produto e será disparada sempre

que houver inclusão ou alteração de dados, tendo o papel de registrar na tabela de agendamentos todos os registros a serem integrados.

O primeiro procedimento a ser executado pela *trigger* é verificar se houve alterações nas informações de descrição ou valor de venda do produto. Todos os registros que passaram por alterações em pelo menos um dos campos, devem ser agendados para integração. Para tanto, será verificado o status de processamento na fila de integração de cada registro. Caso o registro não possua agendamento ou possua com status diferente de pendente, um novo agendamento será inserido para o respectivo produto. Este procedimento visa garantir que caso o registro seja alterado novamente e a execução da integração do seu primeiro agendamento ainda esteja em andamento esta nova informação não será perdida sendo encaminhada no próximo ciclo de integração. Na figura 5.5 é possível observar o *script* de criação da *trigger* como também todo o procedimento detalhado acima.

```
USE [matriz]
GO

CREATE TRIGGER [dbo].[TI_PRODUTO] ON  [dbo].[PRODUTO] AFTER INSERT,UPDATE
AS
BEGIN
    -- SET NOCOUNT ON added to prevent extra result sets from
    -- interfering with SELECT statements.
    SET NOCOUNT ON;

    declare @errno   int
    declare @ds_msg  varchar(255)
    declare @cod_produto integer
    declare @ll_aux integer
    declare @ll_status integer

    declare fila_cursor cursor  for
        select i.cod_produto
        from
            inserted i left join deleted d on (i.cod_produto = d.cod_produto)
        where i.vlr_venda <> d.vlr_venda or
        i.des_produto_fornecedor <> d.des_produto_fornecedor
    for READ ONLY

    open fila_cursor

    fetch fila_cursor into @cod_produto
    set @ll_status = @@FETCH_STATUS

    while @ll_status = 0
    begin

        set @ll_aux = 0

        select @ll_aux = count(*)
        from matriz..PRODUTO_CONTROLE
        where cod_produto = @cod_produto
        and cod_status_processamento = 1

        if @@error = -1
        begin
            set @ds_msg = 'Falha ao executar procedimento de agendamento. #1'
            goto   error
        end

        if @ll_aux = 0
        begin

            insert into matriz..PRODUTO_CONTROLE (cod_produto, dt_inclusao, cod_status_processamento)
            values(@cod_produto, getdate(), 1)

        end

        if @@error = -1
        begin
            set @ds_msg = 'Falha ao executar procedimento de agendamento. #2'
            goto   error
        end

        fetch fila_cursor into @cod_produto
        set @ll_status = @@FETCH_STATUS

    end

    close fila_cursor
    deallocate fila_cursor

    return

    error:
        rollback transaction
        raiserror @errno @ds_msg

END
GO
```

Figura 5.5 – *Script* de criação da *trigger* vinculada a tabela Produto (Oracle)
Fonte: Próprio autor

Para o desenvolvimento das interfaces de integração com BPEL será necessário instalar dois aplicativos: o JDeveloper[3] (ferramenta de desenvolvimento para aplicações SOA) e o SOA Suite[4] (servidor onde serão disponibilizados os processos já concluídos para utilização). As integrações serão desenvolvidas no JDeveloper. Mas, para que possam ser executadas, estas devem ser publicadas no servidor SOA, momento em que passam a estar a disposição para utilização.

Dando continuidade ao desenvolvimento do cenário de integração, a próxima atividade a ser realizada é a configuração da conexão com o banco SQL Server e Oracle, que servirá tanto para este projeto como para os demais. A configuração da conexão é realizada no JDeveloper na guia *Connections*, onde deverá ser incluída uma nova conexão de banco de dados e informados todos os dados necessários para conexão, como o nome do servidor de banco de dados, usuário, senha, etc.. Para a configuração da conexão com SQL Server se faz necessário a utilização do driver JDBC (*Java Database Connectivity*) específico do SQL Server que pode ser obtido no site da Microsoft[5]. Já a configuração da conexão com o Oracle é realizada sem a necessidade de baixar qualquer tipo de driver. Na figura 5.6 é possível observar a tela de criação de novas conexões do JDeveloper.

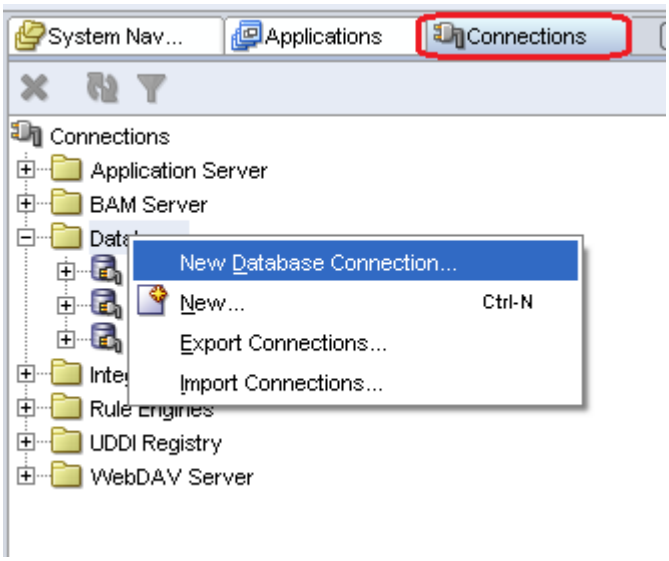

Figura 5.6 – Adição de nova conexão de banco de dados
Fonte: Próprio autor

[3] A versão do JDeveloper utilizada foi a 10.1.3.3.0.
[4] A versão do SOA Suite utilizada foi 10.1.3.1.0.
[5] O download do driver JDBC pode ser realizado diretamente no site da Microsoft em http://www.microsoft.com/downloads/details.aspx?FamilyID=a737000d-68d0-4531-b65d-da0f2a735707&displayLang=pt-br

As figuras 5,7 e 5.8 ilustram as etapas de configuração das conexões com os banco de dados citada acima, sendo apresentadas, respectivamente, a tela de configuração da conexão com o SQL Server e a tela de configuração da conexão com o Oracle.

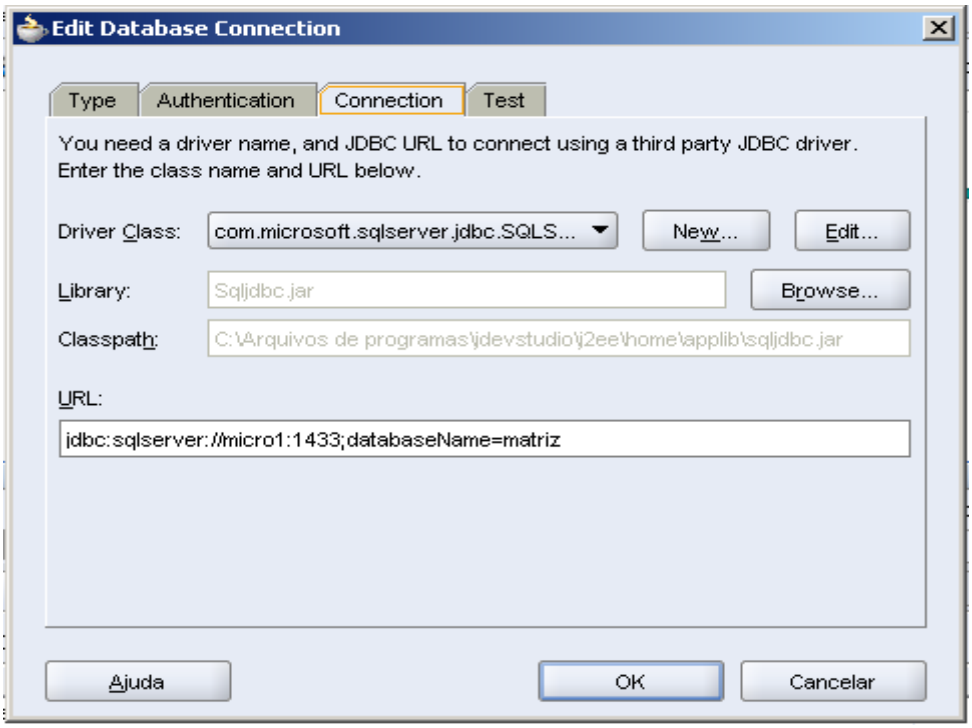

Figura 5.7 – Informações de configuração para o banco SQL Server
Fonte: Próprio autor

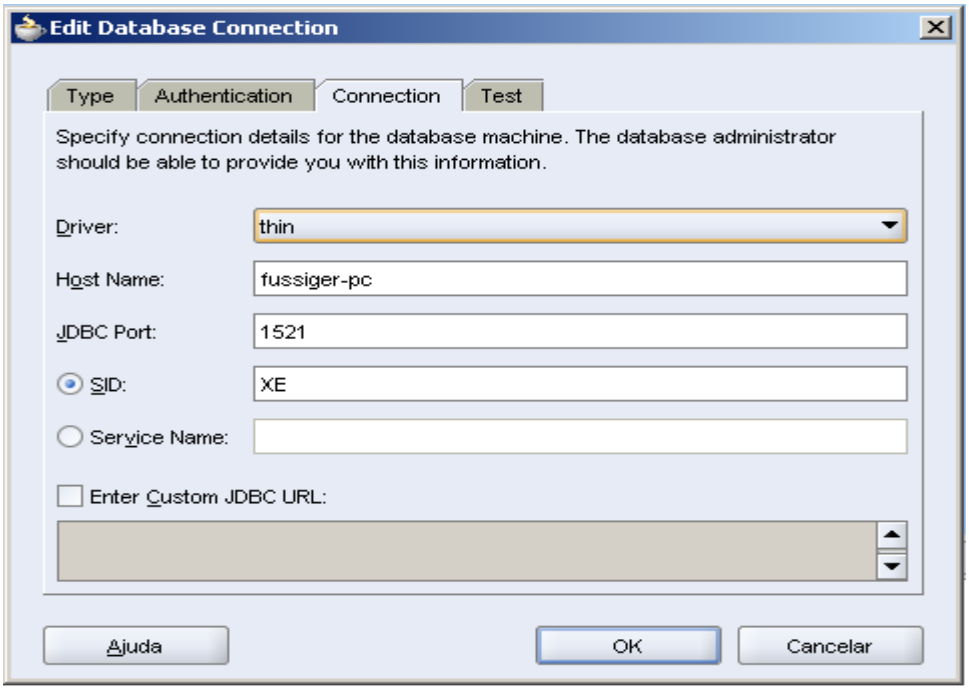

Figura 5.8 – Informações de configuração para o banco Oracle
Fonte: Próprio autor

Realizada a configuração das conexões com os bancos de dados, inicia-se o desenvolvimento do cenário, onde para cada proposta de integração será criado um projeto na ferramenta de BPEL. O projeto será criado com o nome IntgBancoDeDadosBpel e no campo *template* será mantida a opção *Asynchronous BPEL Process* (figura 5.9).

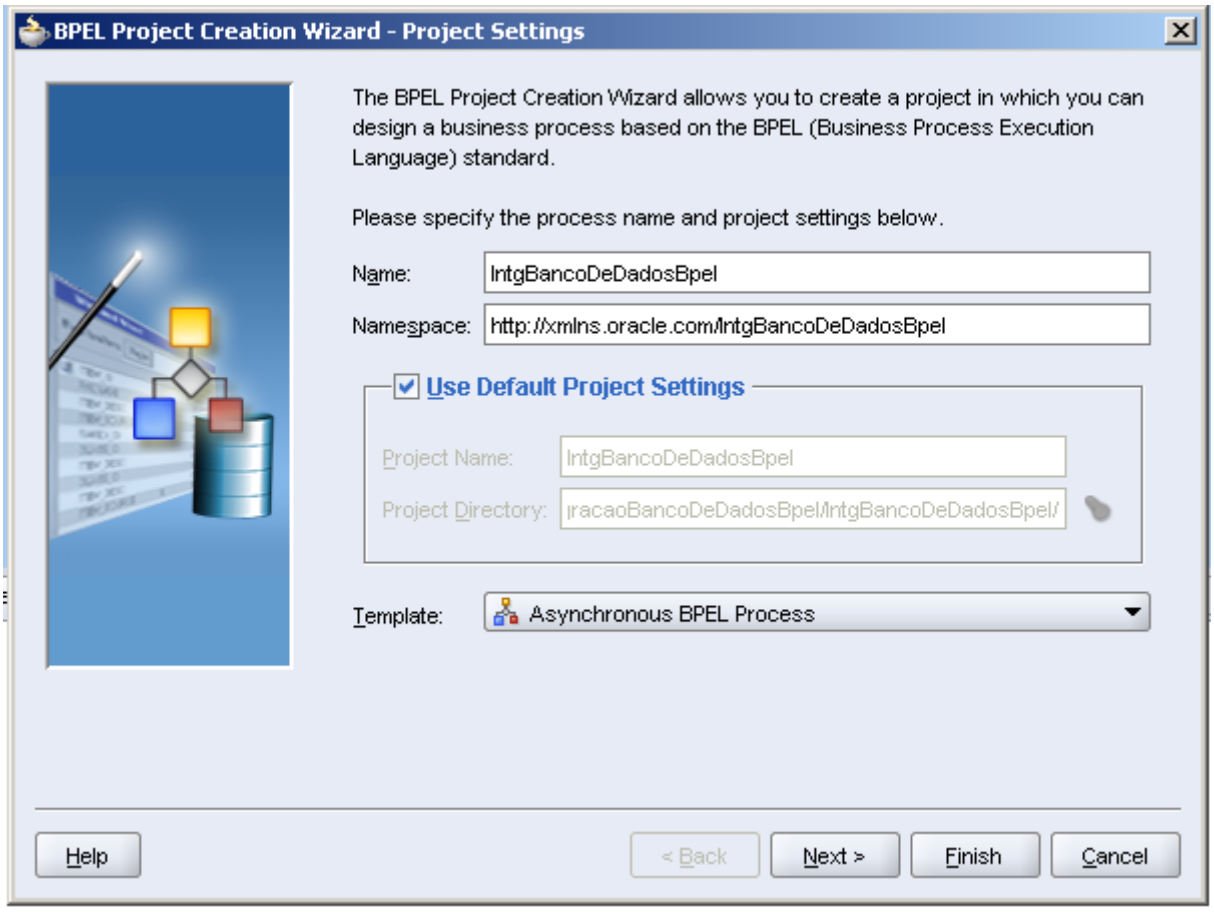

Figura 5.9 – Tela de criação de projeto do JDeveloper (BPELSQL)
Fonte: Próprio autor

Após criado o projeto, a ferramenta irá apresentar o fluxo do projeto com apenas dois componentes (figura 5.10), *receiveInput* e *callbackClient* que são responsáveis, respectivamente, por receber os dados passados como parâmetro (quando houver) e devolver o resultado do processamento. Todo o desenvolvimento necessário para criar o cenário de integração será adicionado entre estes dois componentes.

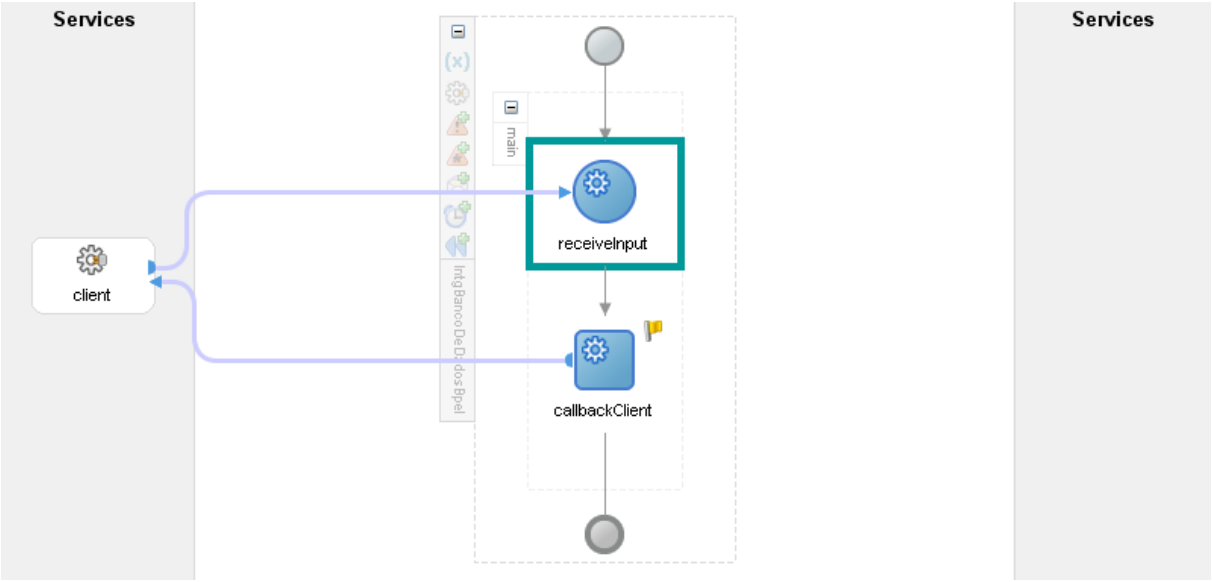

Figura 5.10 – Fluxo de processo no JDeveloper (BPELSQL)
Fonte: Próprio autor

Conforme previsto no projeto desta proposta de integração (figura 5.2), a primeira ação consiste em atualizar o status de todos os registros ainda não integrados que se encontram no servidor de origem de “Pendente” para “Em Processamento”. Para realizar este procedimento será utilizado um componente chamado *DatabaseAdapter*, disponível na classe *Services*. O componente *DatabaseAdapter* estabelece um canal entre a aplicação e o banco de dados, permitindo a execução de *scripts*. Durante sua configuração, deverá ser informado SQL que o componente irá executar para atualizar os dados. A figura 5.11 ilustra a etapa onde é fornecido o script que será executado.

