UNIVERSIDADE CATÓLICA DE SANTOS

ADILSON JERÔNIMO DA SILVA

Negociação entre Aplicações de Comércio Eletrônico utilizando o padrão ebXML.

Santos
2006

ADILSON JERÔNIMO DA SILVA

Negociação entre Aplicações de Comércio Eletrônico utilizando o padrão ebXML.

Dissertação apresentada à Universidade Católica de Santos – UNISANTOS, para obtenção do título de Mestre em Informática.

Área de Concentração: Ciência da Computação.

Orientador: Dr. Manuel de Jesus Mendes

Santos

2006

Dados Internacionais de Catalogação
Sistema de Bibliotecas da Universidade Católica de Santos – UNISANTOS
SibiU

S586n Silva, Adilson Jerônimo da

Negociação entre Aplicações de Comércio Eletrônico utilizando o padrão ebXML / Adilson Jerônimo da Silva – Santos:

[s.n.] 2006

109 f.; 30 cm. (Dissertação de Mestrado – Universidade Católica de Santos, Programa em Informática)

I. Silva, Adilson Jerônimo. II. Negociação entre Aplicações de Comércio Eletrônico utilizando o padrão ebXML.

CDU 681.3:004(043.3)

AGRADECIMENTOS

Inicialmente gostaria de agradecer ao meu orientador, Professor Dr. Manuel de Jesus Mendes, cujo apoio e incentivo foram primordiais para o desenvolvimento e conclusão deste trabalho.

A minha família, pelo apoio e compreensão, principalmente quanto às muitas horas em que não pude estar ao lado deles, devido estar dedicado ao desenvolvimento deste projeto.

Aos diversos amigos que de alguma forma contribuíram com informações e apoio, principalmente aos colegas de trabalho, em especial ao Luis Yasuda e ao Mario.

Um agradecimento especial à Isabel, minha filha, ainda “engatinhando” em informática, mas que me ajudou bastante na parte de tratamento gráfico para a inclusão de algumas figuras neste trabalho.

RESUMO:

A necessidade de cortar custos e aumentar a produtividade, aliada à crescente disponibilização de padrões que visam garantir a interoperabilidade entre aplicações distintas, têm levado um crescente número de empresas a utilizar a Internet como principal meio de comunicação com os seus parceiros comerciais, estimulando o crescimento do comércio eletrônico B2B e conseqüentemente aumentando o interesse na automatização da fase de negociação, que normalmente precede a fase de interação eletrônica entre os parceiros comerciais, buscando assim a agilização de todo o processo de colaboração.

O desenvolvimento de negociações eletrônicas de forma automatizada implica na definição de regras e mensagens que devem ser utilizadas pelas partes, durante o processo de negociação. O padrão ebXML (*Electronic Business Extensible Markup Language*) especifica os componentes CPP (*Collaboration Protocol Profile*) e CPA (*Collaboration Protocol Agreement*) para serem utilizados na fase de negociação. O CPP descreve o perfil da empresa com relação a suas capacidades técnicas para o estabelecimento de um processo de negócio com outras empresas e o CPA é o acordo resultante da intersecção dos CPP dos parceiros comerciais envolvidos em um processo de colaboração de negócios.

Estes componentes podem ser utilizados na automatização do processo de negociação, mas a especificação ebXML deixa em aberto a forma de geração automática do acordo de colaboração (CPA), assim como a forma de interação entre as empresas nesta fase. O problema é como efetuar a automatização desta interação e gerar automaticamente o acordo (CPA), a partir dos perfis (CPP) de cada empresa.

Neste trabalho são analisadas as especificações ebXML e elaborada uma proposta para agilizar o processo de negociação, através da automatização da troca de mensagens que ocorre durante a fase de negociação e a geração de informações para auxiliar um negociador humano a tomar uma decisão final sobre um determinado processo de negociação, com a conseqüente geração de um acordo de colaboração (CPA).

A proposta abrange a definição de um protocolo de negociação baseado em XML, para a troca de mensagens durante a fase de interação e a modelagem de um sistema para o controle destas mensagens e comparação das propostas de acordo, até a finalização do processo de negociação e, em caso de sucesso, com a geração do documento de acordo de colaboração. Para a validação da proposta foi desenvolvido um protótipo utilizando a linguagem Java e plataforma J2EE, abrangendo requisitos básicos para suportar um processo de negociação. Este protótipo foi implementado em um ambiente de testes com um cenário de uso envolvendo um processo de negociação entre duas empresas, visando estabelecer um processo de negócio de compra e venda.

Palavras-chave: Acordo de Colaboração**;** Comércio eletrônico; CPA; CPP; ebXML; EDI; Negociação eletrônica; Serviços Web; XML.

ABSTRACT

The need to cut costs and to increase the productivity, allied to growing readiness of interoperability patterns among different applications, pushed a growing number of companies to use the Internet as the main channel of communication with their commercial partners. As a result, the growth of the electronic B2B trade was stimulated and, consequently, the interest in the automation of the negotiation phase increased, that usually precedes the phase of electronic interaction among the commercial partners, looking for speed up the collaboration process.

The development of electronic negotiations in an automated way implicates in the definition of rules and messages that should be used by the parts, during the negotiation process. The ebXML (Electronic Business Extensible Markup Language) specifies the components CPP (Collaboration Protocol Profile) and CPA (Collaboration Protocol Agreement) for their use in the negotiation phase. The CPP describes the profile of a company businesses and the CPA is the agreement resulting from the intersection of the commercial partners' CPP, involved in a process of business collaboration.

These components can help in the negotiation process automation, but the specification ebXML leaves open how to implement automatically the collaboration agreement (CPA), as well as the ways of interaction among the companies in this phase. The problem is how to realize the automation of this interaction and how to generate the agreement automatically (CPA), starting from the profiles (CPP) of each company.

In this work the specifications ebXML are analyzed and a proposal to activate the negotiation process elaborated, through the automation of the exchange of messages that happens during the negotiation phase and the generation of information to aid a human negotiator to make a final decision on a certain negotiation process, with the consequent generation of an collaboration agreement (CPA).

The proposal includes the definition of a negotiation protocol based on XML for the exchange of messages during the interaction phase, and the modeling of a system for the control of these messages and the comparison of the proposals of agreement to the finalization of the negotiation process and, in case of success, with the generation of the collaboration agreement document . For the proposal validation, a prototype was developed using the language Java and the middleware J2EE, including basic requirements to support a negotiation process. This prototype was implemented in an test environment, with real use scenery, involving a process of electronic negotiation between a company and an financial institution, in a buyer-seller collaboration process.

Key-words: Collaboration Agreement; Electronic Business; CPA; CPP; ebXML; EDI; Electronic Negotiation; Web Service; XML.

Índice

Lista de Figuras

Lista de abreviaturas e siglas

API	Application Program Interface
BPS	Business Process Specification
BPSS	Business Process Specification Schema
BPIM	Business Process and Information Meta Model
BPMN	Business Process Management Notation
B2B	Business to Business
B2C	Business to Consumer
CORBA	Common Object Request Broker Architecture
CPA	Collaboration Protocol Agreement
CPP	Collaboration Protocol Profile
DOM	Document Object Model
EAI	Enterprise Application Integration
ebMS	Electronic Business Message Service
ebXML	Electronic Business Extensible Markup Language
EDI	Electronic Data Interchange
DTD	Document Type Definition
FTP	File Transfer Protocol
HTTP	Hyper Text Transfer Protocol
HTML	Hyper Text Markup Language
J2EE	Java 2 Enterprise Edition
JAXP	Java Aplication for XML Processing
JDBC	Java Database Connectivity
JDOM	Java Document Object Model
MIME	Multipurpose Internet Mail Extensions
NSS	Sistemas de Suporte a Negociação
NDSS	Sistemas de Suporte a Decisão em Negociação
OASIS	Organization for the Advancement of Structured Information Standards
RPC	Remote Procedure Call
RUP	Rational Unified Process

SMTP	Simples Mail Transfer Protocol
TCP/IP	Transmission Control Protocol/Internet Protocol
UDDI	Universal Description, Discovery and Integration
UML	Unified Modeling Language
UMM	UN/CEFACT Modeling Methodology
UN/CEFACT	United Nations Centre for Trade Facilitation and Electronic Business
URI	Uniform Resource Identifiers
URL	Uniform Resource Locator
UTF	Unicode Transformation Format
WS	Web Service
WSDL	Web Service Description Language
W3C	World Wide Web Consortium
XML	Extensible Markup Language
XSL	Extensible Stylesheet Language
XSLT	Extensible Stylesheet Language Transformation

1. INTRODUÇÃO

1.1. Definição do Problema

A troca de informações entre empresas utilizando um meio eletrônico é um processo que iniciou-se nos anos 60, com o aparecimento dos primeiros sistemas de intercâmbio de documentos eletrônicos denominados EDI (*Eletronic Document Interchange*) [Liang *et al*., 2004].

A necessidade das empresas integrarem as suas cadeias de fornecedores para cortar custos e ganhar produtividade implicou no desenvolvimento de vários padrões de EDI, cada qual atendendo a necessidades específicas de formatação de documentos utilizados em segmentos verticais da indústria.