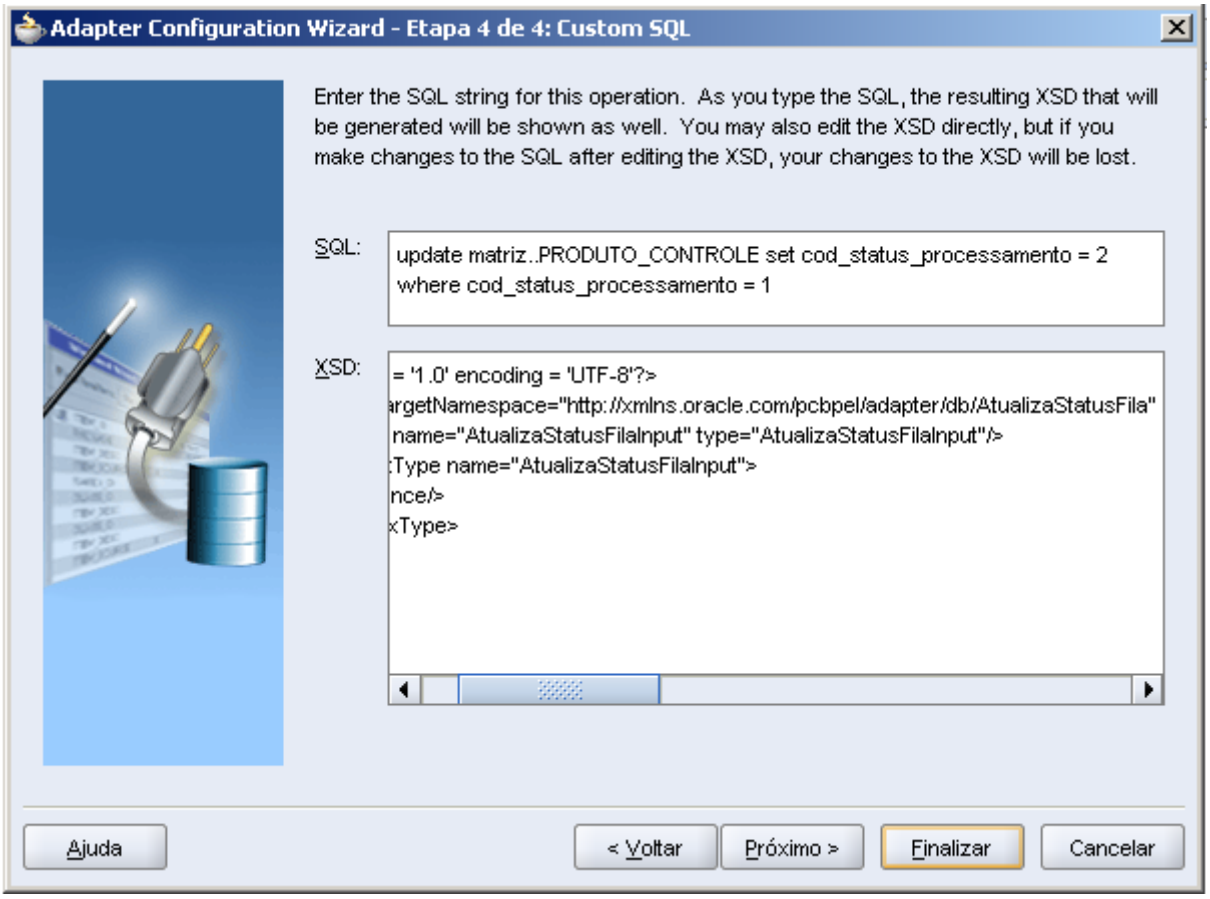

Figura 5.11 – Configuração do componente *DatabaseAdapter*
Fonte: Próprio autor

Depois de configurado o *DatabaseAdapt*er e definido seu nome como AtualizaStatusFila, é necessário incluir um novo componente que terá o objetivo de disparar o procedimento contido no *DatabaseAdapter*. Este componente chama-se *invoke* e deve ser adicionado ao fluxo entre os componentes *receiveInput* e *callbackClient*. O componente está disponível na classe *Process Activities* e após adicioná-lo ao fluxo é necessário associá-lo ao componente *DatabaseAdapter*. Na figura 5.12 é possível observar o fluxo após a adição dos componentes.

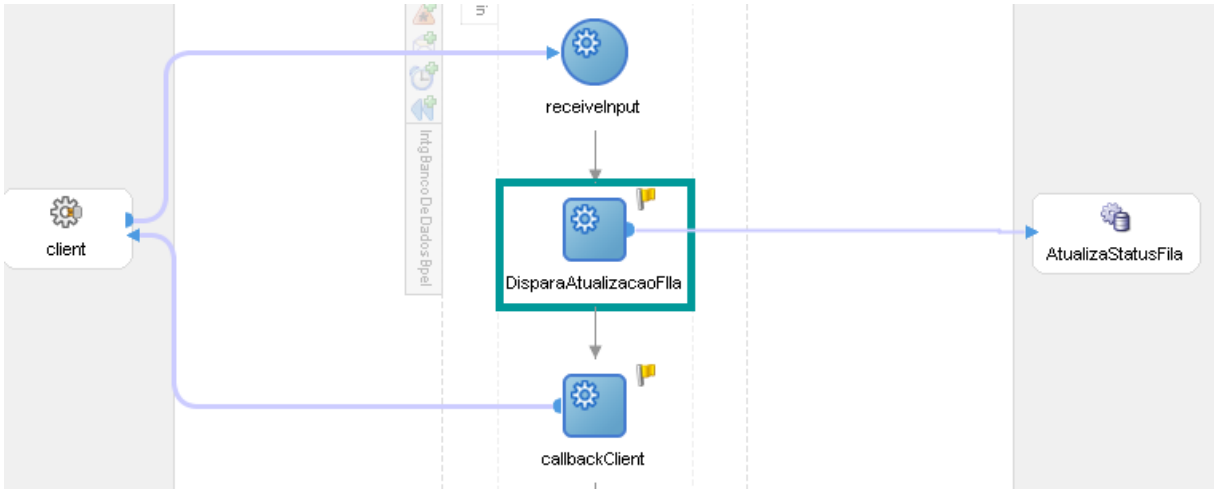

Figura 5.12 – Fluxo de processo no JDeveloper (BPELSQL)
Fonte: Próprio autor

Criado o procedimento para atualização das informações, a próxima ação prevista é obter do banco de dados todos os registros a serem integrados. Para isso, será utilizado novamente um componente *DatabaseAdapter*. O SQL a ser executado pelo componente será fornecido já com as conversões necessárias para atualização no servidor destino, conforme previsto na proposta do projeto, não havendo necessidade de outras conversões. A figura 5.13 mostra o *script* a ser submetido ao banco de dados.

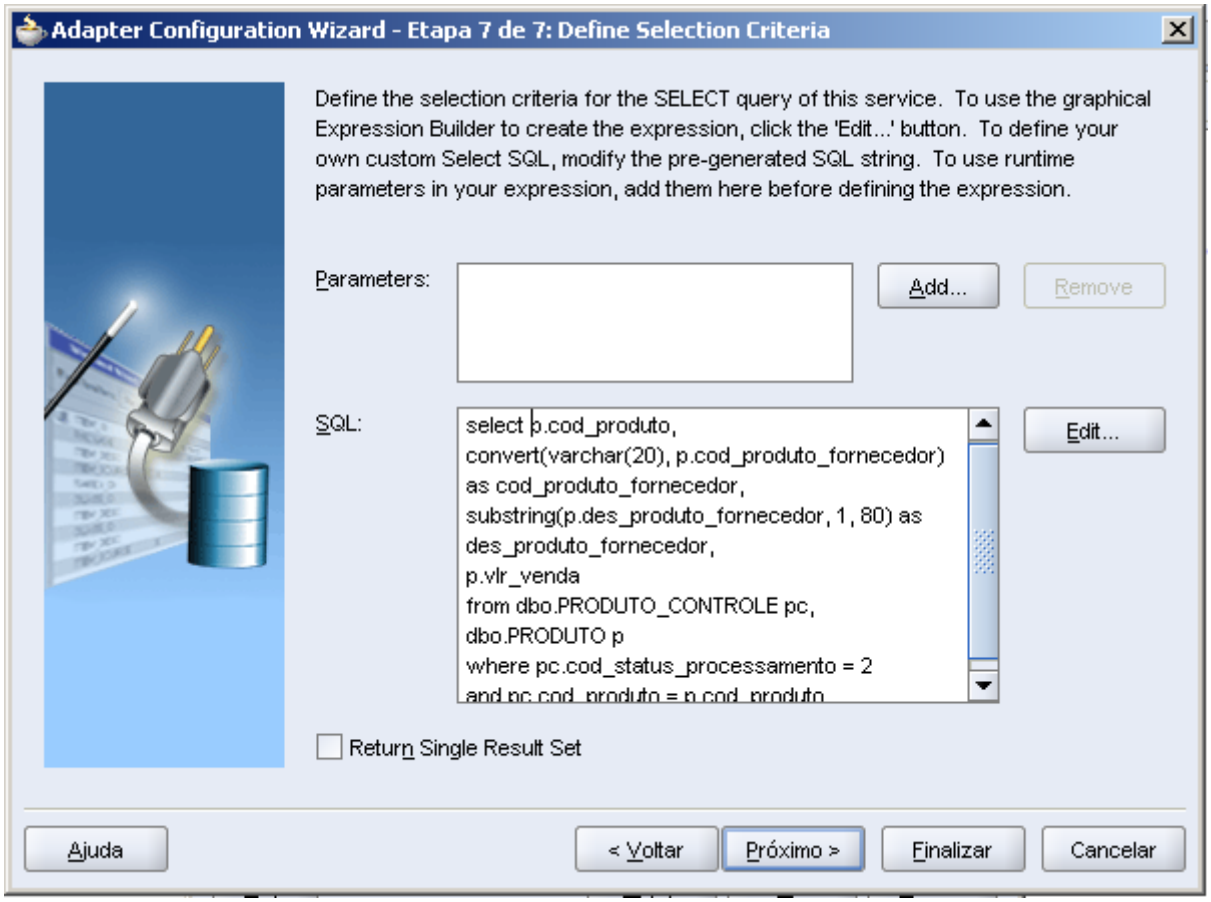

Figura 5.13 – Configuração do componente *DatabaseAdapter*
Fonte: Próprio autor

Configurado o *DatabaseAdapter*, que recebeu o nome de ColetaDadosPendentes, é necessário adicionar o componente que irá executá-lo. Para isso, foi adicionado outro componente *invoke* que, além de atuar como gatilho para execução das instruções do *DatabaseAdapter*, também armazenará temporariamente as informações obtidas do banco de dados. Estas informações são mantidas em uma variável criada dentro do componente definida como variável de saída. A figura 5.14, mostra algumas configurações do componente com destaque para execução do *DatabaseAdapter*.

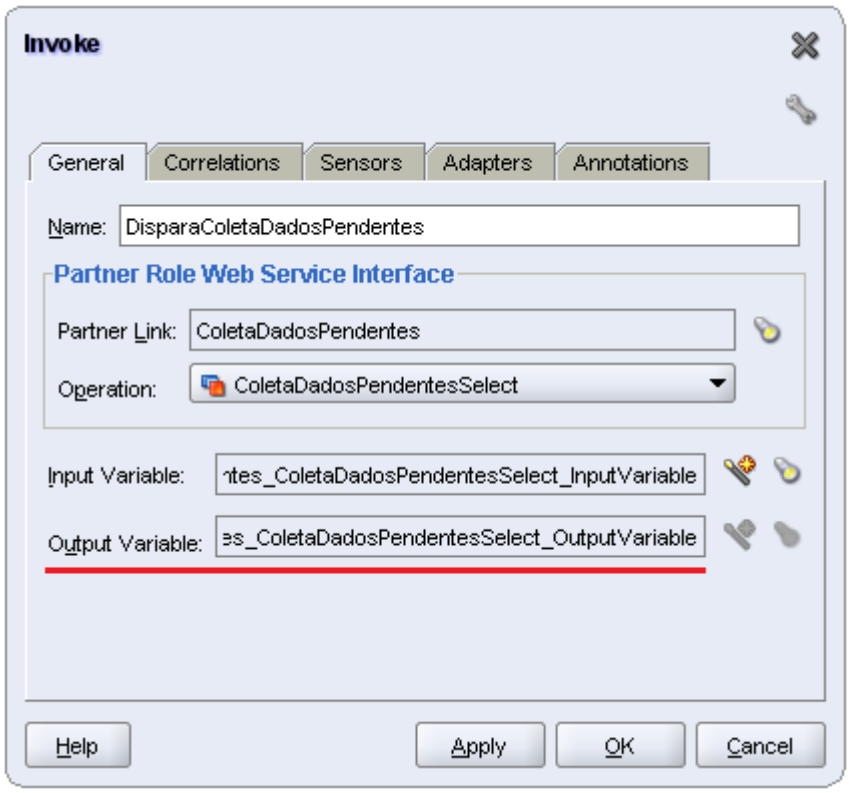

Figura 5.14 – Configuração do componente *Invoke*
Fonte: Próprio autor

Após a execução da consulta, as informações ficarão disponíveis na variável de saída do componente *invoke*, que armazenará tantas linhas quantas retornarem. O próximo passo é processar todas as linhas existentes na variável. Para isso será incorporado ao fluxo o componente *while*. O componente *while* aliado a uma variável que armazena a linha que está em processamento permitirá percorrer todas as linhas retornadas.

A cada linha retornada duas ações devem ser realizadas. A primeira é atualizar os dados no servidor destino. A segunda é atualizar o status na fila de integração de acordo com o resultado obtido na execução da primeira ação. Estas atividades são sequenciais e devem ser realizadas nesta ordem, sendo necessário para tanto adicionar mais alguns componentes ao fluxo.

Levando em consideração a necessidade apresentada, o próximo componente a ser adicionado é o a*ssign*. Este componente será carregado a cada ciclo com um conjunto de informações que correspondem à linha que está sendo processada. Para que estas informações possam ser atualizadas no servidor de destino, outros dois componentes serão adicionados. Um *invoke* que receberá do componente *assign* os dados a serem integrados e outro

componente *DatabaseAdapter*, que será disparado pelo *invoke*, iniciando assim a atualização das informações no servidor destino.

O novo *DatabaseAdapter*, nomeado de AtualizaDadosDestino, recebe as informações e realiza a atualização dos dados no servidor destino. Para isso, ao contrário do procedimento que estava sendo praticado até então, o *DatabaseAdapter* será mapeado para chamar uma *function* (objeto do banco de dados) disponível no banco Oracle. Para realizar este procedimento será utilizada a *function* f_cadastro_material que já havia sido mencionada anteriormente ao descrever o processo de integração do item 4.2. Na figura 5.15 é possível observar o mapeamento da *function* no componente *DatabaseAdapter*.

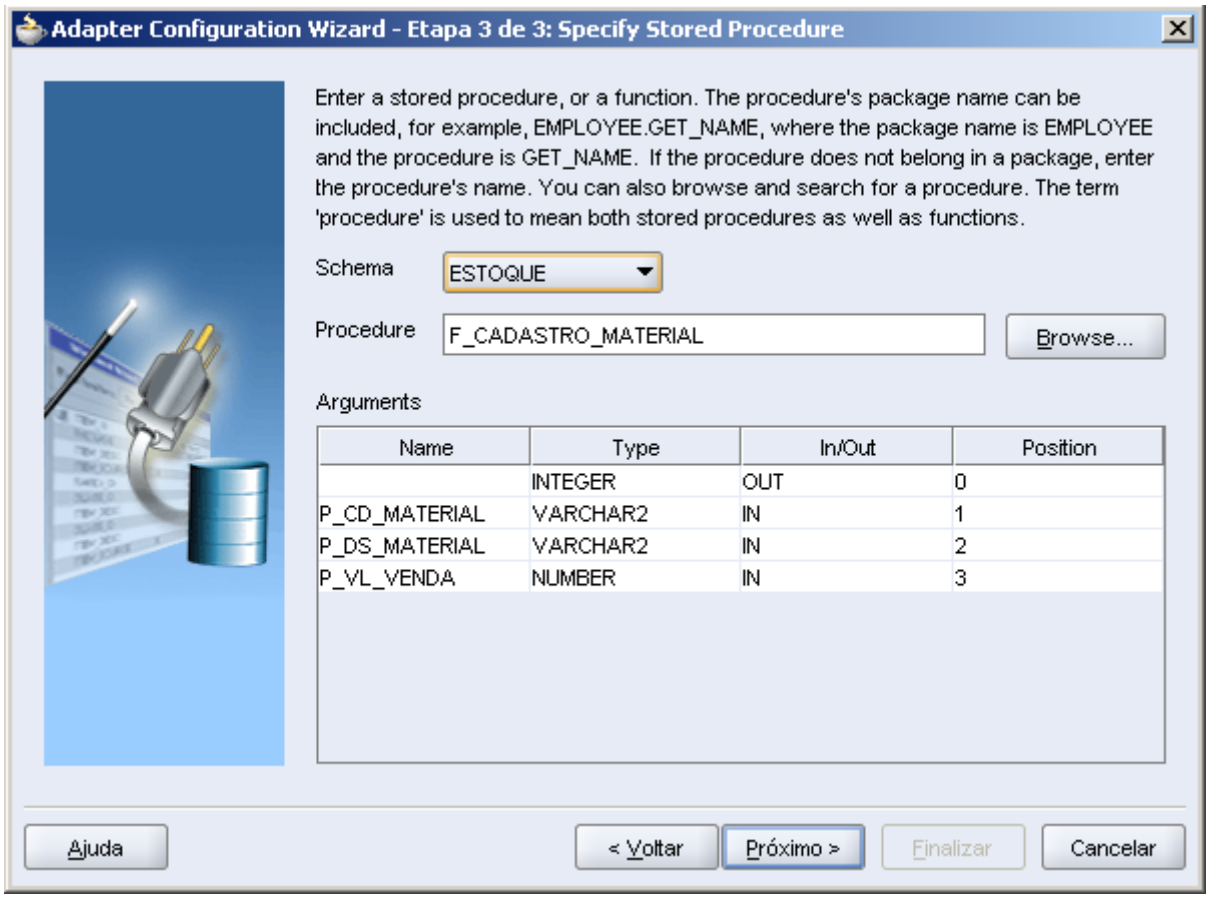

Figura 5.15 – Configuração do componente *DatabaseAdapter*
Fonte: Próprio autor

Na figura 5.16 é possível visualizar a parte do fluxo do processo que contempla estes últimos componentes citados (*while*, *assign*, *invoke* e *DatabaseAdapter*).

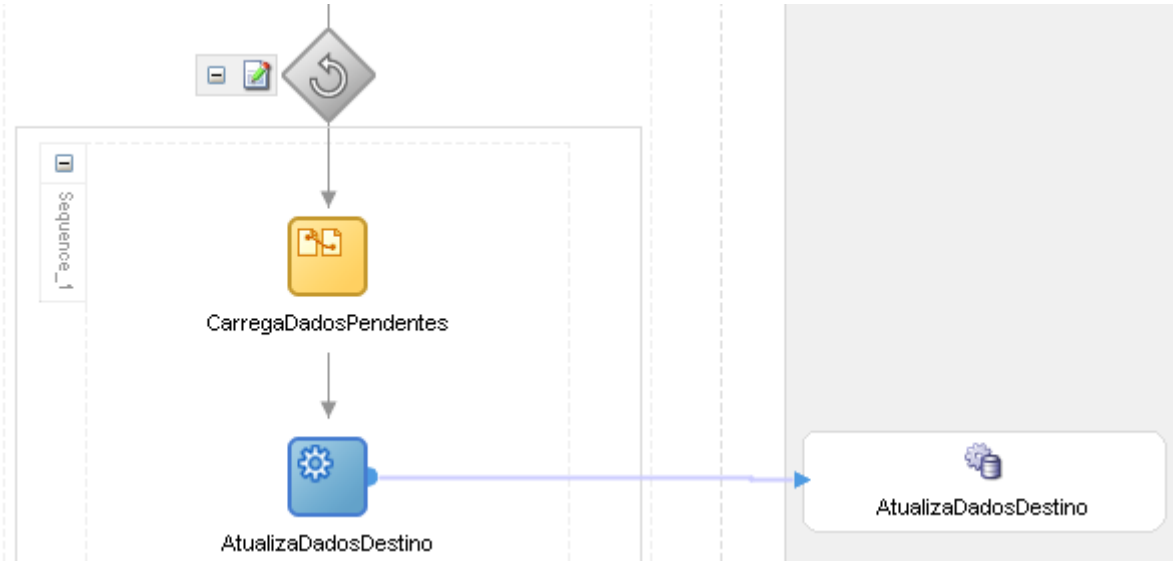

Figura 5.16 – Fluxo de processo no JDeveloper (BPELSQL)
Fonte: Próprio autor

O resultado da atualização submetida ao servidor de destino através do componente *DatabaseAdapter* será carregado na variável de saída do componente i*nvoke*. Com isso, será atualizado o status da fila de integração, permitindo confirmar a atualização dos dados no destino ou não.

A atualização do status será realizada pelos componentes *DatabaseAdapter* e *invoke*. Mas, para que isso ocorra será necessário antes interpretar qual foi o resultado da integração. Isto porque com o retorno da execução de integração apenas é possível identificar se os dados foram integrados com sucesso ou não. Mas para atualizar a fila é preciso ter o código correspondente a sucesso ou falha. Para interpretar o resultado obtido será necessário adicionar um novo componente ao fluxo, chamado *Switch*. Este componente irá avaliar o retorno capturado pelo objeto *invoke*, vinculado ao *DatabaseAdapter* AtualizaDadosDestino, e de acordo com o resultado da atualização encaminhará ao outro componente *invoke*, este associado ao procedimento de atualização do status da fila o código equivalente.

Após a atualização do status na fila, será iniciado um novo ciclo de atualizações que só será encerrado quando o último registro tiver sido processado. O fluxo completo do processo pode ser observado na figura 5.17.

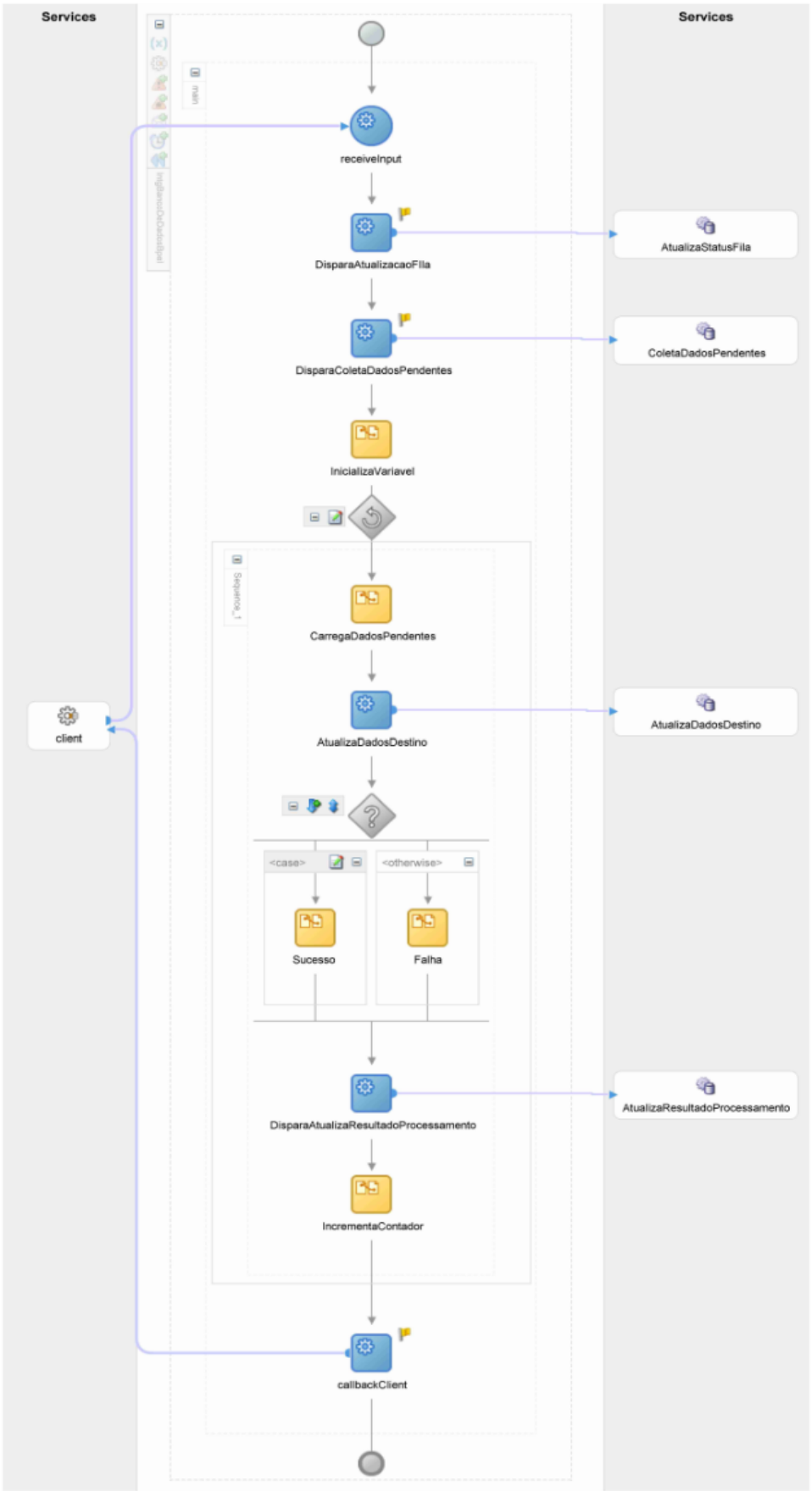

Figura 5.17 – Fluxo completo do processo no JDeveloper (BPELSQL)
Fonte: Próprio autor

5.3 Resultados

Para que seja possível executar o processo de integração é preciso disponibilizá-lo no servidor SOA BPEL. Para tanto, se faz necessária à execução de duas atividades. A primeira é compilar o projeto e gerar o arquivo JAR (*Java Archive*), Este procedimento é realizado selecionando o projeto e marcando a opção *Make*, como demonstrado na figura 5.18. O JDeveloper irá compilar todo o projeto e não havendo erros irá gerar um arquivo JAR.

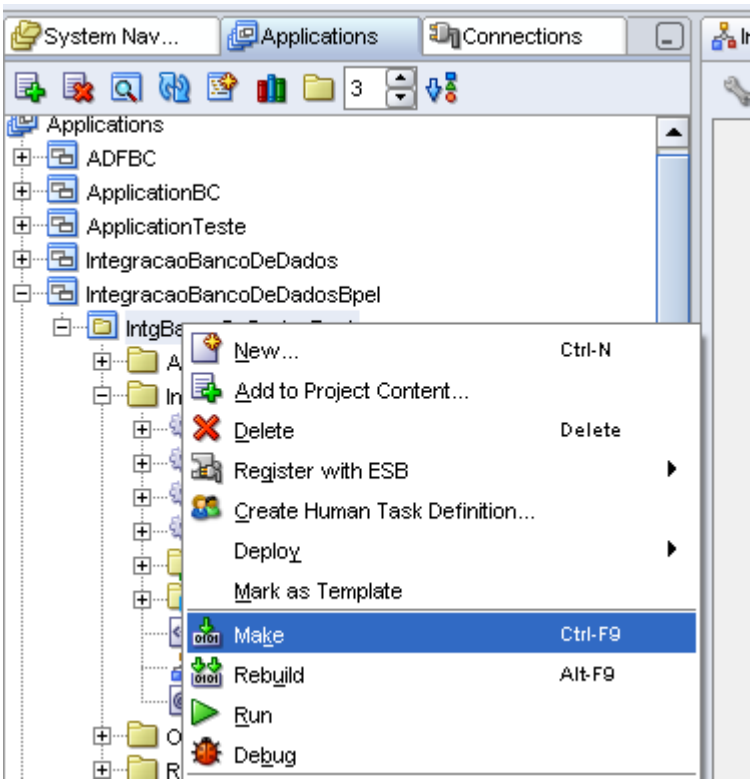

Figura 5.18 – Exportando projeto BPEL
Fonte: Próprio autor

A segunda atividade é publicar o arquivo JAR no servidor SOA BPEL. Para a publicação do projeto é necessário abrir a interface de gerenciamento do servidor SOA BPEL[6]. Com a aplicação aberta seleciona-se a opção "Disponibilizar novo processo". O sistema solicitará o caminho onde se encontra o JAR e depois basta confirmar. O processo publicado pode ser acessado na guia Processos BPEL, como pode ser observado na figura 5.19. Com o processo disponível no servidor já é possível executar o cenário de integração.

[6] Acessa pelo *browser* em http://micro1:8888/BPELConsole

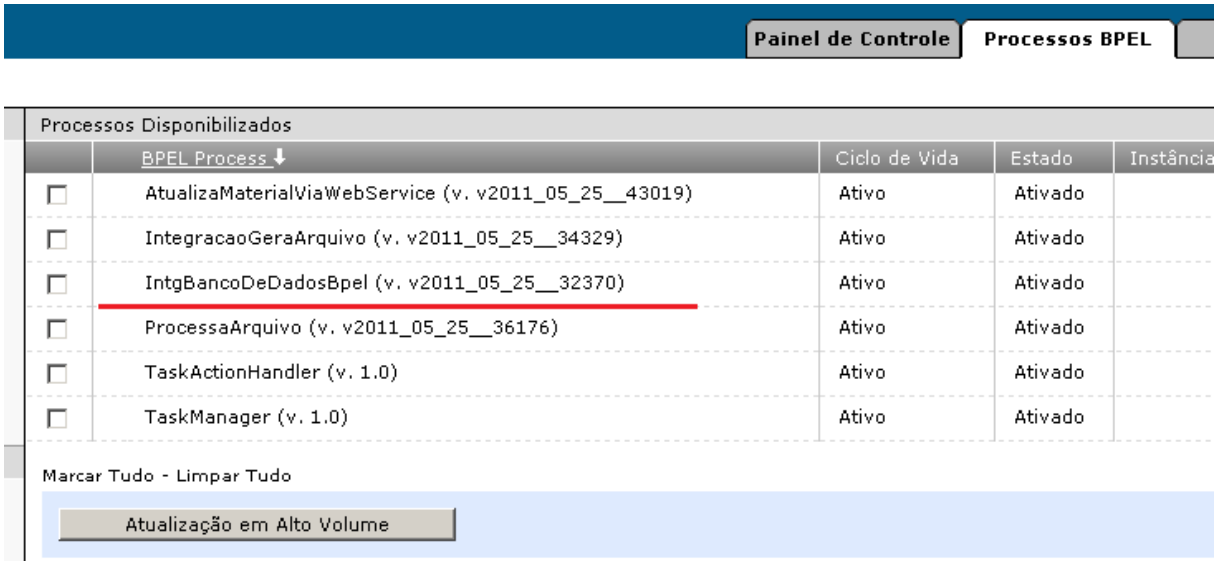

Figura 5.19 – Console Oracle BPEL
Fonte: Próprio autor

Para avaliar o fluxo, os 40 mil itens existentes no servidor SQL Server foram alterados, gerando por sua vez uma demanda de inclusão/atualização de dados para o servidor Oracle. Todas as informações alteradas serão agendadas na tabela de integração e ficarão aguardando a execução do processo de integração para que possam ser encaminhadas ao servidor de destino.