Com a popularização da Internet e o desenvolvimento do padrão XML (*Extensible Markup Language*), vêm sendo desenvolvida uma quantidade crescente de especificações baseadas neste padrão, voltadas principalmente para o estabelecimento da interoperabilidade entre sistemas diversos, através da Internet, contribuindo para impulsionar o comércio eletrônico entre empresas, conhecido como B2B (*Business to Business*) .

O mais promissor dos padrões voltados para o B2B é o padrão de Linguagem de Marcação Extensível para Negócios Eletrônicos [Liang *et al*.,2004], conhecido como ebXML (*electronic business XML)* que é patrocinado por diversas organizações, inclusive pela Nações Unidas através do UN/CEFACT - *United Nations Centre for Trade Facilitation and Electronic Business*, que incentiva a sua utilização em projetos de governo eletrônico pelo mundo e que será utilizado extensivamente neste trabalho.

O aumento da importância do comércio eletrônico B2B tem ocasionado um crescente interesse em automatizar o processo de negociação, que normalmente precede a fase de interação eletrônica entre os parceiros comerciais [*Bertolini et. al.*,2001].

Negociação é um processo através do qual duas ou mais partes interagem visando um ganho mútuo [Segev,2001], mas para isto, é necessária a criação de uma infra-estrutura que permita a entidades independentes interagirem e estabelecerem negociações eletrônicas de forma automatizada [Bertolini *et. al.*,2001].

Essa infra-estrutura abrangeria uma variedade de aspectos, incluindo a definição de um protocolo para negociação, que incluiria a definição de atores, regras e fases de negociação; definição de uma linguagem para estabelecer regras de negociação e expressar propostas.

Os processos de negociação entre empresas (B2B) são importantes por que, dependendo de como são desenvolvidos, podem levar muito tempo para serem concluídos e criar impactos em operações internas, influenciando nos custos e na produtividade dos negócios da empresa [Byde e Piccinelli,2002].

Considerando-se a necessidade de agilização, um dos principais aspectos a considerar em um processo de negociação, é a capacidade de automaticamente detectar incompatibilidades entre os processos propostos pelas partes e negociar possíveis ajustes que cada parte pode adotar, para se fechar um acordo.

Os processos de negociação entre empresas são efetuados basicamente através da intervenção humana, gerando morosidade no processo, baixa produtividade e aumento de custos [Segev,2001].

Neste contexto, o padrão ebXML [ebXMLTA,2001] traz uma proposta interessante na abordagem do processo de negociação, através da utilização dos componentes ebXML denominados Perfil do Protocolo de Colaboração – CPP (*Collaboration Protocol Profile*) e Acordo do Protocolo de Colaboração – CPA (*Collaboration Protocol Agreement*).

O padrão ebXML especifica, na sua arquitetura técnica [ebXMLTA,2001], a utilização dos componentes CPP e CPA para a fase de negociação, mas deixa em aberto a geração automática do acordo de colaboração, assim como a forma de interação das empresas nesta fase.

Um acordo de colaboração entre parceiros comerciais que utilizem o padrão ebXML, representa a interseção das capacidades especificadas nos perfis de colaboração de cada parte (CPP), onde estão descritos diversos itens relativos à infra-estrutura técnica e capacidades comerciais que cada empresa dispõe em seus processos de negócio.

Segundo [Hofreiter e Huemer,2003], o problema é desenvolver os mecanismos para processar esta intersecção e gerar o acordo (CPA), levando a uma automatização do processo de negociação.

As especificações do padrão ebXML, tanto da arquitetura técnica [ebXMLTA,2002] quanto do acordo de colaboração [ebXML CPPA, 2002] ressaltam que teoricamente este processo de geração do acordo de colaboração pode ser automatizado, mas há necessidade de se validar esta hipótese.

Alguns trabalhos que abordam este problema no contexto do padrão ebXML, citam a necessidade de geração automática do acordo de colaboração, como Hofreiter [Hofreiter e Huemer,2003] e Byde [Byde e Piccinelli,2002] mas não propõem um modelo para automatização deste processo.

1.2. Motivação

A morosidade a que está sujeito um processo de negociação pode afetar os negócios de uma empresa, principalmente se esta empresa depende destas negociações para aquisição de uma determinada matéria prima ou algum serviço fornecido por terceiros, por exemplo, para o desenvolvimento do seu processo de negócio.

A rapidez de comunicação proporcionada por um meio eletrônico como a Internet, pode contribuir para agilizar o estabelecimento de um processo de negociação, mas, utilizando modelos ainda bastante dependentes da intervenção humana, esta agilidade pode ser comprometida [Segev,2001].

Conforme previsões do Gartner Group [Gartner, 2003], a utilização de tecnologias como a de serviços web (*Web Services*) e a consolidação de padrões desenvolvidos para atender processos de colaboração eletrônica entre empresas deverão acentuar o crescimento do segmento B2B (*business to business*) nos próximos anos.

Como normalmente os processos de interação comercial entre empresas são precedidos por um processo de negociação, volumes crescentes de negócios canalizados através do meio

eletrônico, implicarão na necessidade de agilização desta fase de negociação, visando acompanhar o dinamismo e a rapidez exigida pelas necessidades de negócio.

A crescente importância do mercado B2B e a conseqüente necessidade de agilização dos processos de negociação eletrônica, aliada à existência de documentos padronizados nas especificações do padrão ebXML que podem ser utilizados durante esta fase de negociação para automatizar a geração de um acordo de colaboração, motivou o desenvolvimento deste trabalho.

A escolha do padrão ebXML para abordar o problema de automatização do processo de negociação entre aplicações de comércio eletrônico, também deve-se ao fato de que este padrão está bem estruturado com relação a especificação dos seus componentes, tendo sido inclusive normatizado pela ISO (*International Standard Organization*) em março de 2004, através da norma ISO 15000.

Além destes motivos, o padrão ebXML também foi criado com um propósito nobre, ou seja, democratizar o acesso ao comércio eletrônico B2B, possibilitando que empresas de pequeno porte também possam, através da Internet, participar de comunidades de comércio eletrônico.

1.3. Objetivos

A proposta deste trabalho é analisar as especificações do padrão ebXML e verificar como o documento de acordo de colaboração (CPA), especificado neste padrão para ser utilizado na fase de negociação, pode ser gerado a partir do perfil de colaboração (CPP) dos parceiros comerciais envolvidos.

Pretende-se também que a proposta idealizada possa automatizar a fase de interação, ou seja, o envio de mensagens durante o processo de negociação, com o objetivo de auxiliar a participação humana nesta fase do processo.

Neste contexto, é proposto um sistema para automatizar a geração do acordo de colaboração, assim como o processo de interação desenvolvido durante a fase de negociação para a elaboração deste acordo, fornecendo informações a um negociador humano, para que este possa tomar as decisões com relação ao desenvolvimento e conclusão de uma determinada negociação.

A proposta abrange a elaboração de um protótipo deste sistema, desenvolvido na plataforma J2EE(*Java 2 Enterprise Edition*).

1.4. Organização do Trabalho

O trabalho encontra-se organizado de acordo com a seguinte estrutura de capítulos:

No capítulo 2 são descritos aspectos gerais relativos aos padrões utilizados na implementação de serviços eletrônicos, como o padrão XML e os padrões derivados a partir do XML, tais como o SOAP (*Simple Object Acess Protocol*), WSDL (*Web Service Description Language*) e UDDI (*Universal Description, Discovery and Integration*), assim como os conceitos e a forma de interação dos componentes do padrão ebXML.

No capítulo 3 são apresentados os conceitos básicos de negociação eletrônica e detalhado o processo de negociação com os componentes CPP e CPA do padrão ebXML.

No capítulo 4 são abordados os aspectos necessários para a elaboração de uma proposta visando atender ao problema de automatizar a geração do acordo de colaboração (CPA) entre aplicações que utilizem o padrão ebXML, abrangendo-se a definição de um protocolo básico de negociação e o respectivo esquema XML, assim como os requisitos funcionais necessários para viabilizar a automatização do processo de negociação.

No capítulo 5 é abordada a implementação de um protótipo desenvolvido para validação da proposta, descrevendo-se a sua utilização, os testes realizados e os respectivos resultados obtidos.

No capítulo 6 são descritas as considerações finais sobre o trabalho, abrangendo a sua contribuição e os trabalhos futuros que poderão ser desenvolvidos a partir deste trabalho.

2. ASPECTOS GERAIS DE PADRÕES UTILIZADOS EM SERVIÇOS ELETRÔNICOS

Neste capítulo apresentaremos os principais conceitos dos padrões utilizados em serviços eletrônicos, assim como do padrão ebXML, sobre o qual este trabalho está baseado.

2.1. Introdução

Com a crescente utilização e disseminação da Internet, diversos serviços eletrônicos foram disponibilizados via Web, com o objetivo de fornecer informações, produtos e interatividade.

As empresas visualizaram que este meio eletrônico poderia ser utilizado não somente como mais um canal de vendas e informações, mas também como uma forma de integração com parceiros comerciais e até de áreas e sistemas internos das próprias empresas.