Para que seja possível comparar os modelos de integração ao final do trabalho, será considerado para este cenário que o processo de integração na matriz executará em intervalos de 10 minutos. Ou seja, na pior das hipóteses o registro após ter sido agendado para integração aguardará até dez minutos para que seja processado. Desta forma além do tempo gasto no processo de integração será acrescido o intervalo de tempo entre cada execução da integração.

As alterações realizadas para avaliar o cenário, foram realizadas em três momentos distintos e os resultados obtidos foram os seguintes:

1ª. **Execução:** as 40 mil linhas foram integradas com sucesso em 2 horas, 48 minutos e 48 segundos, onde destes 10 minutos correspondem ao intervalo entre cada execução da integração e 2 horas, 38 minutos e 48 segundos o tempo de processamento da integração. Levando em média 0,2532 segundos por registro.

2ª. **Execução:** as 40 mil linhas foram integradas com sucesso em 2 horas, 49 minutos e 4 segundos, onde destes 10 minutos correspondem ao intervalo entre cada

execução da integração e 2 horas, 39 minutos e 4 segundos o tempo de processamento da integração. Levando em média 0,2536 segundos por registro.

3ª. **Execução:** as 40 mil linhas foram integradas com sucesso em 2 horas, 19 minutos e 22 segundos, onde destes 10 minutos correspondem ao intervalo entre cada execução da integração e 2 horas, 9 minutos e 22 segundos o tempo de processamento da integração. Levando em média 0,20905 segundos por registro.

5.4 Análise do resultado

Todas as integrações foram realizadas com sucesso. Ao comparar o tempo das duas primeiras execuções a variação foi mínima. Já na terceira execução pode se observar uma melhoria de desempenho.

Os resultados obtidos pelo modelo de integração atenderam a necessidade, mas cabe ressaltar quatro pontos de atenção que devem ser considerados ao optar por este modelo, são eles:

1. **Disponibilidade de rede:** por se tratar de um modelo onde existe acesso direto a base de dados do servidor de destino, intermitência ou interrupção podem atrasar a conclusão da atualização de dados;
2. **Tempo da integração:** para que este modelo de integração fosse implementado uma nova camada de software foi adicionada entre os bancos de dados, o SOA BPEL. Com isso é perceptível o aumento no tempo de integração, ou seja, entre obter o dado a ser integrado, enviar ao servidor destino e atualizar o status deste na origem alguns centésimos de segundos a mais foram necessários;
3. **Atualização *off-line***: como os dados são alterados e integrados somente num segundo momento é importante avaliar se pode haver impacto ao negócio, visto que, durante um determinado intervalo de tempo até que se conclua a integração de dados, os servidores estarão aplicando tabela de preços diferentes para determinados materiais;
4. **Gerenciamento do processamento:** o baixo ou alto desempenho no atendimento das requisições pode interferir diretamente no tempo do processo de integração, o que pode ser comprovado ao observar a diferença entre a execução dos dois primeiros cenários e o terceiro.

Caso os riscos levantados sejam aceitáveis pode se optar por este modelo.

6 INTEGRAÇÃO COM BPEL (EDI)

Este capítulo apresenta a terceira proposta de integração, na qual a integração das informações ocorrerá por intermédio da ferramenta BPEL, que terá o papel de coletar as informações no servidor emissor e encaminhar estas através de arquivos digitais. Estes arquivos serão encaminhados da matriz para filial com as informações a serem atualizadas. A empresa destino, após realizar o processamento, encaminhará um novo arquivo com o resultado da atualização.

6.1 Detalhamento da proposta de integração

A identificação dos dados a serem integrados será realizada da mesma forma como descrito no item 5.1. Serão criadas duas tabelas para armazenar as informações a serem coletadas, uma *trigger* responsável por inserir as informações na tabela de controle e uma *function* para atualização dos dados no servidor de destino.

O processo de integração no modelo EDI difere dos demais, pois ocorre em três etapas que são: (i) a coleta e envio dos dados a filial; (ii) o processamento no servidor da filial, contemplando neste processamento a atualização dos dados, geração e envio do arquivo de retorno com o resultado do processamento ao servidor da matriz; (iii) por fim, o processamento do arquivo de retorno no servidor da matriz. Ao contrário dos modelos anteriores, neste a responsabilidade pelo processamento dos dados é compartilhada entre as duas empresas, que hora devem gerar arquivos e hora processar e vice-versa.

O ciclo de integração propriamente dito inicia com a coleta dos dados no servidor de origem. Este procedimento deverá ser executado periodicamente. A cada execução, todos os dados sinalizados como alterados devem ser selecionados e inseridos em um arquivo texto. A disposição das informações no arquivo deverá ser realizada de acordo com layout acordado entre as empresas. Os procedimentos de atualização da fila e recuperação das informações para geração do arquivo são os mesmos mencionados no item 5.1 do processo de integração entre banco de dados com BPEL. O arquivo será gerado somente se haver dados a serem integrados. A figura 6.1 apresenta este fluxo.

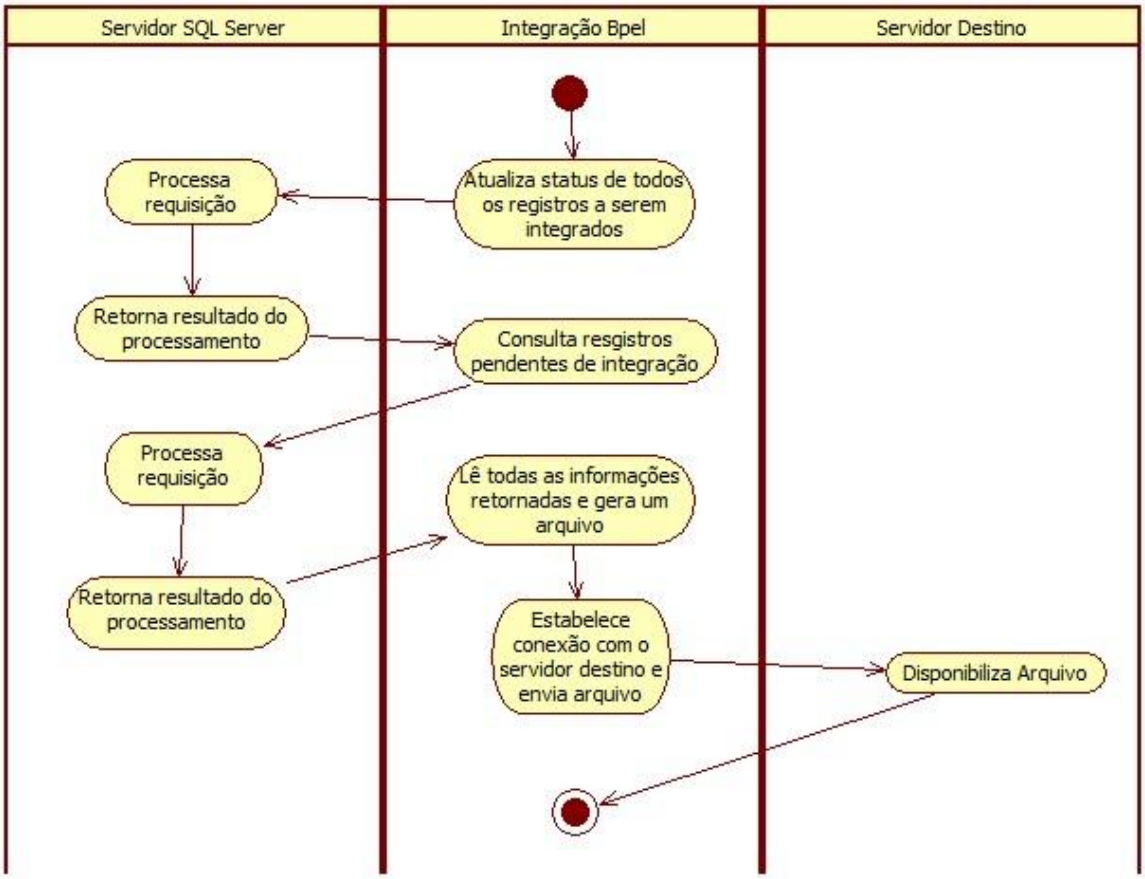

Figura 6.1 – Fluxo para coleta de dados no servidor de origem (BPEL EDI)
Fonte: Próprio autor

No servidor destino um sistema (a linguagem de desenvolvimento e/ou tecnologia deste sistema não é relevante), valida periodicamente a existência de arquivos no diretório da integração. Ao identificar um arquivo no diretório, inicia o processamento e partir deste momento, as informações capturadas serão atualizadas uma a uma no servidor destino. A atualização das informações será realizada por intermédio de uma *function*, denominada f_cadastra_material, detalhada no item 4.2. O resultado do processamento é obtido ao avaliar o retorno da *function*, e esta informação será utilizada para gerar o arquivo de retorno. O arquivo de retorno depois de gerado será transmitido ao servidor de origem. O arquivo de origem recebido nesta integração após ter sido processado é movido para o diretório de processados, finalizando esta etapa do processo. Este fluxo é representado na figura 6.2.

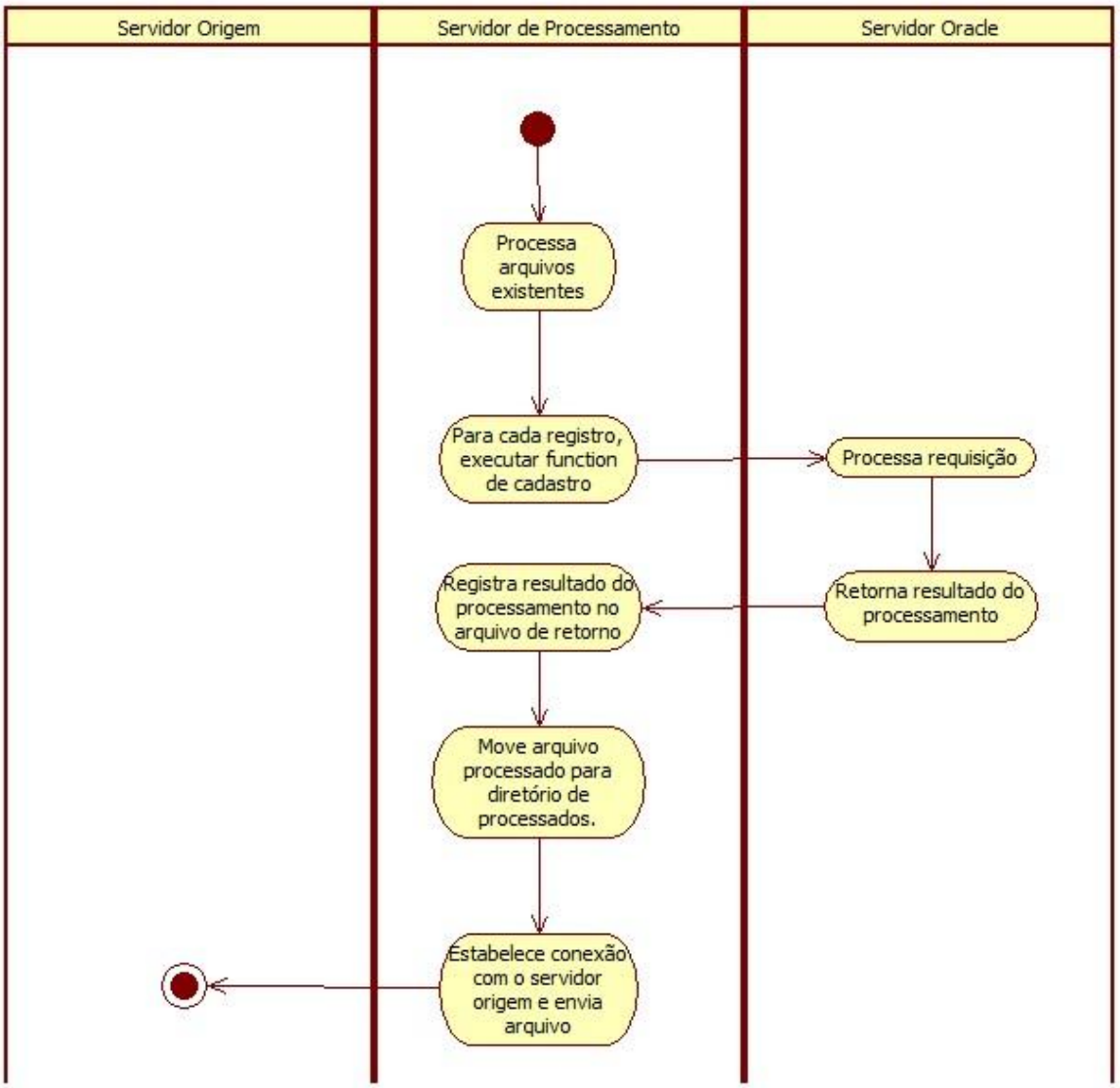

Figura 6.2 – Fluxo para processamento de dados no servidor destino
Fonte: Próprio autor

A última etapa da integração é realizada no servidor de origem, onde uma rotina desenvolvida em BPEL avalia o diretório de integração em busca do arquivo de retorno. Ao identificar um arquivo no diretório de integração inicia o seu processamento. As informações são submetidas uma a uma ao servidor de origem, realizando desta forma a atualização do status do processamento. Ao finalizar o processamento do arquivo o mesmo será movido para o diretório de processados, finalizando o ciclo de integração. O fluxo detalhado é apresentado na figura 6.3.

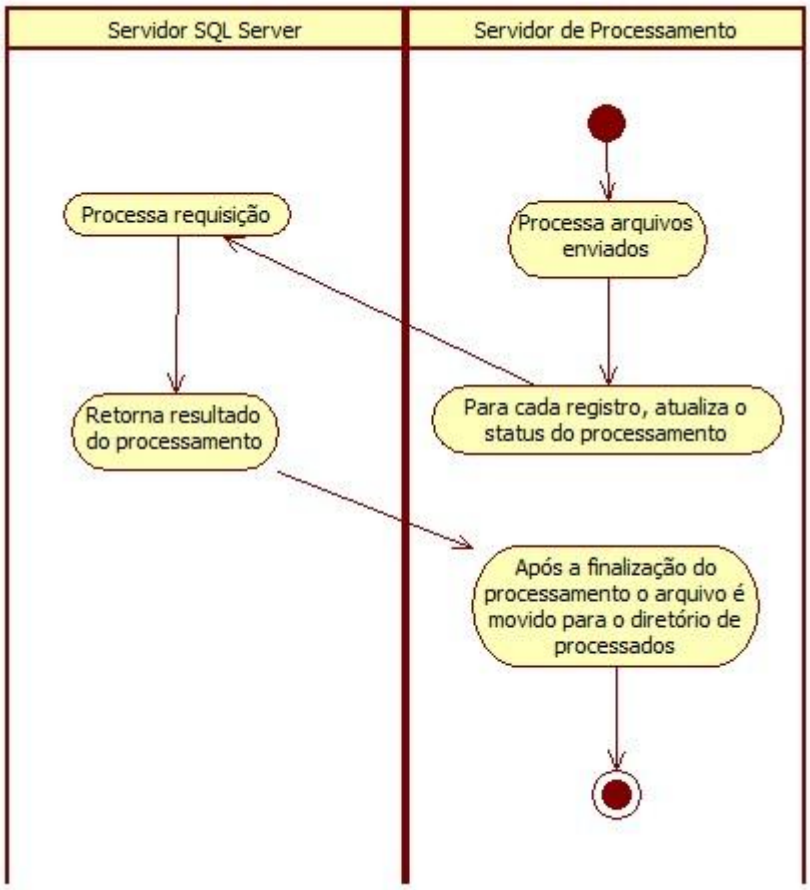

Figura 6.3 – Fluxo para atualização do status de integração no servidor de origem
Fonte: Próprio autor

6.2 Desenvolvimento

O desenvolvimento desta integração começa pela definição dos layouts dos arquivos de remessa e retorno. Ambos os arquivos partilham praticamente da mesma estrutura, sendo diferenciados somente pelo número de campos que os compõem. A seguir as principais características em comum que os arquivos possuem:

1. **Nome do arquivo:** será composto por 16 caracteres, onde as três primeiras posições representam a origem do arquivo, as próximas três a ação, as quatro seguintes às informações trafegadas no arquivo e por fim um número sequencial, que deve ser incrementado a cada envio de arquivo;
2. **Leitura:** define a forma como as informações serão separadas umas das outras dentro do arquivo. O tipo posicional define que o tamanho do campo no arquivo é fixo, iniciando em determinada coluna e finalizando na mesma ou em outra. A segunda opção é o uso de delimitadores, onde as informações são separadas

através de um caractere específico, permitindo que o tamanho dos campos possa variar de acordo com o tamanho das informações. Para o desenvolvimento dos testes foi utilizada a leitura posicional.

3. **Cabeçalho:** não será utilizado.
4. **Rodapé:** será composto por seis caracteres numéricos que representam o número de registros no arquivo.
5. **Diretórios:** a estrutura de diretórios em cada um dos servidores será composta de duas pastas, uma delas denominada de Pendente, utilizada para receber os arquivos trafegados entre as empresas e uma segunda pasta denominada Processado, que armazenara todos os arquivos já processados.

Levando em consideração as definições acima, o arquivo de envio gerado pela matriz receberá o nome MATENVPRODNNNNNN.txt, onde MAT representa matriz, ENV representa envio, PROD faz referência as informações contidas no arquivo e NNNNNN a sequência numérica do arquivo. As informações que compõem o arquivo são três: código do produto, descrição do produto e valor de venda. Estas informações serão obtidas através de consultas realizadas no banco de dados. Detalhes do arquivo de envio podem ser observados na tabela 6.1.

NOME DO ARQUIVO	MATENVPRODNNNNNN.txt
LEITURA	POSICIONAL
RODAPÉ	NNNNNN

CAMPOS	TIPO DADO	TAMANHO	INÍCIO	FIM	ALINHAMENTO	REGRA DE NEGÓCIO
CÓDIGO DO PRODUTO	NUMERIC	15	1	15	DIREITA	-
DESCRIÇÃO DO PRODUTO	VARCHAR	100	16	115	ESQUERDA	-
VALOR DE VENDA	NUMERIC	14	116	129	DIREITA	-

Tabela 6.1 – Definição do layout do arquivo de remessa
Fonte: Próprio autor

O segundo arquivo que será gerado pela filial - resultado do processamento – deverá receber o seguinte nome FILRETPRODNNNNNN.txt, onde FIL representa filial, RET representa retorno, PROD faz referencia as informações contidas no arquivo e NNNNNN a sequência numérica do arquivo. As informações que devem compor o arquivo são duas: código do produto e o status do processamento. Na tabela 6.2 é possível observar os detalhes do arquivo.

NOME DO ARQUIVO	FILRETPRODNNNNNN.txt					
LEITURA	POSICIONAL					
RODAPÉ	NNNNNN					
CAMPOS	TIPO DADO	TAMANHO	INÍCIO	FIM	ALINHAMENTO	REGRA DE NEGÓCIO
CÓDIGO DO PRODUTO	VARCHAR	20	1	15	DIREITA	-
STATUS DO PROCESSAMENTO	INTEGER	4	21	24	ESQUERDA	1 - SUCESSO 2 - ERRO

Tabela 6.2 – Definição do layout do arquivo de retorno
Fonte: Próprio autor

Definidos os layouts dos arquivos, é possível iniciar o desenvolvimento do processo que receberá o nome de IntegracaoGeraArquivo. Esta integração terá por finalidade atualizar o status da fila, coletar as informações a serem integradas e se encerra com a geração do arquivo de envio. Como as duas primeiras atividades de atualização e consulta são idênticas às executadas na integração anterior (já detalhadas na seção 5.2), o processo passara a ser detalhado a partir do retorno dos dados. Na figura 6.4 é possível observar as duas primeiras atividades mencionadas acima.

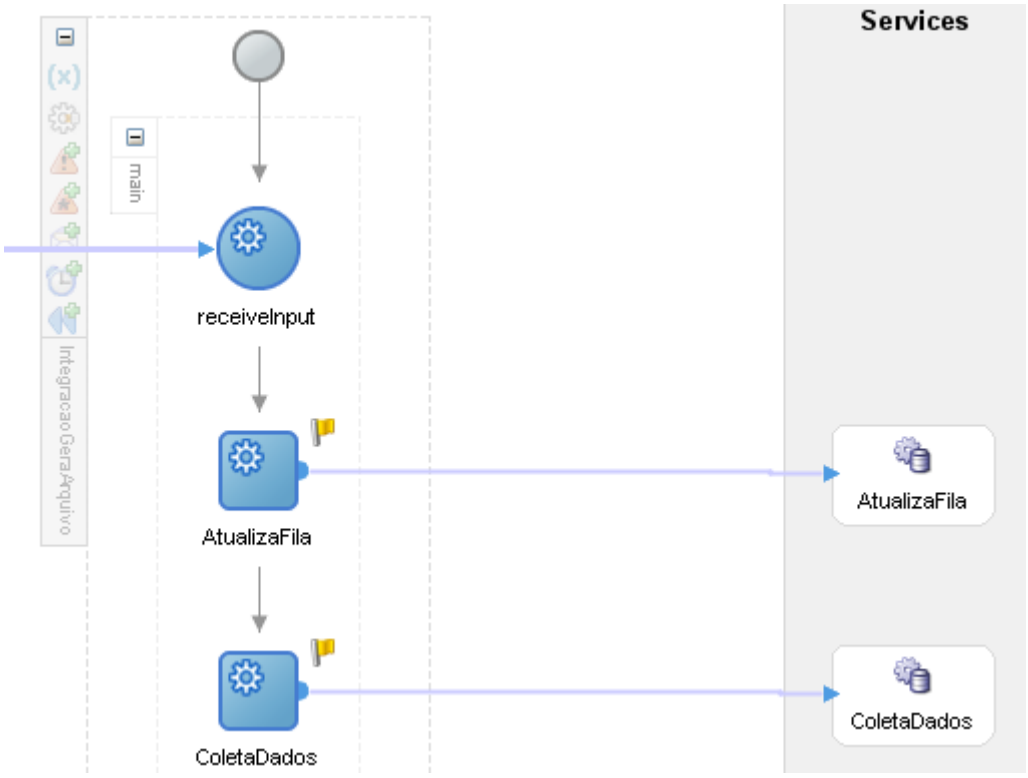

Figura 6.4 – Fluxo de processo no JDeveloper (BPEL EDI)
Fonte: Próprio autor

Após o retorno dos dados, que ficarão armazenados na variável de saída do componente *invoke* denominado no fluxo como ColetaDados, o próximo passo é gerar o arquivo. Para isso será utilizado um componente chamado de *FileAdapter*. Para configurar o componente algumas informações serão solicitadas como tipo de operação (leitura ou escrita), que neste caso será escrita, o diretório onde o arquivo será gerado, o nome do arquivo e por fim a definição do layout. Para definição do layout o componente disponibiliza um recurso que permite a partir de um arquivo já gerado no modelo proposto, carregar as configurações e montar a partir deste as estruturas do arquivo. A opção mencionada pode ser observada na figura 6.5.

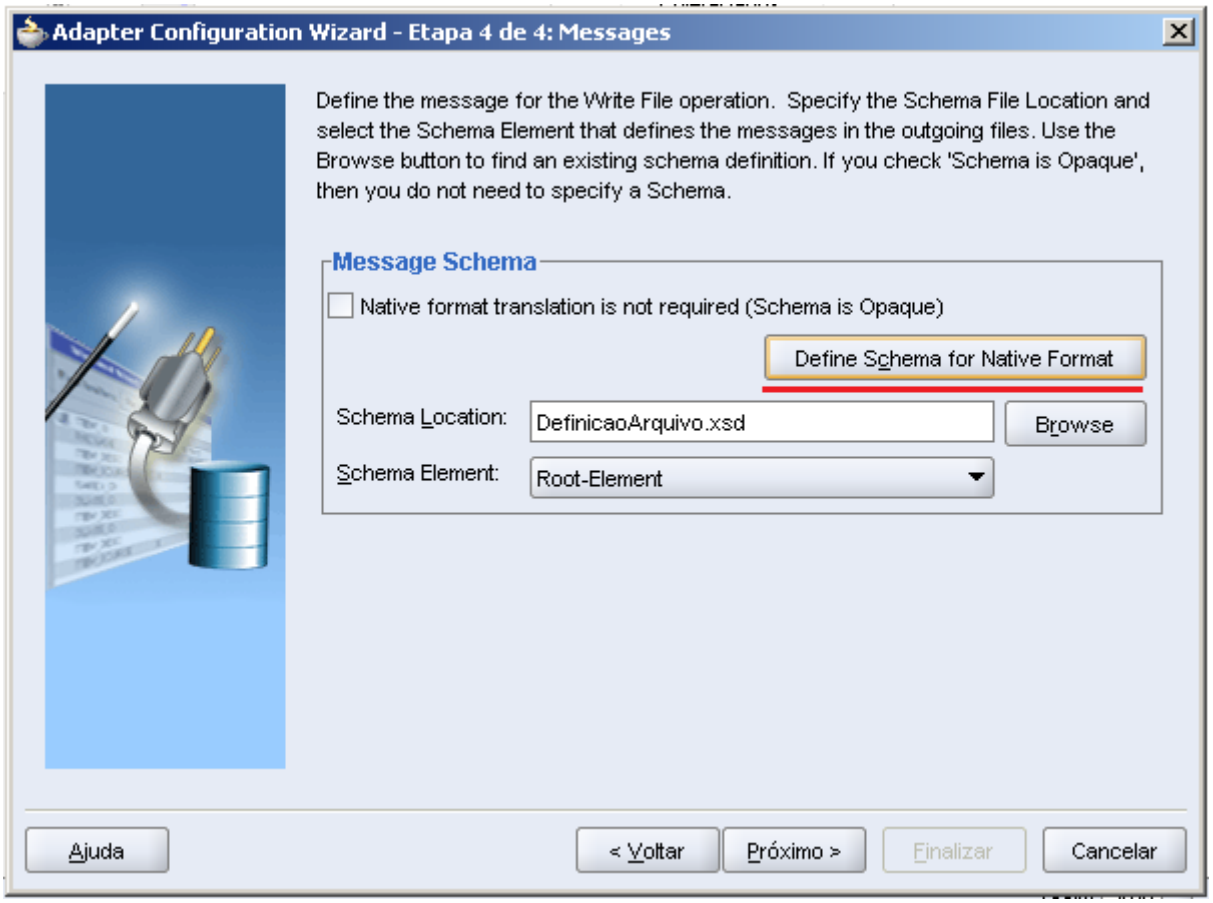

Figura 6.5 – Configuração do componente *FileAdapter*
Fonte: Próprio autor

Definido o layout, o próximo passo é carregar os dados da variável de saída do componente *invoke* para o *FileAdapter*. Para isso, será necessário adicionar dois novos componentes. O primeiro como já utilizado em outras ocasiões é o *invoke*, que receberá as informações e encaminhará as mesmas ao *FileAdapter*. O segundo componente a ser adicionado chama-se *transform*. O *transform* é um componente que suporta o processamento

de várias linhas, como também permite mapear os campos retornados na consulta diretamente para o componente *invoke* associado ao *FileAdapter*. Ou seja, a cada linha retornada na consulta, enviará as informações por intermédio do *invoke* para o *FileAdapter* que por sua vez criará o arquivo. Na figura 6.6 é possível observar os campos retornados na consulta e o mapeamento do mesmo para o *FileAdapter*. O componente t*ransform* é última ação a ser adicionada ao fluxo.

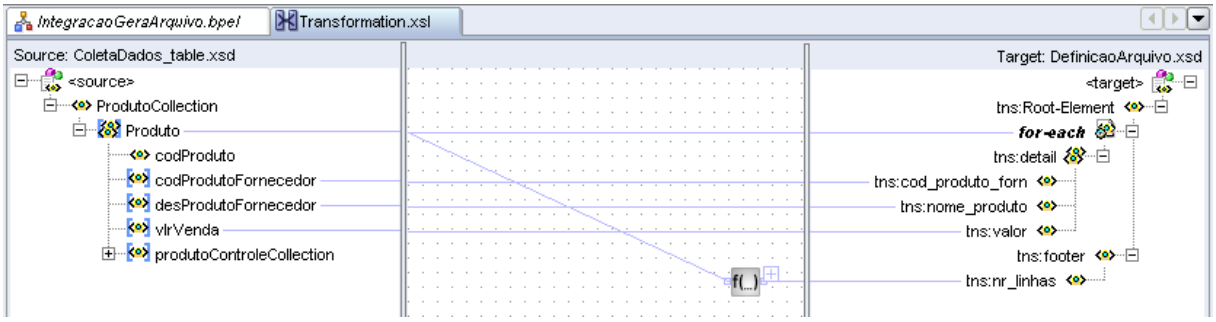

Figura 6.6 – Configuração do componente *Transform*
Fonte: Próprio autor

Com isto, a parte de coleta e envio de dados à filial está finalizada. Resumindo, nesta etapa os dados a serem integrados foram coletados e os registros armazenados no arquivo a ser encaminhado à filial. O fluxo completo do processo de envio de arquivo pode ser observado na figura 6.7.

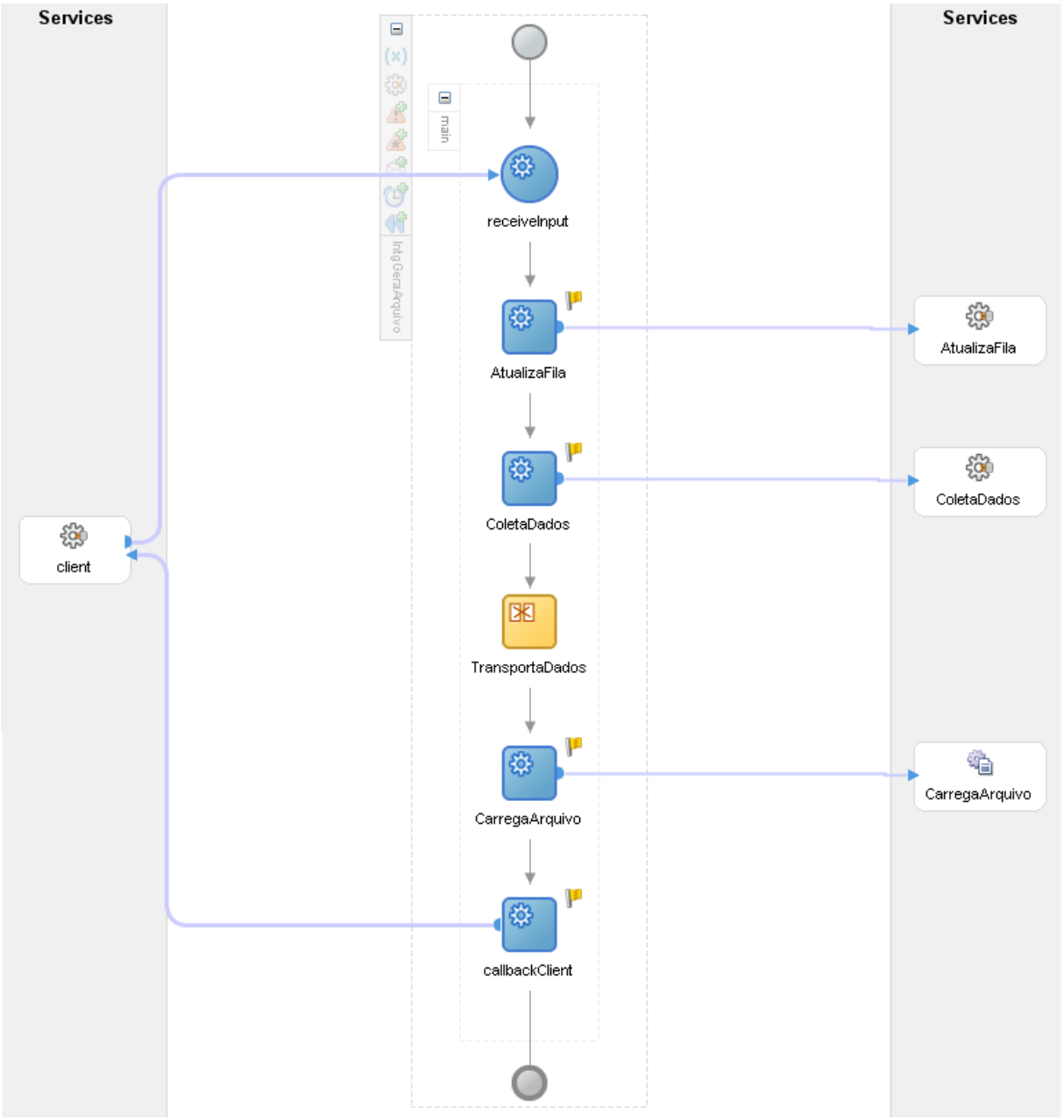

Figura 6.7 – Fluxo de processo no JDeveloper (BPEL EDI)
Fonte: Próprio autor

Na etapa dois o processamento é realizado na filial, que aguardará o arquivo com as informações a serem integradas encaminhado pela matriz. A filial através de uma aplicação própria desenvolvida em Delphi, mas que poderia ter sido desenvolvida em Power Builder, Java ou até mesmo em BPEL, ao identificar o arquivo disponível no diretório de pendentes irá iniciar o processamento. Durante o processamento do arquivo irá incluir ou atualizar as informações encaminhadas em sua base de dados, após ter processado o último registro irá gerar um novo arquivo com o status do processamento, indicando para cada registro se o

mesmo foi atualizado em sua base de dados ou não. O arquivo após concluído será encaminhado a matriz.