Esta utilização gerou a necessidade de desenvolvimento de padrões e aplicações que pudessem assegurar o transporte, a segurança e a troca de mensagens entre os diversos sistemas envolvidos nos processos de colaboração eletrônica entre empresas, conhecido como B2B (*Business to Business*).

Para atender a necessidade de interoperabilidade entre aplicações distintas, o mercado passou a utilizar o padrão XML – *eXtensible Markup Language*, desenvolvido pelo W3C (*World Wide Web Consortium*) em 1996, a partir do qual, diversos outros padrões foram criados, tais como o SOAP (*Simple Object Acess Protocol*), WSDL (*Web Service Description Language*), UDDI (*Universal Description, Discovery and Integration*) e o ebXML (*Eletronic Business Extended Markup Language*), visando atender a necessidades específicas, dentro do universo de serviços eletrônicos.

Os serviços eletrônicos (*e-services*) criados a partir da utilização destes padrões, podem ser formalmente definidos como serviços oferecidos eletronicamente através da Internet por um provedor de serviços [Tizzo *et al*., 2004].

Segundo [Lhotka,2005], serviços podem ser definidos como uma coleção de procedimentos atômicos ou métodos, que executam uma determinada função quando são requisitados através de uma chamada.

2.2. O padrão XML (Extensible Markup Language)

A linguagem de marcação extensível, conhecida pela sigla XML (***Ex**tensible **M**arkup **L**anguage*), é um padrão desenvolvido pelo W3C (*World Wide Web Consortium*) a partir de um outro padrão utilizado na indústria de editoração denominado SGML (*Standard Generalized Markup Language*), que permite a definição de marcações personalizadas, através do uso de marcadores de texto.

A XML, assim como a HTML (*HyperText Markup Language*), são chamadas de linguagens de marcação por causa do modo como elas estruturam o texto de um documento, cercando partes do texto com um conjunto de marcadores que funcionam como etiquetas, conhecidas como *tags*, que são utilizadas para atribuir determinadas características a informação contida entre estes marcadores.

O objetivo principal do W3C, conforme declarado nas especificações do padrão XML [W3C,2004a], foi definir uma linguagem portável para descrição de dados através de um novo modelo de marcação que pudesse ser utilizado na Web, fornecendo ao autor do documento recursos para a definição e criação de seus próprios rótulos de marcação (*tags*) e atributos, em vez de estar restrito ao esquema de marcação da HTML (*HyperText Markup Language*), que utiliza *tags* fixas.

O XML é considerado uma metalinguagem, ou seja, uma linguagem que permite definir novas linguagens através da criação de marcações específicas [Niemeyer e Knudsen, 2004], possibilitando, por exemplo, a criação de uma linguagem que descreva dados matemáticos, químicos ou comerciais, através de marcações personalizadas para estas áreas.

Um documento XML pode conter elementos que sejam eles próprios, formados por outros elementos, resultando em uma estrutura hierárquica de elementos (estrutura em árvore), dentro do documento XML, que pode ser utilizado para descrever um cliente, uma ordem de compra ou qualquer outra entidade física ou lógica, que necessite ser descrita em um documento.

O documento XML pode ser completo, bem formado, ou seja, com todas as *tags* de inicio e fechamento declaradas corretamente, mas para ser válido, necessita estar de acordo com um esquema que contenha as definições válidas para este documento.

Um *parser* é uma aplicação que lê um documento e entende suas convenções de formatação, ou seja, uma aplicação que funciona como um analisador gramatical para validar documentos XML, verificando sua estrutura e regras de marcação de acordo com as definições contidas em um esquema [Niemeyer e Knudsen, 2004].

Um esquema pode ser escrito através de um DTD (Definição do Tipo de Documento) ou de um esquema XML (*XML Schema*), e especifica as restrições de integridade que deverão ser observadas para que um documento possa ser considerado válido [W3C, 2004a].

2.2.1. Definição do Tipo de Documento – DTD

Conforme as especificações do padrão XML [W3C, 2004a], o DTD consiste de um documento de tipo texto, que define as restrições da estrutura lógica que devem ser seguidas na elaboração de um documento XML.

A função do DTD é definir todas as *tags* que um documento XML pode conter, determinando a ordem em que estas *tags* devem aparecer, se são obrigatórias ou opcionais, os seus atributos, assim como qualquer entidade que possa ser utilizada no documento.

2.2.2. Esquema XML (XML Schema)

O esquema XML é uma especificação mais recente do que o DTD, tendo sido originalmente proposto pela Microsoft, tornou-se uma recomendação oficial do W3C em maio de 2001.

Um esquema XML objetiva, assim como o DTD, descrever a estrutura e as restrições de um documento XML.

No entanto, conforme descrito nas especificações do esquema XML [W3C,2004b], há algumas diferenças básicas entre eles:

- Um esquema XML utiliza a mesma sintaxe do padrão XML, consistindo portanto em um documento XML, enquanto um DTD utiliza uma sintaxe própria.
- A especificação da sintaxe a ser utilizada em um esquema XML providencia funcionalidades além daquelas providenciadas por um DTD, ou seja, possibilita mais recursos para a definição de restrições a serem aplicadas aos elementos de um documento XML.
- O esquema XML possibilita que o autor do esquema defina os seus próprios tipos, enquanto em um DTD não há recursos para controlar que tipo de informação determinado elemento ou atributo pode conter.
- As declarações em um DTD são globais (definição válida para todo o documento), não permitindo, portanto, a definição de dois elementos diferentes com o mesmo nome, mesmo se aparecerem em contextos separados. Um esquema XML permite tanto a definição de elementos globais quanto a definição de elementos locais, com significado restrito a um determinado contexto do documento.

No esquema XML é possível especificar o tipo de texto que se deseja que um elemento contenha, ou seja, o seu conteúdo, atribuindo a ele uma determinada definição de tipo simples (*simpleType*) ou de tipo complexo (*complexType*).

Elementos de tipo simples são aqueles que podem conter apenas texto. Existem diversos tipos simples definidos nas especificações do esquema XML, como, por exemplo, *date, integer e string*, entre outros, que podem ser utilizados para a definição do conteúdo de um elemento.

O esquema XML também permite a construção de tipos simples personalizados, para se controlar de uma forma mais rígida, por exemplo, o conteúdo de um elemento. A definição abaixo restringe o conteúdo dos elementos definidos com o nome “codigopostal” a uma *string* com um padrão de cinco dígitos, seguidos por um hífen e mais quatro dígitos:

```
<xsd:simpleType name=”codigopostal”>
    <xsd:restriction base=”xsd:string”>
    <xsd:pattern value=”\d{5}(-\d{4})?”/>
    </xsd:restriction>
</xsd:simpleType>
```

Por outro lado, elementos do tipo complexo são aqueles que podem conter outros elementos ou que contêm atributos.

No exemplo abaixo, temos a definição de um tipo complexo denominado “Endereço”, contendo vários outros elementos de tipo simples (rua, cidade, estado, cep):

```
<xsd:complexType name=”Endereço”>
    <xsd:sequence>
        <xsd:element name=”rua”     type=”xsd:string”/>
        <xsd:element name=”cidade” type=”xsd:string”/>
        <xsd:element name=”estado” type=”xsd:string”/>
        <xsd:element name=”cep”     type=”xsd:string”/>
    </xsd:sequence>
    </xsd:complexType>
```

A conseqüência da definição acima é que qualquer elemento que aparecer em um documento XML, declarado como sendo deste tipo “Endereço”, deverá consistir de quatro elementos denominados rua, cidade, estado e cep, conforme especificado pela declaração *element name*, e deverá ter os valores de seu conteúdo de acordo com o tipo especificado (*string*), além de obrigatoriamente aparecer na mesma seqüência em que foram declarados.

2.2.3. Espaço de nomes XML (XML *namespace*)

Um simples documento XML pode conter elementos que poderão ser utilizados por múltiplas aplicações, que por sua vez poderão combinar esses elementos com elementos de outros documentos, gerando assim um novo documento XML, onde haverá a mistura de elementos de diversas fontes.

Quando ocorre esta situação, existe a possibilidade de que dois ou mais documentos tenham a especificação de nomes redundantes, ou seja, pode ocorrer uma colisão de elementos com o mesmo nome.

Para evitar este problema, utiliza-se um mecanismo denominado espaço de nomes (*namespace*), que permite que um analisador gramatical (*parser*) possa diferenciar os elementos, quando estiver analisando uma instância de um determinado documento.

Um espaço de nomes XML é um modo de dizer qual esquema está sendo utilizado para um dado elemento, permitindo que elementos definidos em esquemas distintos possam ser misturados.

A notação padrão "xmlns:prefix" é utilizada para declarar um *namespace* associando-o a um prefixo, o que normalmente é efetuado dentro do elemento raiz de um documento XML [Harold e Means, 2004, p.60] ou mesmo aplicado a um elemento e todos os seus filhos.