A etapa três definida como processamento do arquivo de retorno da filial, consiste em processar o arquivo de retorno da filial, analisar o status e atualizar esta informação para cada um dos registros encaminhados. Durante este processamento, será avaliado o resultado informado no arquivo de retorno para cada um dos registros e atualizado o status da fila de integração.

Para iniciar o desenvolvimento do terceiro processo um novo projeto será aberto e será denominado de ProcessaArquivo. No momento da criação, deverá ser selecionada a opção *Empty BPEL Process* no campo *template*. A opção de *template* selecionada sinaliza que o processo não recebe informações por parâmetro e não é disparado por estímulos externos.

O primeiro componente a ser adicionado ao fluxo é o *FileAdapter*, que terá o papel de verificar periodicamente a existência do arquivo num determinado diretório e, caso este seja encontrado, iniciar seu processamento. As informações solicitadas durante a configuração do componente são as seguintes: tipo de operação (escrita ou leitura), para este processo será leitura, a localização do arquivo a ser processado, o local para onde será transferido após processamento, o nome do arquivo a ser processado, a frequência que o diretório deverá ser analisado em busca de arquivos e por fim, o mapeamento das informações do arquivo para dentro do processo.

Configurado o *FileAdapter*, é necessário adicionar o próximo componente que terá o papel de receber as informações. O nome do componente é *receive* e após adicionado ao fluxo deverá ser associado ao componente *FileAdapter*. A figura 6.8 mostra os dois componentes adicionados ao fluxo.

Figura 6.8 – Fluxo de processo no JDeveloper (BPEL EDI)
Fonte: Próprio autor

As informações capturadas pelo *FileAdapter* serão transferidas para uma variável de saída do componente *receice*, dando inicio a próxima etapa do processo que consiste em ler as informações e atualizar o status da fila de integração. Como no retorno do arquivo podem ser encaminhadas 1 ou n linhas, a variável do componente *receive* também terá esta característica. Logo, se faz necessário criar uma estrutura que permita processar todas as linhas da variável. A solução aplicada neste cenário será a mesma já aplicada no item 4.2 onde foi utilizado o componente *while*, permitindo o processamento linha a linha de cada um dos registros e a atualização do resultado na tabela da integração. Com a aplicação destes componentes o processo está finalizado. A figura 6.9 apresenta o fluxo completo para atualização do status da fila a partir do processamento do arquivo de retorno.

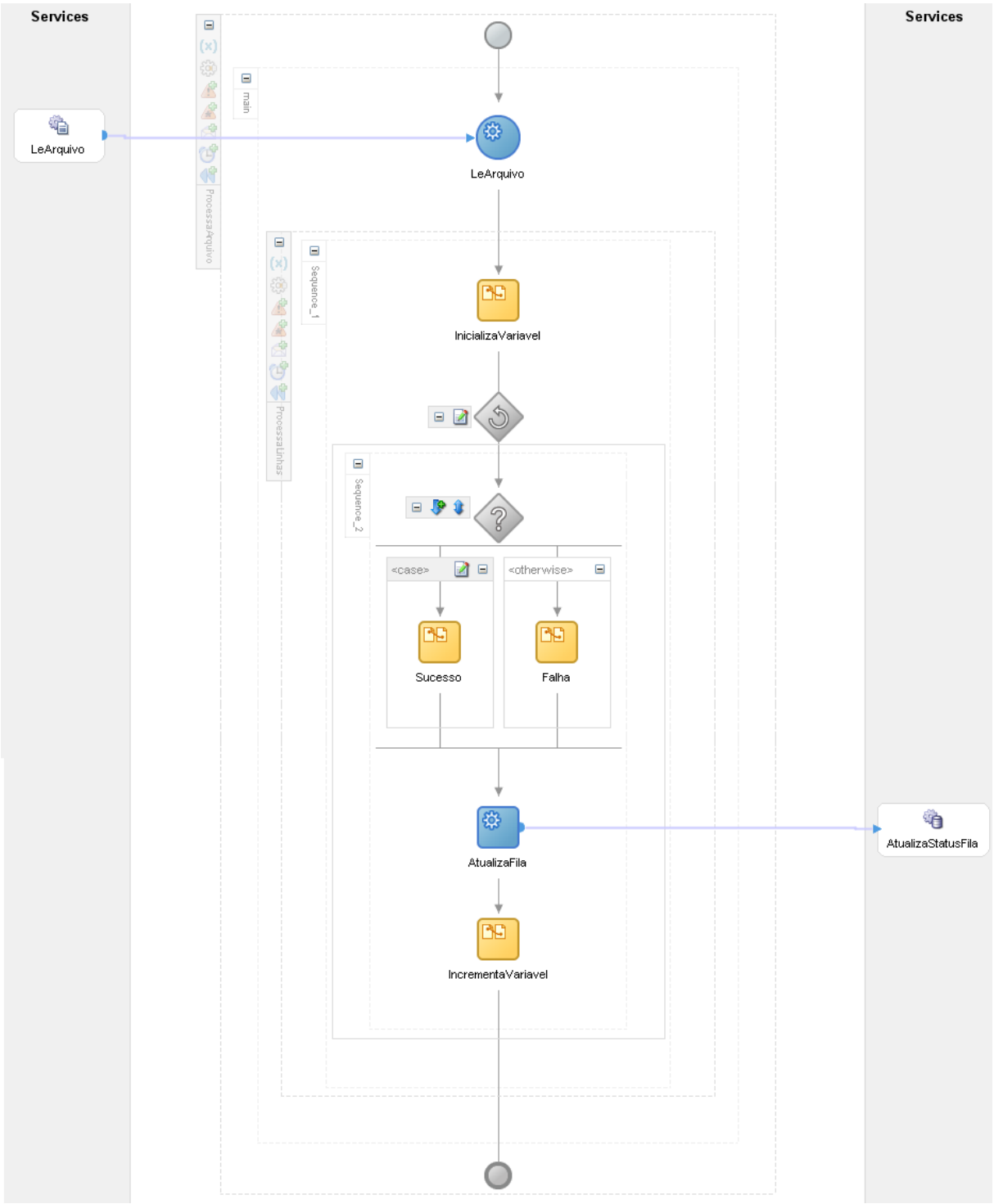

Figura 6.9 – Fluxo completo do processo no JDeveloper (BPEL EDI)
Fonte: Próprio autor

6.3 Resultados

Com o processo disponível no servidor SOA BPEL, já é possível executar o cenário de integração. Para avaliar o fluxo, os 40 mil itens existentes no servidor SQL Server foram alterados, gerando por sua vez uma demanda de inclusão/atualização de dados para o servidor

Oracle. Todas as informações alteradas serão agendadas na tabela de integração e ficarão aguardando a execução do processo de integração para que possam ser encaminhadas ao servidor de destino.

Na matriz o processo BPEL de envio de arquivos será agendado para execução a cada trinta minutos. Já na filial o processamento do arquivo encaminhado pela matriz será iniciado tão logo tenha recebido o mesmo e ao finalizar seu processamento encaminhará a matriz o resultado. O processamento do arquivo de retorno encaminhado pela filial a matriz estará agendado para execução a cada dez minutos.

Para que seja possível comparar os modelos de integração ao final do trabalho, será considerado para este cenário que a filial sempre realizará o processamento do arquivo encaminhado pela matriz no intervalo de dez minutos. Desta forma, além do tempo gasto na geração do arquivo e processamento do retorno, será acrescido o tempo de processamento na filial.

Para a avaliação do cenário foram realizadas alterações nos registros de cadastro de materiais em três momentos distintos e os resultados obtidos foram os seguintes:

1ª. **Execução:** as 40 mil linhas foram integradas com sucesso em 55 minutos e 7 segundos, onde destes 6 minutos e 36 segundos correspondem ao tempo gasto para geração do arquivo para filial, 10 minutos, o intervalo entre a execução dos processos de envio e leitura do retorno e 38 minutos e 31 segundos o tempo de processamento do arquivo de retorno da filial para atualização do status da fila de processamento. Levando em média 0,082675 segundos por registro.

2ª. **Execução**: As 40 mil linhas foram integradas com sucesso em 57 minutos e 57 segundos, onde destes 6 minutos e 15 segundos correspondem ao tempo gasto para geração do arquivo para filial, 10 minutos, o intervalo entre a execução dos processos de envio e leitura do retorno e 41 minutos e 42 segundos o tempo de processamento do arquivo de retorno da filial para atualização do status da fila de processamento. Levando em média 0,086925 segundos por registro.

3ª. **Execução:** as 40 mil linhas foram integradas com sucesso em 55 minutos e 53 segundos, onde destes 5 minutos e 54 segundos correspondem ao tempo gasto para geração do arquivo para filial, 10 minutos, o intervalo entre a execução dos processos de envio e leitura do retorno e 45 minutos e 53 segundos o tempo de processamento

do arquivo de retorno da filial para atualização do status da fila de processamento. Levando em média 0,083825 segundos por registro.

6.4 Análise do resultado

Todas as integrações foram realizadas com sucesso com variações mínimas entre as execuções, o modelo de integração atendeu a necessidade. Entretanto, cabe ressaltar quatro pontos de atenção que devem ser considerados ao optar por este modelo, são eles:

1. **Falha na composição do arquivo:** tanto do ponto de vista da filial como da matriz, arquivos gerados fora do layout proposto poderão impossibilitar a leitura do arquivo em sua totalidade. Estes problemas podem ser causados pela própria aplicação ou até mesmo pelos dados que possam conter caracteres de quebra de linha. Falhas deste tipo necessitam por vezes ações manuais, seja no próprio arquivo ou na identificação dos registros não processados para que possam ser encaminhados novamente;

2. **Tempo de integração:** para que este modelo de integração fosse implementado uma nova camada de software foi adicionada, o SOA BPEL, com isso é perceptível o aumento no tempo de integração, seja na geração do arquivo de envio, como também no processamento do arquivo de retorno;

3. **Atualização *off-line*:** como os dados são alterados e integrados somente num segundo momento é importante avaliar se pode haver impacto ao negócio, visto que, durante um determinado intervalo de tempo até que se conclua a integração de dados, os servidores estarão aplicando tabela de preços diferentes para determinados materiais;

4. **Falhas na aplicação do destinatário:** neste modelo de integração existe uma forte dependência entre os sistemas, onde, em caso de falhas em qualquer lado da integração paralisam todo o processo.

Caso os riscos levantados sejam aceitáveis pode se optar por este modelo.

7 INTEGRAÇÃO COM BPEL (SOA)

Este capítulo apresenta a quarta proposta de integração, na qual a integração entre as duas empresas ocorrerá por intermédio de um *Web Service*, ou seja, sempre que forem identificadas alterações nas informações de produto na matriz, a aplicação em BPEL consumirá o *Web Service* de atualização cadastral disponibilizado pela filial.

7.1 Detalhamento da proposta de integração

A identificação dos dados a serem integrados será realizada da mesma forma como descrito no item 5.1. Ou seja, serão criadas duas tabelas para armazenar as informações a serem integradas, uma *trigger* responsável por inserir as informações na tabela de controle e uma *function* para atualização dos dados no servidor de destino.

O processo propriamente dito será executado periodicamente e a cada execução da aplicação BPEL, processará todas as informações alteradas pendentes de integração. Para cada informação obtida, a aplicação BPEL irá consumir um *Web Service* disponibilizado no servidor da filial e ficará aguardando até que o serviço finalize seu processamento para dar continuidade às próximas ações do fluxo. O *Web Service* receberá um a um os dados a serem alterados retornando ao final da execução o resultado do processamento.

O *Web Service* terá três parâmetros: código do produto, descrição do produto e valor de venda. O serviço sempre que disparado, estabelecerá conexão com o banco de dados da filial e atualizará as informações em sua base de dados. O procedimento de atualização de dados será realizado por uma *function* f_cadastra_material, que recebe como parâmetro as mesmas informações encaminhadas ao *Web Service*. O serviço possui um único parâmetro de retorno. Em caso sucesso responderá “OK”, caso contrário o motivo pelo qual não pode realizar a atualização.

A aplicação BPEL ao receber o resultado da execução dará continuidade a atualização do status do processamento. Neste momento, identificará a correspondência entre o dado retornado pelo *Web Service* e a tabela de status de processamento, para então realizar a atualização dos dados. Finalizada a atualização dos dados, segue com o processamento do próximo registro, finalizando somente após o processamento do último registro. Na imagem 7.1 é possível observar todo o fluxo de integração detalhado acima.

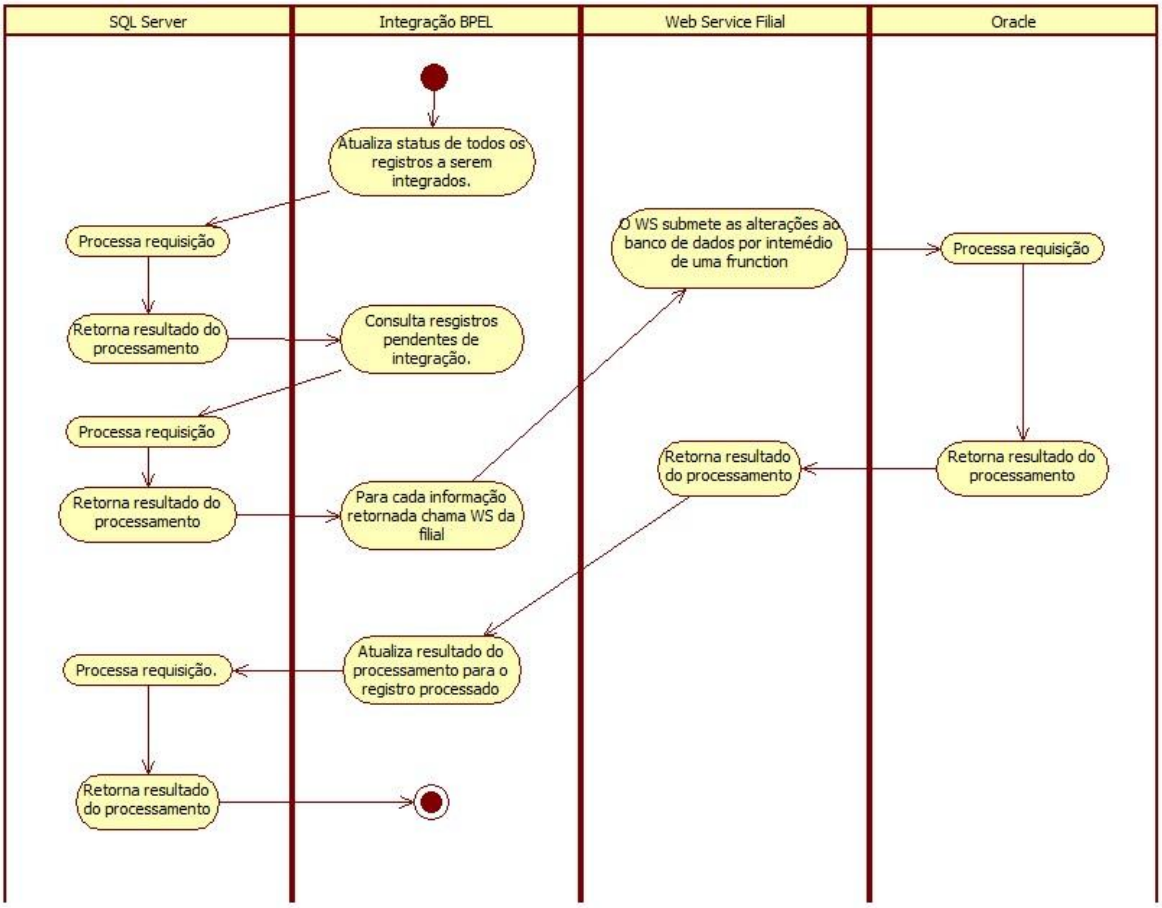

Figura 7.1 – Fluxo de integração para atualização de dados do produto (BPEL SOA)
Fonte: Próprio autor

7.2 Desenvolvimento

O desenvolvimento do cenário inicia com a criação de um novo projeto que receberá o nome de AtualizaMaterialViaWebService que deverá ser configurado como *Asynchronous BPEL Process* na opção *template*. Esta integração terá por finalidade atualizar o status da fila, coletar as informações a serem integradas, submeter os dados ao servidor de destino por intermédio de um *Web Service*, que após realizar o processamento dos dados, responderá com o resultado da atualização. Este resultado será avaliado e o status equivalente será atribuído ao registro corrente. Os dados a serem integrados serão encaminhados ao servidor de destino um a um via *Web Service* e somente após processado o último registro o processo será finalizado. Como as duas primeiras atividades, atualização e coleta de dados são idênticas às executadas na integração 4.2, o processo passará a ser detalhado a partir do retorno dos dados. Na figura 7.2 é possível observar as duas primeiras etapas mencionadas acima.