Um prefixo é mapeado para uma URL (*Universal Resource Locator*) e anexado em cada elemento e atributo de um documento XML, indicando o espaço de nomes a que pertence. Exemplo:

```
<?xml version = “1.0” encoding = “UTF-8”?>
<mensagens xmlns:msg = “http://www.endereço.com/mensagens/1.0”
                    xmlns:ped = “http://www.endereço.com/pedido”
                    xmlns = “http://www.endereço.com/exemplo”>
        <msg:numero>001</msg:numero>
        <ped:codigo>xy</ped:codigo>
        <nome>Abcd</nome>
</mensagens>
```

No exemplo acima foram declarados três *namespaces*: o namespace padrão (*default*) “http://www.endereço.com/exemplo“, que não está associado com nenhum prefixo, o namespace “http://www.endereço.com/mensagens/1.0”, que está associado ao prefixo “msg” e o namespace “http://www.endereço.com/pedido“, associado ao prefixo “ped”.

Os prefixos foram utilizados para qualificar os componentes do documento XML como pertencentes a um determinado *namespace*, observando-se que o elemento “nome” não possui prefixo, estando associado ao *namespace* padrão.

O *namespace* permite distinguir entre definições e declarações de diferentes vocabulários, como por exemplo, distinguirmos entre uma declaração de um nome “*element*”, pertencente ao vocabulário padrão da linguagem de esquema XML, cujo *namespace* é http://www.w3.org/2001/XMLSchema e uma declaração de um nome “*element*”, pertencente ao vocabulário de uma hipotética linguagem de química, que está definido em outro espaço de nome [W3C, 2004b].

Existe ainda uma variedade de padrões definidos dentro do universo representado pelo padrão XML, tais como a linguagem de folha de estilo XSL (*Extensible Stylesheet Language*), que define um conjunto de elementos que descrevem como os dados devem ser formatados para gerar documentos impressos e até arquivos de áudio, a XSLT (*Extensible Stylesheet Language for Transformations*), que é um vocabulário XML que descreve como converter um documento XML em um documento HTML ou em outro documento XML, entre outros [Tidwell, 2002a].

2.2.4. O intercâmbio de dados

O ponto mais forte do padrão XML é sua habilidade para o intercâmbio de dados, pois utilizando XML, uma empresa ou mesmo áreas diferentes de uma mesma empresa, podem receber dados com marcação XML através da web, sem precisar saber como o sistema de origem está organizado, ou seja, usando XML, cada lado envolvido em uma troca de dados pode utilizar uma ferramenta diferente para gerar e tratar documentos no padrão XML [Tidwell,2002a].

Quando houver um outro parceiro ou fornecedor desejando se conectar com uma organização, não haverá a necessidade de se escrever um novo código XML para trocar dados com este parceiro. A organização simplesmente solicitará que sejam observadas as

regras para a elaboração de um documento, definidas no seu DTD ou no seu esquema XML.

Como documentos XML podem ser estruturados para identificar cada parte de uma informação, é possível escrever uma aplicação que processe documentos XML sem a intervenção humana [Tidwell, 2002a].

Os recursos de marcação do padrão XML têm possibilitado o desenvolvimento de diversos padrões baseados em XML, notadamente aqueles voltados para o suporte ao intercâmbio eletrônico de dados, como os padrões utilizados para a implementação de serviços *WEB* (*Web Services*).

2.3. Serviços WEB (Web Services)

A simplicidade e a onipresença da Web, aliada a sua facilidade de uso, possibilitaram a criação de uma comunidade global e a sua utilização não somente para disponibilizar informações, mas também para disponibilizar serviços.

A implementação de serviços *Web* (WS – *Web Services*) está baseada nas facilidades de integração entre aplicações, proporcionadas pelos recursos de marcação do padrão XML aliados à ampla utilização do protocolo HTTP(*HyperText Transfer Protocol*) [Tidwell, 2002a].

Segundo Vinoski, serviços *Web* são aplicações que podem ser publicadas, localizadas e requisitadas via Internet, utilizando o protocolo HTTP (*HyperText Transfer Protocol*) e padrões baseados em XML, tais como o SOAP (*Simple Object Protocol)*, UDDI (*Universal Description Discover and Integration)* e a WSDL (*Web Service Description Language*) [Schmidt e Vinoski, 2002].

Serviços *Web* são também vistos como funções encapsuladas e fracamente acopladas, oferecidas em uma arquitetura orientada a serviço.

Encapsulada significa que a implementação da função nunca é vista do outro lado, ou seja, do lado cliente que utiliza a função, e fracamente acoplada significa que mudanças de implementação em uma função que disponibiliza um serviço, não requerem mudanças nas funções que efetuam chamadas a este serviço.

Os serviços *Web* podem executar funções que vão desde o processamento de uma simples requisição de serviço até complexos processos de negócios [Tidwell, 2002b].

Alguns exemplos de serviços que podem ser providos são: fornecer informações sobre o mercado de ações, reserva de passagens, informações geográficas a partir de um código postal, processamento de transações de cartão de crédito, entre outros.

A funcionalidade disponibilizada por um serviço *Web* pode também envolver um cenário mais complexo, conectando aplicações de várias organizações em uma mesma cadeia de fornecedores (*Supply chain*), assim como ser utilizado como solução para integração de aplicações empresariais (EAI – *Enterprise Application Integration*) tanto internas como externamente a uma empresa [Newcomer,2002,p.2].

2.3.1. Arquitetura básica de serviços Web

Conforme as definições do W3C [W3C, 2004c], a arquitetura básica de serviços *Web* é formada pelos seguintes elementos:

Registro (*Registry*)

O repositório ou registro de serviços é onde são publicados os serviços *Web*, possibilitando que uma empresa consumidora interessada em determinados serviços, possa efetuar uma pesquisa para verificar os serviços *Web* disponíveis. A organização das informações no repositório é efetuada através do padrão UDDI (*Universal Description, Discovery and Integration*).

Requisitante do serviço (cliente)

O consumidor de serviços é qualquer entidade que utilize um serviço *Web* disponibilizado por um provedor de serviços, podendo ser, por exemplo, um outro serviço.

Provedor do serviço

O provedor de serviços é a entidade que cria o serviço *Web*, disponibilizando-o para ser acessado por outras entidades que tenham interesse em utilizar o serviço.

O provedor de serviços publica as informações de seus serviços no registro utilizando o protocolo SOAP para encapsular essas informações e efetuar a ligação da aplicação disponibilizada com a aplicação UDDI do repositório.

Estas informações por sua vez são descritas utilizando-se a linguagem WSDL (*Web Services Description Language*), baseada no padrão XML.

2.3.2. Padrões utilizados com os serviços Web

Para que os serviços *Web* possam ser publicados, localizados e requisitados via Internet, é necessária a utilização de padrões, que possam ser entendidos por qualquer aplicação que deseje acessar os serviços.

A utilização de padrões, tais como o protocolo SOAP (*Simple Object Access Protocol*), o protocolo HTTP (HyperText Transfer Protocol), a UDDI (*Universal Discovery, Description and Integration*) e a WSDL (*Web Services Description Language*) visa principalmente garantir a interoperabilidade entre as aplicações e a transparência entre as diversas implementações de serviços na web, permitindo que estes sejam acessados independentemente da linguagem, fornecedor e plataforma operacional em que foram implementados.

Neste contexto, abordaremos o protocolo SOAP, que também é utilizado pelo padrão ebXML, objeto principal deste trabalho.

O protocolo SOAP (*Simple Object Acess Protocol*)

O protocolo SOAP *(Simple Object Access Protocol* - Protocolo Simples de Acesso a Objetos) é um protocolo baseado em XML, idealizado para possibilitar a troca de informações em um ambiente descentralizado e distribuído.

A especificação do SOAP versão 1.2 [W3C, 2004d] define um *framework* para a transmissão de mensagens entre sistemas distribuídos, e convenções para a chamada de procedimentos remotos (RPC – *Remote Procedure Call*).

O *framework* do SOAP não define a semântica das mensagens, mas suporta extensões que podem ser utilizadas por aplicações específicas, como por exemplo, por aplicações que utilizam o padrão ebXML (*e-business XML*), que define mensagens padronizadas entre parceiros de negócio.

Uma mensagem SOAP é especificada como um conjunto de informações XML e normalmente é transportada através do protocolo HTTP, tornando-se independente com relação a ambientes operacionais e a restrições impostas por dispositivos de rede.

A partir da versão 1.1 do SOAP, a especificação foi expandida, abrangendo outros protocolos como o *Simple Mail Transport Protocol* (SMTP) e o *File Transfer Protocol (FTP)*, que também podem ser utilizados como protocolos de transporte.

O formato de uma mensagem SOAP é constituído de um envelope obrigatório, um cabeçalho opcional e um elemento de corpo obrigatório (figura 1). O envelope é um documento XML, com o elemento "Envelope" como raiz, que contém os outros dois elementos (cabeçalho e corpo).

O corpo é o elemento do envelope que contém a mensagem a ser transportada, enquanto o cabeçalho, quando utilizado, contém informações que serão utilizadas para processamento intermediário por nós SOAP que estão entre o nó de origem e o nó de destino final da mensagem.