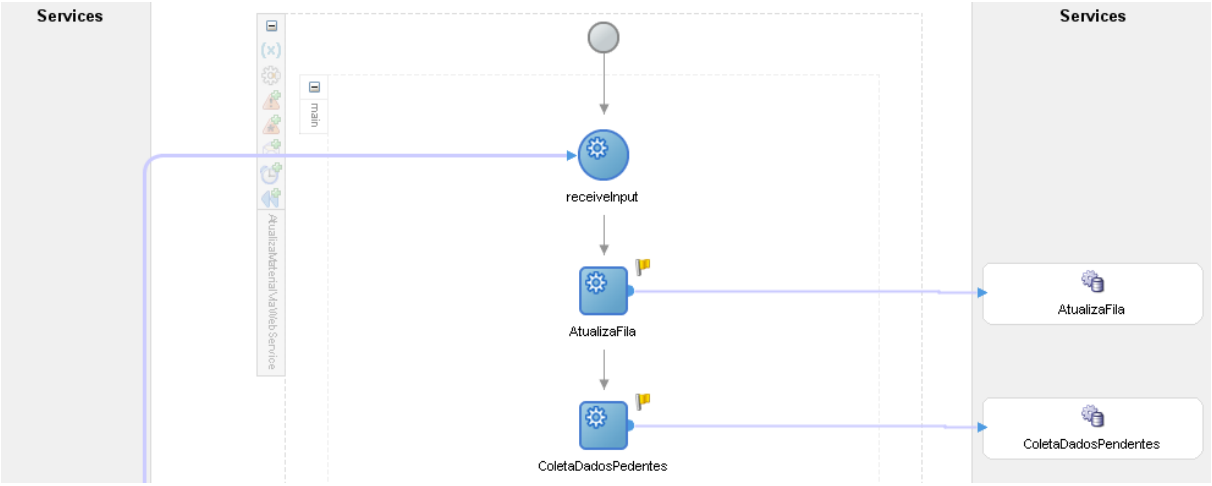

Figura 7.2 – Fluxo de processo no JDeveloper (BPEL SOA)
Fonte: Próprio autor

Realizada a consulta ao banco de dados, as informações a serem integradas serão armazenadas na variável de saída do componente *invoke*, denominado neste fluxo como ColetaDadosPendentes. Como o resultado desta consulta pode ser zero, um ou n registros, será necessário implementar uma instrução de repetição denominada *while*, da mesma forma como já realizado no item 4.2.

A cada ciclo do *while*, as informações necessárias para atualização no servidor destino serão armazenadas no componente *assign* denominado no fluxo como CarregaDados. Para que possam ser encaminhadas ao servidor de destino, serão adicionados ao fluxo dois componentes, um *invoke* e um *partner link* (disponível na classe *Services*). Há um pré-requisito para iniciar a configuração do componente *partner link*, que consumirá um *Web Service*. O serviço deve estar disponível para que seus parâmetros de entrada e saída possam ser mapeados pelo componente. Está validação pode ser realizada por intermédio de um navegador de internet. O procedimento de validação consiste em colocar a URL (*Uniform Resource Locator*) que corresponde ao endereço da localização do serviço na Web (WSDL), no campo endereço e selecionar a opção consulta do navegador. O resultado pode ser observado na figura 7.3, confirmando que o serviço está disponível e é possível iniciar a configuração do componente.

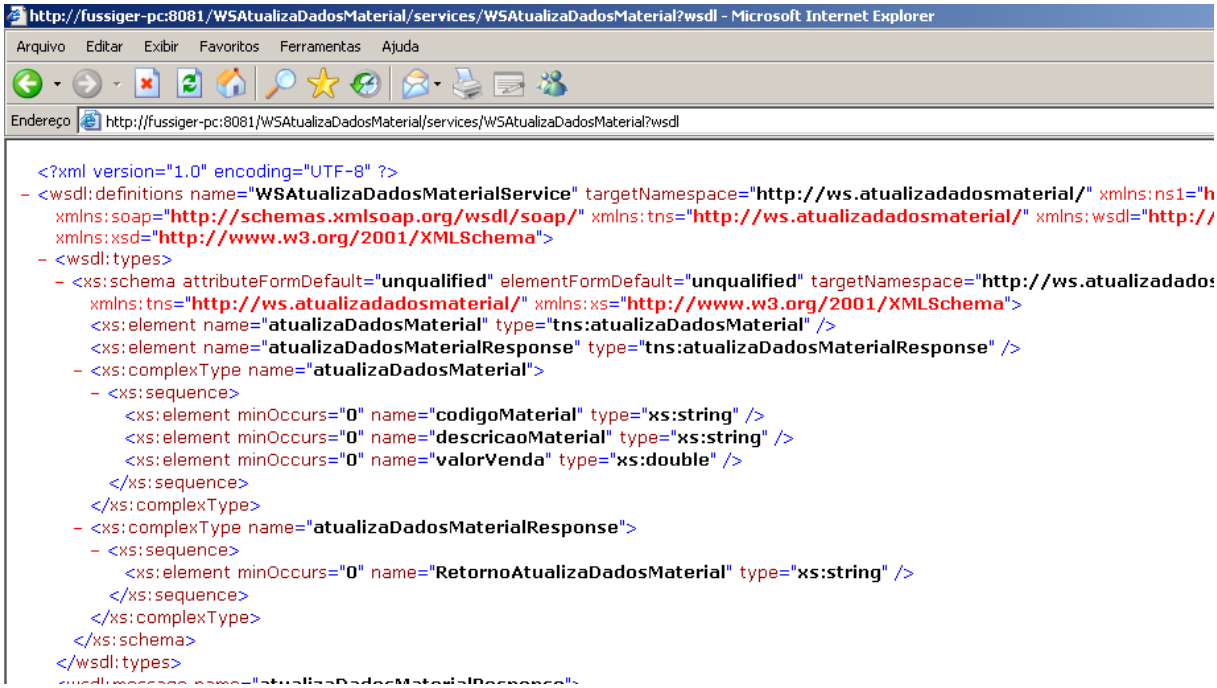

Figura 7.3 – Visão parcial da estrutura do *Web Service* WSAtualizaDadosMaterial
Fonte: Próprio autor

Depois de adicionado o componente *partner link*, será exibida a tela de configuração do mesmo, onde, no campo WSDL File será informada a URL. Depois de informada a URL, o componente irá mapear as informações do *Web Service* automaticamente e após finalizado, os demais campos de configuração da tela ficam disponíveis para seleção. Para concluir a configuração, no combo *Partner Role* a opção WSAtualizaDadosMaterial_role deve ser selecionada, tornando o componente pronto para ser utilizado. Na figura 7.4 é possível observar as opções citadas acima bem como as informações atribuídas durante a configuração.

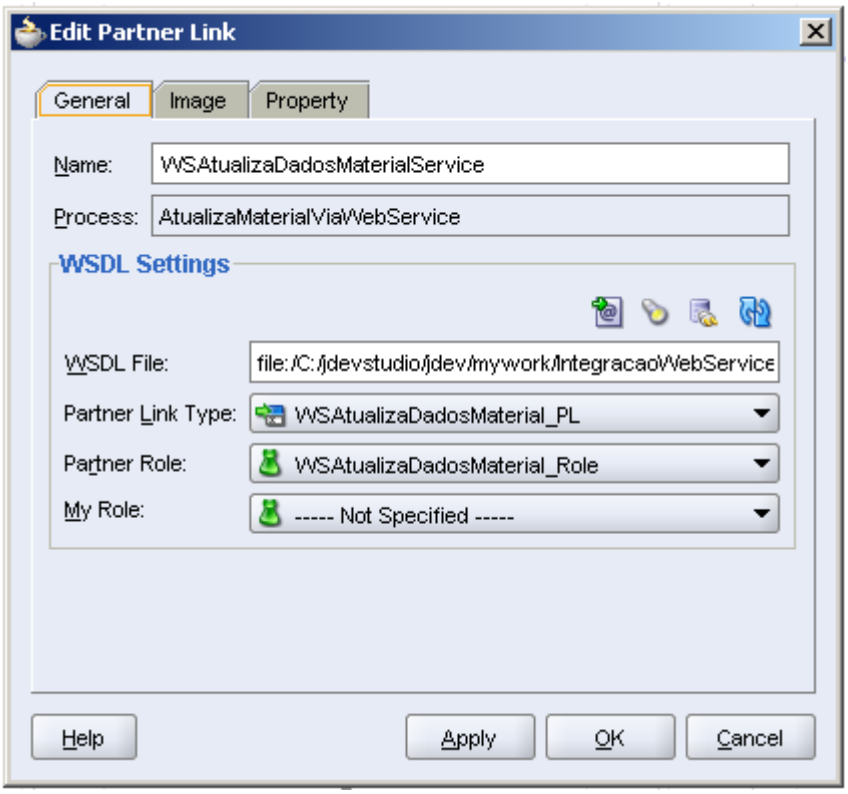

Figura 7.4 – Configuração do componente *Partner Link*
Fonte: Próprio autor

Configurado o componente *partner link*, pode ser adicionado um componente denominado *invoke*. Este terá o papel de encaminhar as informações a serem atualizadas ao *partner link* e disparar sua execução, que por sua vez irá consumir o *Web Service*, e ao final do seu processamento armazenar em sua variável de retorno o resultado retornado pelo *Web Service*.

Para avaliar o resultado do *Web Service* será adicionado um novo componente denominado *switch*, que identificará a correspondência entre a informação de retorno do *Web Service* com os status equivalentes de processamento. Posteriormente, realizará a atualização desta informação na fila de integração encerrando o ciclo de atualização. Havendo mais registros a serem processados, um novo ciclo será iniciado. A execução será encerrada somente após todos os registros terem sido processados será encerrado. O fluxo descrito a cima a partir do retorno do *Web Service* é idêntico ao aplicado no item 4.2, onde maiores detalhes sobre sua configuração podem ser obtidos. Com a aplicação destes componentes o processo está pronto para ser utilizado. Na figura 7.5 é possível observar o fluxo completo deste processo.

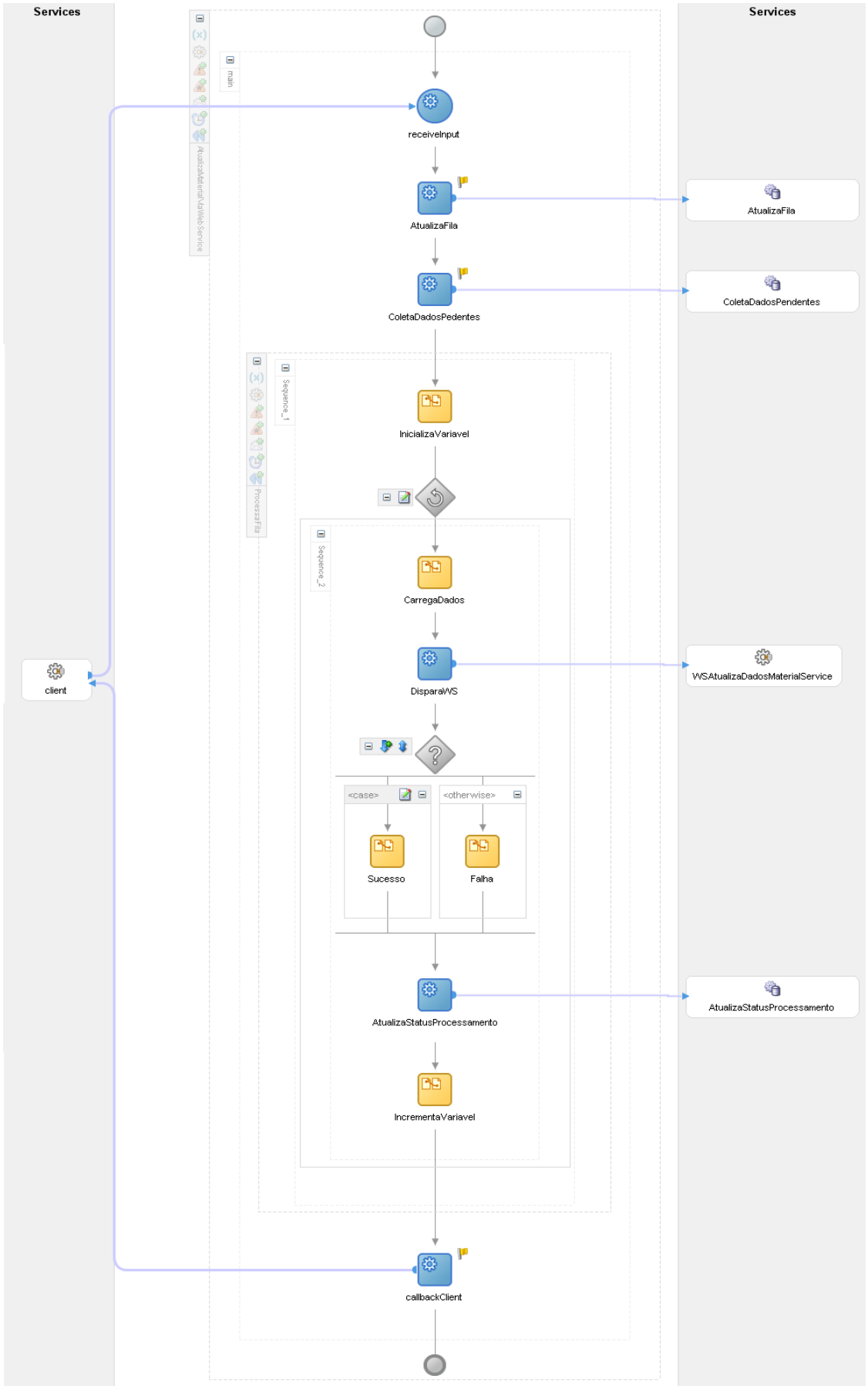

Figura 7.5 – Fluxo completo do processo no JDeveloper (BPEL SOA)
Fonte: Próprio autor

7.3 Resultados

Com o processo disponível no servidor SOA BPEL, já é possível executar o cenário de integração. Para avaliar o fluxo, os 40 mil itens existentes no servidor SQL Server foram alterados, gerando por sua vez uma demanda de inclusão/atualização de dados para o servidor Oracle. Todas as informações alteradas serão agendadas na tabela de integração e ficarão aguardando a execução do processo de integração para que possam ser encaminhadas ao servidor de destino.

Na matriz a rotina de integração será executada a cada dez minutos, fazendo com que todos os registros pendentes de integração sejam encaminhados a filial. Ou seja, na pior das hipóteses o registro após ter sido agendado para integração aguardará até dez minutos para que seja processado. A atualização dos dados na filial será realizada por intermédio de um *Web Service*, disponibilizado pela mesma, não havendo acesso direto ao banco de dados da filial por parte da matriz. A confirmação do processamento das informações pelo servidor da filial será obtida *on-line*, ou seja, submetidas as informações ao *Web Service*, este processará os dados e ao término devolverá o resultado. Desta forma, além do tempo gasto no processo de integração será acrescido o intervalo de tempo entre cada execução da integração.

As alterações realizadas para avaliar o cenário, foram realizadas em três momentos distintos e os resultados obtidos foram os seguintes:

1ª. **Execução:** as 40 mil linhas foram integradas com sucesso em 2 horas, 10 minutos e 33 segundos, onde destes 10 minutos correspondem ao intervalo entre cada execução da integração e 2 horas e 33 segundos o tempo de processamento da integração. Levando em média 0,195825 segundos por registro.