O cabeçalho, por exemplo, pode ser utilizado para providenciar uma assinatura digital para uma requisição contida dentro do corpo. Neste caso, um servidor de autenticação ou de autorização pode processar a entrada contida no cabeçalho, independentemente do corpo, visando validar esta assinatura [Clements, 2002].

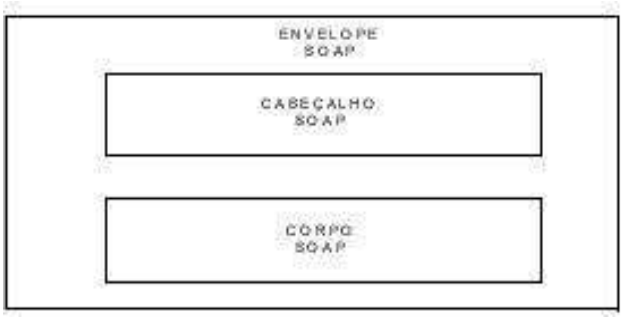

Figura 1: Envelope SOAP[W3C,2004]

A especificação SOAP[W3C, 2004d] descreve o seguinte exemplo de uma mensagem:

```
<env:Envelop exmlns:env=http://www.w3.org/2003/05/soap-envelope>
  <env :Header>
      <n:alertcontrol   xmlns:n=http://example.org/alertcontrol>
        <n:priority>1</n:priority>
```

```
    <n:expires>2001-06-22T14:00:00-05:00</n:expires>
  </n:alertcontrol>
  </env:Header>
  <env:Body>
    <m:alert xmlns:m="http://example.org/alert">
    <m:msg>Pick up Mary at school at 2pm</m:msg>
   </m:alert>
  </env:Body>
</env:Envelope>
```

No exemplo acima uma mensagem de notificação é expressa em SOAP. A primeira linha contém o elemento raiz Envelope e a indicação do *namespace* que contém o esquema XML com as especificações do Envelope do SOAP.

O cabeçalho (*Header*) contém um elemento criado pelo autor do documento, denominado *alertcontrol* com a indicação do seu respectivo esquema XML. Este elemento contém os elementos *priority* e *expires* e as informações contidas nestes elementos poderão ser utilizadas por um nó SOAP intermediário, bem como pelo último nó de destino da mensagem.

Neste exemplo, um nó intermediário deve priorizar a entrega da mensagem, baseada na prioridade e informação de expiração contida nestes elementos do cabeçalho.

O corpo contém a mensagem a ser entregue, neste exemplo, uma mensagem de alerta, definida no elemento indicado pela *tag* "<m:alert>" e contida no elemento filho <m:msg>.

Estes elementos apresentam o prefixo "m", indicando que estão associados ao namespace xmlns:m="http://example.org/alert".

O funcionamento do protocolo SOAP

O SOAP consiste basicamente em mensagens enviadas entre um remetente e um destinatário, que são capazes de processar mensagens SOAP. Estas mensagens podem transmitir dados, conforme o exemplo anterior, ou podem ser utilizadas para transmitirem chamadas a métodos remotos, encapsulando chamadas RPC (*Remote Procedure Call*).

No contexto de serviços Web, o SOAP deve ser implementado tanto do lado cliente quanto do lado servidor.

Um cliente SOAP é um programa que cria um documento XML contendo a informação necessária para invocar remotamente um método, dentro de um sistema distribuído, podendo estar em uma aplicação *desktop* ou mesmo em um servidor Web [Clement, 2002].

Um servidor SOAP é um programa que aguarda a chegada de mensagens SOAP e atua como um distribuidor e interpretador destas mensagens, assegurando que o documento ou a invocação recebida possa ser convertida para a linguagem que o objeto receptor possa entender, efetuando o papel inverso no envio da resposta para a aplicação que enviou a mensagem.

O SOAP utiliza o modelo de mensagem de solicitação/resposta do HTTP para o encaminhamento de mensagens (figura 2).

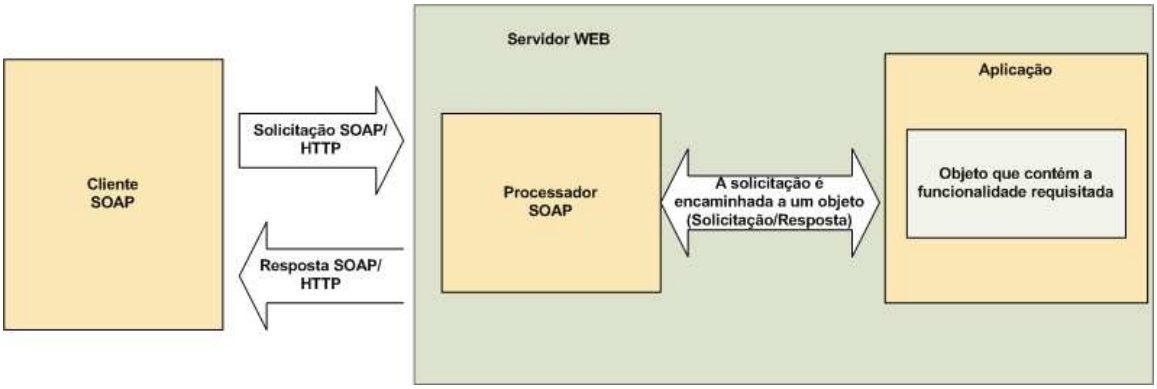

Figura 02: Arquitetura SOAP [Hendricks et.al.,2002,p41]

Uma aplicação cliente encapsula a chamada de um método remoto no formato XML/SOAP. Esta mensagem SOAP é então vinculada ao protocolo HTTP, utilizando uma mensagem de solicitação HTTP para transmissão ao servidor.

No servidor, a requisição é validada e processada por um processador SOAP, que efetua a leitura do nome do método e os parâmetros informados, passando-os para a respectiva aplicação que contém este método, para processamento da solicitação.

O processador SOAP é uma aplicação executada no servidor que efetua a análise da solicitação XML, faz a chamada ao procedimento indicado e posteriormente encapsula o resultado em XML para enviar como resposta, ou seja, atua como uma ponte entre o cliente SOAP e a aplicação que contém o serviço solicitado.

O SOAP possibilita que sistemas remotos se comuniquem diretamente, mas não define um mecanismo para a descrição e localização de objetos em sistemas distribuídos. Esses recursos são definidos respectivamente pelas especificações WSDL (*Web Service Description Language*) e UDDI (*Universal Description Discover and Integration*).

Uma estrutura similar à estrutura básica dos serviços Web é a estrutura proposta pelo padrão ebXML, embora estas tecnologias estejam focadas em segmentos diferentes com relação aos serviços eletrônicos.

Segundo [Deitel *et. al*.,2003, p.39], enquanto as especificações que suportam os serviços Web estão mais voltadas para a integração e disponibilização de serviços de uma forma geral, com ampla forma de uso, o padrão ebXML está focado na integração de processos de negócio entre empresas (B2B), disponibilizando mecanismos adicionais, específicos para a definição deste tipo de colaboração comercial.

2.4. ebXML: Um padrão XML para o comércio eletrônico entre empresas

Em 1999, o UN/CEFACT (United Nations Centre for Trade Facilitation and Electronic Business) e o OASIS (Organization for the Advancement of Structured Information Standards) iniciaram um projeto para consolidar e padronizar as diversas especificações XML existentes, voltadas para a facilitação do comércio eletrônico.

O resultado deste esforço foi a aprovação, em Maio de 2001, de um conjunto modular de especificações, abrangendo diversos aspectos de um processo de interação eletrônica entre empresas, denominado de *Electronic Business XML*, ou simplesmente *ebXML*.

Em março de 2004 as especificações ebXML foram padronizadas pela ISO (*International Standard Organization*) através da norma ISO 15000, abrangendo os principais componentes do padrão ebXML.

O padrão ebXML combina a experiência destas organizações com o comércio eletrônico através do antigo padrão EDI - *Electronic Data Interchange*, e a flexibilidade do padrão XML, para possibilitar a construção de uma infra-estrutura uniforme de comércio eletrônico entre empresas.

2.4.1. O Intercâmbio Eletrônico de Dados - EDI

A troca de informações comerciais entre empresas através da forma eletrônica [Chiu, 2002, p.23], foi impulsionada na década de 70, quando agências governamentais e grandes empresas sentiram a necessidade de melhorar a qualidade dos serviços e reduzir o esforço com atividades manuais na interação com os seus fornecedores.

Para viabilizar esta troca de informações entre empresas, algumas vezes envolvendo também transações comerciais, diversas organizações representativas de segmentos comerciais específicos, criaram padrões para serem utilizados pelas empresas pertencentes a sua área de atuação, visando efetuar este intercâmbio eletrônico de documentos.

A partir da utilização destes padrões, diversas redes privadas foram criadas, geralmente por grandes empresas, para suportar a troca de informações comerciais com seus parceiros de negócio.

Este processo de intercâmbio eletrônico de documentos, denominado EDI (*Electronic Data Interchange* - Intercâmbio Eletrônico de Documentos*)* consiste na utilização de sistemas específicos visando à integração de parceiros comerciais através de uma rede privada, para conduzir transações baseadas na troca de documentos com formatos padronizados, que possam ser reconhecidos e interpretados pelos parceiros conectados a esta rede [Deitel et al., 2003, p.6].

Negócios eletrônicos baseados em EDI são normalmente centralizados em uma empresa dominante, geralmente aquela que possui maior poder de negócio, que impõe uma forma de integração proprietária para os seus parceiros de negócio [ebXMLTA, 2002], ou centralizados em um provedor de serviço, que atua como intermediário no processo, interligando todos os parceiros através de seu sistema EDI.

Um sistema de EDI possui recursos para transferir, armazenar e gerenciar os documentos eletrônicos trocados entre as empresas [Deitel *et al*., 2003, p.6].

Entretanto, devido ao fato de sistemas EDI envolverem formatos e softwares específicos que não se comunicavam entre si, basicamente somente grandes organizações foram capazes de suportar os custos de implementação e manutenção da infra-estrutura necessária para participar de um processo EDI, pois dependendo das necessidades de negócio, uma mesma empresa pode ter que adquirir mais de uma solução de EDI para poder se comunicar com diversos parceiros.

Atualmente os principais padrões EDI são o EDIFACT, padronizado pela UN/CEFACT, e o padrão ANSI X12, padronizado pela *American National Standards Institute* (ANSI), mas diversas organizações da indústria utilizam somente partes específicas destes padrões, para complementar padrões próprios, desenvolvidos especificamente para o segmento comercial em que atuam [Chiu, 2002, p.24].

Com o desenvolvimento do *Extensible Markup Language* (XML), um padrão aberto de formatação de mensagens e a utilização da Internet como meio eletrônico de comunicação, criaram-se as condições técnicas favoráveis para o desenvolvimento de soluções de comércio eletrônico entre empresas, denominadas de B2B (*Business to Business*), com custos mais baixos e com abrangência mais significativa que em sistemas EDI.

Neste contexto, a UN/CEFACT (United Nations Centre for Trade Facilitation and Electronic Business) e o OASIS (Organization for the Advancement of Structured Information Standards) entidades responsáveis por padrões na área do EDI, elaboraram o padrão ebXML (e-business XML - Linguagem Extensível de Marcação de texto para Negócios Eletrônicos), visando a sua utilização não somente por segmentos verticais do mercado, mas como um padrão XML a ser utilizado por todos os segmentos.

2.4.2. Objetivos do ebXML

Conforme a definição do UN/CEFACT e OASIS, o objetivo principal das especificações ebXML é habilitar um mercado global eletrônico, onde empresas de qualquer tamanho e de qualquer localidade possam se encontrar e conduzir negócios umas com as outras, através da troca de mensagens baseadas em XML [ebXMLTA, 2002].

A proposta do ebXML é basicamente permitir o acesso de variados tipos de empresas ao comércio eletrônico, aproveitando a experiência de seus idealizadores com o EDI e disponibilizando publicamente um conjunto de especificações que visam compatibilizar e garantir a interoperabilidade necessária para a interação comercial eletrônica entre diferentes empresas em um ambiente aberto como a Internet, independentemente do tamanho destas empresas.

O ebXML espera suceder o EDI e além de ser patrocinado pelas mesmas organizações envolvidas com as especificações EDI, como o UN/CEFACT e o OASIS, conta também com a participação de várias empresas que atuam nesta área, tais como IBM, SUN, IONA, Fujitsu, entre outras, órgãos governamentais de diversos países e outros órgãos responsáveis pela elaboração de padrões, como o W3C, NIST - *National Institute of Standards and Technology*, entre outros.

2.4.3. Arquitetura ebXML - Componentes e terminologia

Para o estabelecimento de um processo de colaboração eletrônica entre parceiros comerciais, há a necessidade de se definir o processo de negócio que será executado nesta interação, assim como padronizar previamente os elementos que possibilitarão o estabelecimento deste relacionamento [Irani, 2001].

Um processo de negócio detalha as regras e responsabilidades que um parceiro de negócio deve cumprir em uma colaboração de negócio [Deitel *et. al*.,2003, p106].

Um processo de negócio é um conjunto de atividades, também chamadas de subprocessos, que são executadas em uma determinada seqüência, visando atingir um objetivo, como por

exemplo, em um processo de negócio de venda de um produto, existe a execução da atividade de identificação do cliente [Willaert,2001].

Os elementos de um processo de negócio compreendem, por exemplo, as transações de negócio que serão utilizadas no processo, a seqüência destas transações, assim como as especificações técnicas que deverão ser utilizadas, para garantir a interoperabilidade entre os sistemas dos parceiros envolvidos neste processo.

Para garantir a interoperabilidade entre os processos de negócio, tanto nos aspectos de infra-estrutura técnica, quanto nos aspectos de interação comercial, o padrão ebXML define uma arquitetura modular, com vários componentes, conforme especificado em sua arquitetura técnica [ebXMLTA, 2002]:

- O Registro (*Registry*), que executa o papel de repositório dos componentes ebXML e de diretório de aplicações ebXML. Na prática é um servidor que armazena uma variedade de dados necessários para a execução de processos ebXML, possibilitando que empresas possam desenvolver aplicações ebXML e descobrir novos parceiros, efetuando consultas (*queries)* ao Registro.
- O Modelo do Processo de Negócio (BP - *Business Process*), que são os meta modelos de informação e processos de negócio, expressados em UMM (*Unified Modeling Methodology*), necessários para auxiliar uma empresa a descrever seus processos de negócio. Um subconjunto deste modelo é o Esquema de Especificação do Processo de Negócio (BPSS – *Business Process Specification Schema*), que é um conjunto de vocabulários XML, indicando os elementos e atributos que podem ser utilizados para descrever um processo de negócio no padrão ebXML;
- O Perfil do Protocolo de Colaboração (CPP – *Colaboration Protocol Profile*), onde uma empresa especifica o seu perfil de negócio para o estabelecimento de um processo de colaboração e o Acordo do Protocolo de Colaboração (CPA – *Collaboration Protocol Agremment*), que especifica as regras que regerão um processo de colaboração;
- Os componentes centrais (*Core Components*) do ebXML, que são componentes formados por objetos comuns a vários processos de negócio, armazenados em uma biblioteca central (*Core Library*) no Registro e que podem ser utilizados por uma empresa para a elaboração do documento XML de seu processo de negócio (BP);
- O serviço de mensagens (ebMS– *e-business Message Service*), utilizado para a troca de mensagens e para garantir a interoperabilidade entre os parceiros;

A figura 3 mostra uma visão de alto nível de um ciclo de interação ebXML.

No passo 1, a empresa A consulta o Registro ebXML, analisando cenários e perfis de negócio existentes no Registro, além da biblioteca de Componentes Centrais de negócio (Core Components) e da biblioteca Central de Processos de Negocio (Core Process), baixando diagramas UML (*Unified Modeling Language*), esquemas e documentos XML,

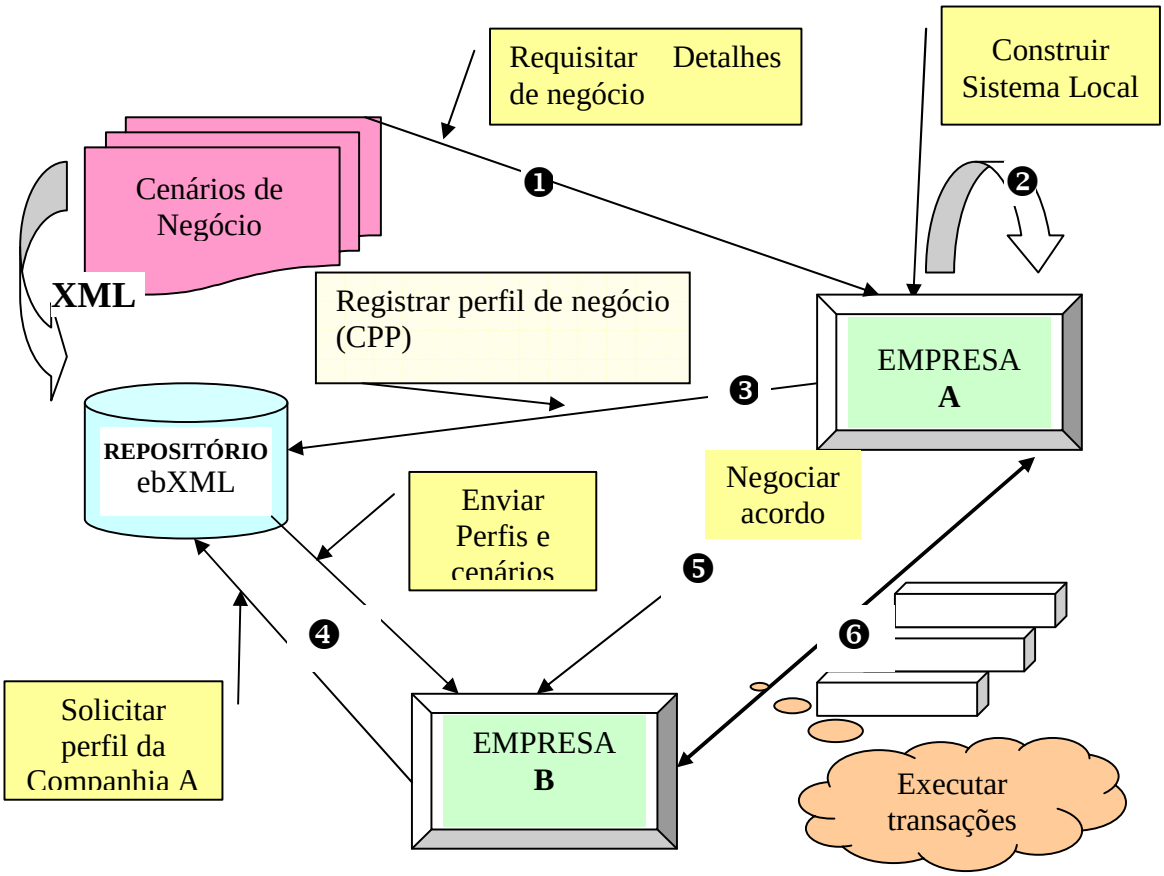

Figura 3 Visão geral de uma interação ebXML [ebXML TA,2002]

os quais permitirão determinar os requisitos para o desenvolvimento ou aquisição, de sua própria aplicação ebXML.