2ª. **Execução:** as 40 mil linhas foram integradas com sucesso em 2 horas, 20 minutos e 23 segundos, onde destes 10 minutos correspondem ao intervalo entre cada execução da integração e 2 horas, 10 minutos e 23 segundos o tempo de processamento da integração. Levando em média 0,210575 segundos por registro.

3ª. **Execução:** as 40 mil linhas foram integradas com sucesso em 2 horas, 17 minutos e 29 segundos, onde destes 10 minutos correspondem ao intervalo entre cada execução da integração e 2 horas, 7 minutos e 29 segundos o tempo de processamento da integração. Levando em média 0,206225 segundos por registro.

7.4 Análise do resultado

Todas as integrações foram realizadas com sucesso com pequenas variações de tempo entre as execuções, o modelo de integração atendeu a necessidade, cabe apenas ressaltar três pontos de atenção que devem ser considerados ao optar por este modelo, são eles:

1. **Tempo de integração**: para que este modelo de integração fosse implementado uma nova camada de software foi adicionada entre os bancos de dados, o SOA BPEL. Com isso é perceptível o aumento no tempo de integração, ou seja, entre obter o dado a ser integrado, enviar ao servidor destino e atualizar o status deste na origem, alguns centésimos de segundos a mais foram necessários;

2. **Atualização *off-line*:** como os dados são alterados e integrados somente num segundo momento é importante avaliar se pode haver impacto ao negócio, visto que, durante um determinado intervalo de tempo até que se conclua a integração de dados, os servidores estarão aplicando tabela de preços diferentes para determinados materiais.

3. **Falhas na aplicação do destinatário:** neste modelo de integração existe uma forte dependência entre os sistemas, onde, em caso de falhas em qualquer lado da integração paralisam todo o processo.

Caso os riscos levantados sejam aceitáveis pode ser optar por este modelo.

7.5 Análise comparativa

Durante a execução dos testes, o único critério de avaliação medido foi o tempo total de execução de cada integração. Contudo, durante o desenvolvimento das integrações ou até mesmo durante sua execução, alguns riscos ou pontos de atenção foram identificados. Estes pontos podem ser determinantes para escolha da tecnologia a ser utilizada. Desta forma, além do tempo de execução de cada uma das integrações, serão considerados como critérios de avaliação os pontos de atenção ou riscos envolvidos em todo o contexto de integração. Os itens a seguir detalham estes pontos de atenção e/ou riscos:

1. **Conexão de rede dedicada:** o cenário proposto neste trabalho previu a utilização de uma VPN como canal de comunicação, que atendeu completamente as necessidades da integração. Assim, outros cenários de integração que utilizem este recurso poderiam aproveitar seus benefícios. No entanto antes de optar por este tipo

de recurso é importante verificar se a configuração do tamanho e a velocidade de tráfego de informações é o suficiente para manter a integração funcionando em sua totalidade. Cabe lembrar também que, caso não houvesse um canal de comunicação dedicado entre os dois pontos a serem integrados, ou ainda, não fosse possível estabelecê-lo seja pelo custo ou por outro motivo qualquer, isto já seria o suficiente para inviabilizar qualquer tipo de integração que dependesse deste pré-requisito;

2. **Atualização de dados o*n-line*:** os dados alterados são integrados no mesmo momento, ou seja, a alteração realizada no emissor é enviada para o destinatário e só será confirmada para o emissor caso tenha sido atualizada com sucesso no destinatário. Este tipo de integração trás consigo uma alta dependência de conectividade, tanto de rede como de banco de dados. Por outro lado, garante que não haverá divergência nos dados;

3. **Controle de recuperação de falhas:** define que em casos de falha, seja por perda de conectividade ou até mesmo exceções de banco de dados, o processo poderá ser reiniciado – quando necessário - sem perda de informações e sem a necessidade de ações manuais para ajuste de dados;

4. **Integração compartilhada:** define que o fluxo de integração de dados não é realizado por uma única aplicação, o que aumenta a complexidade da integração e torna fundamental o uso de uma ferramenta que possa gerenciar tal atividade, principalmente para localizar pontos de falha. Caso em determinado contexto seja necessária a utilização de outras aplicações para realizar todo o fluxo de integração, a ausência de uma ferramenta de gerenciamento poderá trazer grandes riscos a operação, tanto para identificação de falhas de integração como também causadas pela execução da integração em outros sistemas. A ferramenta que possua tal recurso terá vantagens sobre as demais;

5. **Acesso direto a base de dados:** identifica que a aplicação de integração tem acesso a base de dados do destinatário. Este tipo de acesso geralmente é concedido quando se tratam de integrações internas, ou seja, dentro da própria empresa. Para integrações com outras empresas não é comum utilizar deste expediente e geralmente a integração é realizada de forma indireta a partir de uma camada de aplicação;

6. **Desempenho por registro (em segundos):** este critério diz respeito ao tempo médio gasto para que um único registro possa ser integrado, compreendendo a coleta

da informação no servidor de origem, envio e processamento pelo destinatário. Para efeito de avaliação, foram descartados o pior e o melhor tempo apurado durante a execução de cada cenários.

Na tabela 7.1 é possível observar o desempenho das tecnologias para cada um dos critérios detalhados acima.

Critérios	4 -Integração direta entre bancos de dados	5 - Integração entre bancos de dados com BPEL (SQL)	6 - Integração com BPEL (EDI)	7 - Integração com BPEL (SOA)
Conexão de rede dedicada	SIM	SIM	NÃO	NÃO
Atualização de dados on-line	SIM	NÃO	NÃO	NÃO
Controle de recuperação de falhas	SIM	SIM	NÃO	SIM
Integração compartilhada	NÃO	NÃO	SIM	SIM
Acesso direto a base de dados	SIM	SIM	NÃO	NÃO
Desempenho por registro (em segundos)	0,00665	38 vezes mais lento	13 vezes mais lento	31 vezes mais lento

Tabela 7.1 – Tabela de análise de tecnologias de integração
Fonte: Próprio autor

Por fim, cabe ressaltar que a opção por uma técnica em relação à outra dar-se-á considerando as necessidades de uma determinada integração. Neste trabalho, o cenário tinha como base a necessidade de que todas as empresas do grupo trabalhassem com a mesma tabela de valores para materiais e que as alterações fossem replicadas no menor tempo possível. Com base nestas premissas, a técnica que obteve melhor resultado foi a integração direta entre bancos de dados. Os principais fatores que contribuíram para sua escolha foram: atualização de dados *on-line*, o melhor desempenho de integração e controle de falhas. Entretanto, é necessário lembrar que os resultados obtidos pela integração têm como pré-requisito a existência de uma conexão de rede dedicada entre as duas empresas, sem a qual não seria possível construir a integração.

Apenas para registro, cabe mencionar que os resultados obtidos durante a execução das integrações utilizando BPEL, no que diz respeito ao item “Desempenho por registro”, poderiam ter registrado melhor desempenho, visto que o ambiente não atendia a todos os requisitos de hardware recomendados pelo fabricante[7].

[7] O site da Oracle com os requisitos de hardware pode ser acessado em http://www.oracle.com/technetwork/middleware/soasuite/learnmore/vmsoa-172279.html.

CONCLUSÃO

A integração de sistemas faz parte do cotidiano das empresas atuais, visto que é comum estas empresas possuírem mais de um sistema para atender suas necessidades. Vários são os motivos que levam as empresas a aquisição de mais de um sistema, entre eles é possível citar: necessidade de sistemas especialistas, altos custos envolvidos na manutenção/evolução dos sistemas legados, necessidade de profissionais com conhecimento das melhores práticas do negócio. Em suma, são muitos os motivos e todos eles têm um ponto em comum: à necessidade das empresas manterem-se competitivas.

Para as empresas manterem-se competitivas, mais do que nunca, precisam conhecer a forma com que se organizam para execução de suas atividades. O surgimento de ferramentas de modelagem de processos, torna possível entender como os processos das empresas interagem, sejam estes manuais ou sistêmicos.

É notável o quanto os sistemas dentro de uma organização por vezes são responsáveis por operacionalizar uma grande quantidade de processos. Esta realidade valoriza ferramentas de modelagem que possam representar de que forma os sistemas automatizam estes processos ou até mesmo interagem com outros sistemas. Desta forma, aproveitando o conhecimento agregado na modelagem de processos manuais (notações), esta nova geração de ferramentas de modelagem que agregam a interação entre os sistemas, tem recebido cada vez mais destaque.

Neste contexto as ferramentas de BPEL apresentam grande contribuição, permitindo automatizar e gerir as relações e integrações existentes entre os sistemas de uma organização. As ferramentas de BPEL destacam-se também por serem aderentes aos padrões de notação até então utilizados e por orquestrarem as relações ou integrações entre estes sistemas, disponibilizando vários recursos de conectividade. A utilização de uma ferramenta de BPEL durante a execução deste trabalho possibilitou confirmar estes recursos, como também mostrar que a partir delas é possível elaborar processos de integração utilizando as técnicas mais utilizadas no mercado.

Apesar de existirem ferramentas que atendem boa parte das necessidades de integrações entre sistemas, ainda assim, é nítida a dificuldade em definir que ferramenta pode desempenhar melhor a atividade de integração para um determinado contexto. A construção e aplicação de técnicas de integração disponíveis no mercado para solucionar o cenário

proposto provaram que mais de uma técnica pode ser aplicada para um mesmo cenário, sendo necessário ponderar as vantagens e riscos envolvidos em cada uma delas.

Como sugestão para trabalhos futuros cabe ainda avaliar outras tecnologias de integração de mercado como, por exemplo, o ODI (*Oracle Data Integrator*), realizando uma análise comparativa entre as duas tecnologias (BPEL x ODI) ou até mesmo, avaliar suas potencialidades através do uso de cenários de integração. Outra possibilidade é - ainda com a utilização de BPEL - utilizar cenários variados onde seja possível identificar em que contexto cada uma das tecnologias melhor desempenha suas atividades.

REFERÊNCIAS BIBLIOGRÁFICAS

AMARAL, Vinicius. 2006. **BPM - Afinal, o que é (e o que não é) isso?**. Disponível em: <http://www.baguete.com.br/artigosDetalhes.imprime.php?id=117>. Acesso em 05/06/2010.

BITENCOURT, Maurício. 2007. **Modelagem de Processos com BPMN**. Disponível em: <http://www.projeler.com.br/download/pdf/artigo_bpmn_projeler_mauricio_bitencourt.pdf>. Acesso em 23/08/2010.

BORTOLINI, Rafael. 2006. **Padronizando Processos: BPMN, BPML, XPDL e BPEL**. Disponível em: <http://www.baguete.com.br/artigosDetalhes.php?id=136>. Acesso em 23/08/2010.

CARNEIRO, Sérgio Roberto. 2008. **Oracle Heterogeneous Services.** Disponível em:<http://www.devmedia.com.br/articles/viewcomp.asp?comp=11070>. Acesso em: 31/03/2011.

CAVALCANTI, Janeide**Albuquerque. Integração de sistemas heterogêneos através da interface com o usuário.** Campina Grande, PB. 15 p. Dissertação de Mestrado, Universidade Federal da Paraíba, 2001.

COLCHER, Raul e VALLE, André. **Guia de IDI e Comercio Eletrônico**. Rio de janeiro. Publicação:Simpro Brasil, 2000.

CUMMINS, Fred A. Integração de sistemas: Enterprise Integration: arquiteturas para Integração de Sistemas e Aplicações Corporativas. Rio de Janeiro: Campus, 2002.

CUNHA, Davi. 2002. **Web Services, SOAP e Aplicações Web**. Disponível em: <http://devedge-temp.mozilla.org/viewsource/2002/soap-overview/index_pt_br.html>. Acesso em 13/10/2010.

D. REIS, Antonio e ROCHA, Jose F e GAMEIRO, Atilio S. CARVALHO, José P. **Sistemas de comunicação Síncrona e Assíncrona de dados**. Departamento de Física, Universidade da Beira Interior Covilhã.

EAN, Associação Brasileira de Automação Comercial. 2010. **EDI - Intercâmbio Eletrônico de Dados**. Disponível em: <http://www.eanbrasil.org.br/main.jsp?lumPageId=FF8080810CC51BE1010CD4D64DB0105A>.Acessoem 24/08/2010.

FEDERAL COORDINATING COUNCIL FOR SCIENCE (Washington, D.C.).**Office of Science and Technology Policy. High performance computing & communications: toward a national information infrastructure**. Washington D.C., 1994. 176 p. (Committee on Physical, Mathematical, and Engineering Sciences Report).

GONÇALVES, José Ernesto Lima. 2000. **Processo, que processo**. Disponível em: <http://www.projeler.com.br/download/pdf/processo_que_processo.pdf>. Acessoem 23/08/2010.

HOHPE, Gregor. 2010. **Patterns and Best Practices for Enterprise Integration**. Disponível em: <http://www.eaipatterns.com/>. Acesso em 23/08/2010.

KIOSKEA. **Workflow – Gestão de Processos de Negócio.** Disponível em: <http://pt.kioskea.net/contents/entreprise/workflow.php3>. Acesso em 23/11/2010.

MELO,Heberton. 2010. **Integrando bancos de dados Utilizando Linked Server do SQL Server 2008**.Rio de Janeiro. Editora DevMedia 2010.

MESQUITA, Cláudia do Socorro Ferreira, BRETAS, Nazaré Lopes. **PANORAMA DA INTEROPERABILIDADE NO BRASIL**. Brasília. Editora: MP/SLTI, 2010.

MICROSOFT. 2011. **ConfiguringLinked Servers.** Disponível em: <http://msdn.microsoft.com/en-us/library/aa213778%28v=sql.80%29.aspx>. Acesso em: 31/03/2011.

NETO, Jorge AbilioAbinader, LINS, Rafael Dueire. **WEB SERVICES EM JAVA**. Rio de Janeiro. Editora Brasport, 2006.

OLIVEIRA, Carlos H. 2010. **Implementando Web Services usando PHP e SOAP**. Disponível em: <http://www.carlosholiveira.com.br/tecnologia/1/?q=node/5>. Acesso em 13/10/210.

OLIVER, Paulo Roberto Costa. **Projetos de ECM/BPM – Os Segredos da Construção.** São Paulo. Editora Biblioteca 24X7, 2010.

ORACLE. 2009. Integração de dados em tempo real possibilita melhorar as informações do negócio e disponibilidade para sistemas de missão crítica. Disponível em: <http://www.oracle.com/global/br/corporate/press/2009_nov/pr_br_091130.html>. Acesso em 02/09/210.

PAMPLONA, Vitor Fernando. **Web Services. Construindo, disponibilizando e acessando Web Services via J2SE e J2ME**. 2010. Disponível em: <http://javafree.uol.com.br/artigo/871485/>. Acesso em: 24/08/2010.

RABELO, Ricardo J. 2009. **Business Process Modeling Notation –(BPMN) .** Disponível em:
<http://www.das.ufsc.br/~rabelo/Ensino/DAS5316/MaterialDAS5316/PARTE2/BPM/BPMN%20%e2%80%93%20Business%20Process%20Modeling%20Notation%202009.pdf>.
Acesso em 27/11/2010.

REIS, Glauco dos Santos. Modelagem **de Processos de Negócio com BPMN – Curso Completo**. São Paulo. Editora PortalBMP, 2008.

SAMPAIO, Cleuton. **SOA e Web services em Java**. Rio de Janeiro. Editora Brasport, 2006.

SAP. 2010. **SAP NetWeaver EXCHANGE INFRASTRUCTURE**. Disponível em: <http://www.sap.com/brazil/platform/netweaver/exchangeinfrastructure/index.epx>.
Acessoem: 04/09/2010.

SHETH, A. P. 1998**. Changing Focus on Interoperatibily in Information Systems: From System, Syntax, structure to Semantics**. Disponível em: <http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.18.629&rep=rep1&type=pdf>. Acesso em 23/04/2011.

SILVA, Luciano. 2010. **Apresentando o BPEL com o Adapter File**. Disponível em: <http://www.lucianosilva.com/2009/02/13/apresentando-o-bpel-com-o-adapter-file/>.Acesso em: 23/08/2010.

UFPE. 2010**. INTEROPERABILIDADE E HETEROGENEIDADE DE SISTEMAS**. Disponível em: <http://www.cin.ufpe.br/~if696/referencias/integracao/_Interoperabilidade_de_Banco_de_Dados_e_Sistemas_Heterogeneos.pdf>. Acessoem: 27/11/2010.

W3C, World Wide Web Consortium. 2009. **Web Services Architecture**. Disponível em: <http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/>. Acesso em 24/08/2010.