No passo 2, a empresa A constrói a sua aplicação ebXML e cria o documento de Perfil do Protocolo de Colaboração (CPP – *Colaboration Protocol Profile*), com as informações necessárias para que um possível parceiro possa verificar as capacidades técnicas e o processo de negócio que a empresa A suporta e está interessada em estabelecer.

No passo 3 efetua o registro do seu perfil de negócio (CPP) no Registro ebXML.

No passo 4, uma empresa B que está buscando no Registro os perfis de possíveis parceiros comerciais compatíveis com o seu processo de negocio, solicita e recebe o CPP da empresa A, iniciando os contatos e negociações (passo 5) baseado nas informações do CPP, para a elaboração de um acordo de colaboração (CPA).

Após o fechamento do acordo, são estabelecidas as transações de negócio entre as duas empresas (passo 6), envolvendo a troca de mensagens de negocio conforme o acordo de colaboração que foi negociado.

A seguir, abordaremos cada um dos componentes do padrão ebXML e como eles estão relacionados, conforme a especificação ebXML que define cada componente.

2.4.4. O Registro (*Registry*)

O ebXML possui um depositório de informações denominado Registro, conceitualmente similar a registros UDDI utilizados com os Serviços Web, mas com um escopo mais abrangente.

A especificação de Serviços do Registro ebXML [ebRS,2002], define as funcionalidades de repositório e de registro de informações, para formar uma base de dados que permita aos parceiros de negócio efetuar a busca e descoberta de novos parceiros, assim como compartilhar as informações necessárias ao estabelecimento de uma interação comercial utilizando o padrão ebXML.

Estas informações abrangem dados relativos às empresas participantes de uma comunidade ebXML, como por exemplo nome, informações de contato, o processo de negócio que suporta, informações técnicas referentes a seu perfil de negócio, tais como as mensagens suportadas, protocolos de transporte utilizados e recursos de segurança, que compõe o Perfil do Protocolo de Colaboração (CPP) de uma empresa.

A especificação do Modelo de Informação para um Registro ebXML [ebrim,2002], é uma outra especificação para este componente, que define como as informações devem ser organizadas e as classes, atributos e métodos que devem ser utilizados por desenvolvedores para a implementação de um Registro ebXML.

Essa especificação também define que o modelo de informação pode ser implementado em um Registro ebXML através de um esquema de banco de dados relacional, banco de dados orientado a objetos ou outro esquema físico.

Como pode ser observado na figura 04 a seguir, a arquitetura de um Registro ebXML é composta de um Registro, onde estão o índice de acesso e as referências ao conteúdo, assim como a interface de serviço, e de um Repositório, onde está armazenado o conteúdo apontado pelo Registro.

O Registro pode trocar mensagens com outro Registro ebXML, e neste caso, deve ser definido um conjunto de processos e mensagens a serem trocadas com o Registro Cliente [ebRS,2002], de forma a garantir a sincronização das informações.

As consultas ao Registro são efetuadas através de mensagens encaminhadas via serviço de mensagens ebXML (ebMS) ou de outro protocolo de mensagens XML, utilizando o modelo de mensagens do tipo Requisição/Resposta, destinadas à interface de serviço do Registro.

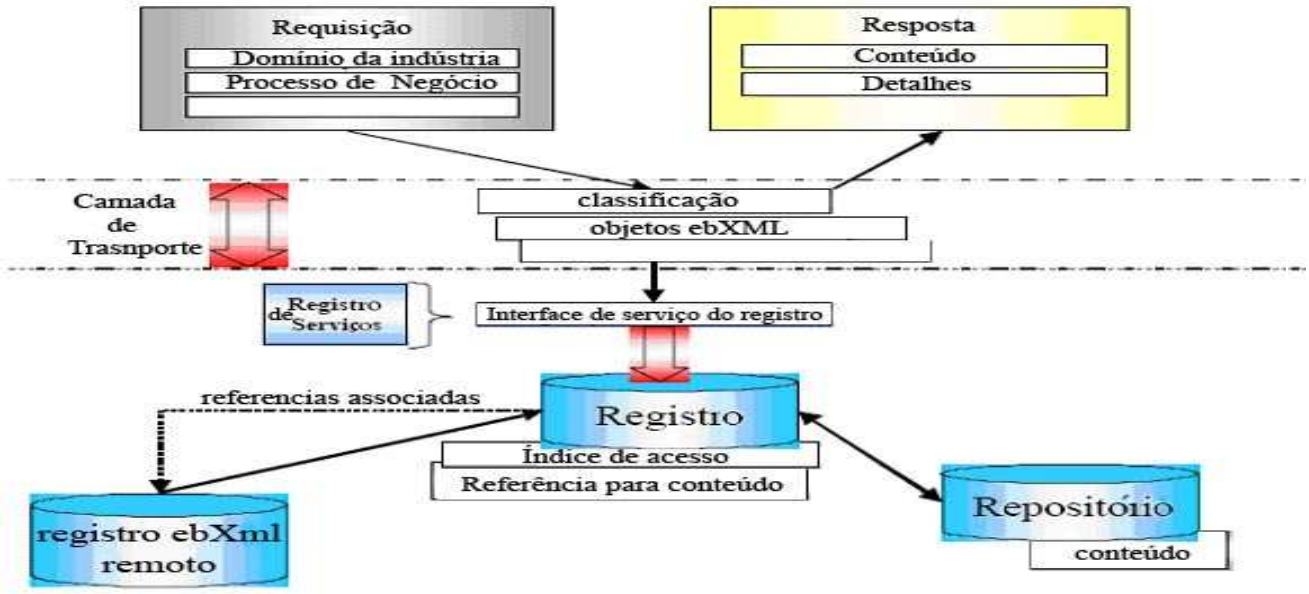

Figura 04: Arquitetura do Registro ebXML [ebXML TA,2002]

O acesso ao Registro é efetuado através desta interface de serviço, que deve ser independente do protocolo de transporte como o HTTP, SMTP ou FTP, por exemplo, e suportar a interoperabilidade com interfaces de serviço de outros Registros, como UDDI e Corba (*Commom Object Request Broker Architecture*) [ebXML TA,2002].

A interface do Registro ebXML serve como um mecanismo de acesso para aplicações e suporta também o acesso para interações humanas através de um *browser*, fornecendo serviços para facilitar o processo de descoberta e consulta ao conteúdo do Registro.

Estas consultas utilizam um esquema de classificação de informações definido dentro do Registro para organizar as informações [ebRS,2002], como por exemplo, um comprador pode consultar as empresas que estão classificadas como Industria Automotiva, que executam o papel de vendedor, suportem um processo de negocio especifico (*buySell*) e vendam rodas de alumínio.

O conteúdo do Registro abrange o armazenamento de diversos documentos relativos ao padrão ebXML, tais como o CPP, o CPA, modelos UML de processos de negócio, esquemas (BPSS e DTD) e documentos XML, a biblioteca de Processos de Negócios e a biblioteca de Componentes Centrais de negócio, assim como qualquer outro documento utilizado em uma interação comercial com o padrão ebXML.

O Registro gerencia e mantém estas informações como objetos dentro do Repositório, disponibilizando serviços para acesso, busca, registro, recuperação e atualização de todos estes documentos [ebRS,2002].

O Registro ebXML pode residir em um servidor Web de alguma empresa participante do processo de colaboração ou ser hospedado por um provedor de serviço de aplicações (ASP – *Application Service Provider*), podendo ser descentralizado, ou seja, ser implementado de forma distribuída em provedores regionais, por exemplo.

Todos os itens dentro de um Registro têm uma identificação única, denominada UID (*Unique Identifier*).

Este identificador pode ser implementado, por exemplo, através do uso de uma referência URI (*Uniform Resource Identifier*), a qual pode representar o nome ou endereço de um objeto ou recurso em uma rede através de uma URL, assegurando a identificação única e facilitando o processo de consulta ao Registro.

2.4.5. O Processo de Negócios (*ebBP – ebXML Business Process*)

A definição de processos de negócios no ebXML é efetuada de acordo com o Esquema de Especificação de Processos de Negócios denominado de BPSS (*Business Process Specification Schema*), definido na especificação [ebBP, 2006].

A meta principal do ebBP (ebXML *Business Process*) conforme definido na sua especificação, é padronizar a semântica do contexto de negócio, definir o processo de troca de dados e reduzir a ambigüidade na comunicação entre as partes envolvidas, com relação ao processo de negócio que será utilizado em uma colaboração.

No contexto do ebXML, um processo de negócio descreve como parceiros comerciais devem interagir para efetuar uma colaboração comercial, ou seja, as regras e atividades de negócio que deverão fazer parte desta interação.

A colaboração entre os parceiros de negócio é efetuada através de uma seqüência de transações de negócio e cada transação é expressa como uma troca eletrônica de documentos de negócio, como por exemplo, uma ordem de compra, seguindo a coreografia definida para a execução do processo de negocio, conforme ilustrado na figura 05.

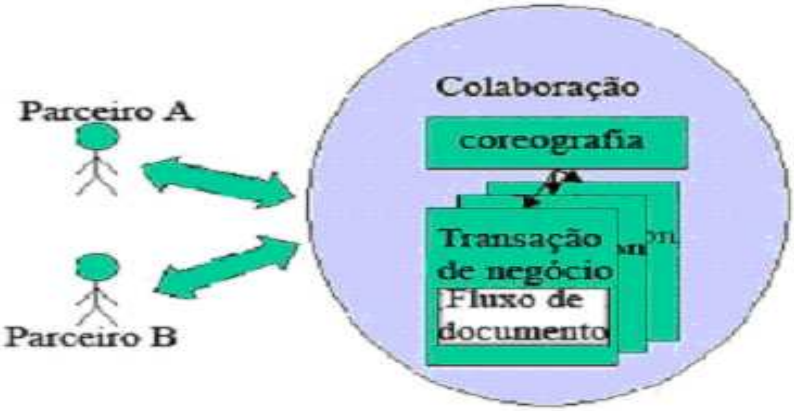

Figura 05: Semântica básica de uma colaboração de negócio [ebBP, 2006]

Uma colaboração de negócio pode ser efetuada por dois ou mais parceiros comerciais, que seguem um conjunto de regras de colaboração, executadas através de uma coreografia de mensagens de transações de negócio, onde cada transação consiste de um fluxo predefinido de documentos de negocio [ebBP,2006].

Este fluxo é constituído de uma atividade de requisição de negócio e de uma atividade de resposta de negócio que pode ser opcional, dependendo do tipo de transação (uma mensagem de notificação, por exemplo, não necessita de resposta).

A coreografia de uma colaboração comercial descreve a ordem e transições entre as transações de negócio, dentro de um processo de colaboração [ebBP,2006].

O documento de negócio utilizado nestas transações pode ser composto por objetos de informação de negocio, como nome da empresa, endereço para remessa, itens adquiridos, data da compra, entre outros, formando uma fatura ou uma nota fiscal, por exemplo.

Em um nível mais baixo de detalhe, processos de negócio podem ser compostos de Processos Centrais padronizados (*Core processes*) e objetos de informações de negócio podem ser compostos de Componentes Centrais padronizados (*Core Components*).

Estes elementos são obtidos de um conjunto comum de processos de negócio e componentes, disponibilizados em uma biblioteca Central no Registro, conforme definido na arquitetura do padrão ebXML [ebXMLTA,2002], comuns a vários tipos de indústria, ou seja, processos e componentes que são normalmente utilizados na composição de um processo de negócio, independentemente da área de negocio. Usuários podem estender estes elementos ou criar os seus próprios elementos.

A especificação técnica do padrão ebXML [ebXML TA, 2002] recomenda a utilização de uma metodologia de modelagem denominada UMM (UN/CEFACT Modeling Methodology), baseada no método RUP (*Rational Unified Process*) e na utilização da UML (*Unified Modeling Language*) para descrever um processo de negocio.

Uma especificação de processo de negócio também pode ser modelada utilizando-se a BPMN (*Business Process Management Notation*) ou um diagrama de atividades da UML, por exemplo [ebBP,2006].

Este processo de modelagem é efetuado de forma independente dos sistemas e plataformas a serem integrados e gera o Meta Modelo de Informação e Processos de Negócios, denominado de BPIM (*Business Process and Information Meta Model*).

No processo de modelagem também devem ser identificados os meta modelos de informação e os processos de negócio que podem ser candidatos para padronização como componentes reusáveis, ou seja, componentes que podem ser armazenados na biblioteca Central (indicado como Biblioteca de Negócios UMM na figura 06) para uso por todas as organizações que desejem efetuar suas definições de processos de negócio conforme o padrão ebXML.

Um subconjunto deste modelo é o Esquema de Especificação do Processo de Negócio (BPSS – *Business Process Specification Schema*), que define as regras para elaboração de um documento XML que descreva os processos de negócio de uma organização.

A UMM é uma metodologia para definir aspectos de negocio de colaborações B2B e o BPSS é tipicamente expresso em diagramas UMM e mapeado para um esquema XML ou um DTD, podendo ser considerado uma representação XML de um subconjunto do metamodelo UMM [Hofreiter e Huemer,2006].

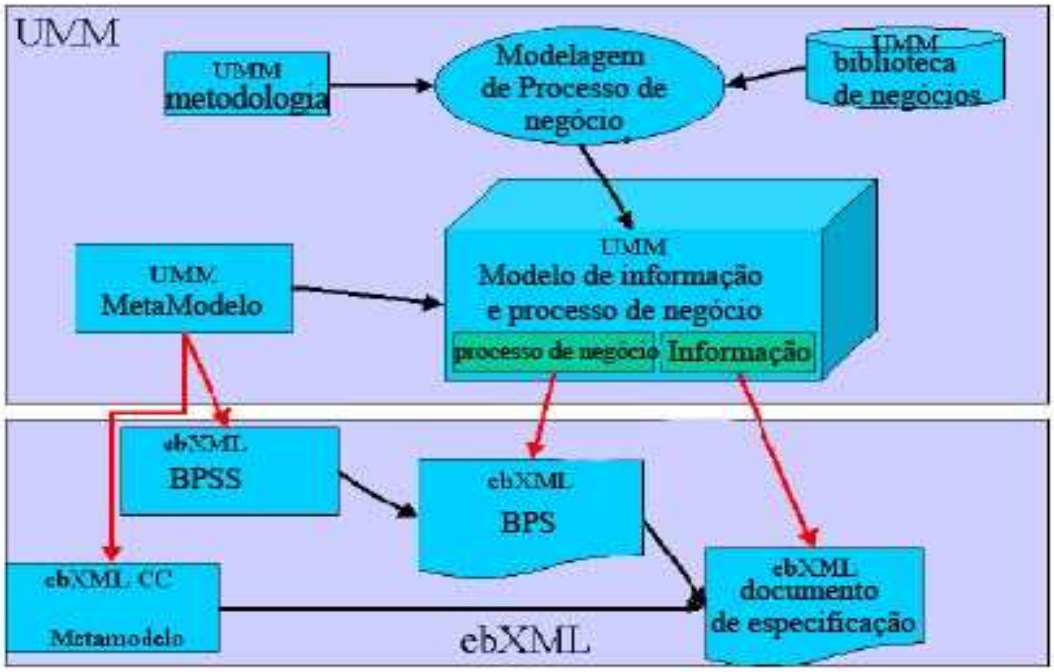

Figura 06: Relacionamento entre o Meta Modelo UMM e o BPSS[ebBP, 2006]

A figura 06 mostra a relação entre o processo de modelagem de negócio e a formação, a partir desta modelagem, do esquema XML (BPSS) que governa o documento de especificação do processo de negócio (ebBPS - ebXML *Business Process Specification*).

Este documento é composto por elementos e atributos especificados no esquema (BPSS) e pode incluir modelos de processos já existentes no Registro ou especificados durante o processo de modelagem.

O documento de especificação de processo de negócio (ebBPS) é uma instância do BPSS e faz referência a um ou mais documentos de especificação de negócio (*Business Document Specification*) que serão utilizados neste processo.

A definição do documento de negócio a ser utilizado em um processo de negócio é efetuada a partir dos componentes centrais já existentes no Registro e das informações de negócio modeladas especificamente para este processo.

As informações de negócio que compõem um documento de negócio, podem ser compostas por componentes reusáveis, obtidos a partir da biblioteca Central de Componentes *(Core Components)* de um Registro ebXML, como por exemplo, uma fatura, composta por componentes atômicos como data, valor e quantidade.

O documento de especificação do processo de negócio acrescenta o contexto no qual um documento de negócio será utilizado, como, por exemplo, como uma fatura será utilizada dentro de um processo de negócio de compra e venda.

Neste ponto, é necessário ressaltar que a especificação do documento de negócio é efetuada através de um esquema XML que especifica os elementos XML (*Core Components*) que deverão compor o documento.

Conforme pode ser observado na figura 06 anterior, o documento de especificação do processo de negócio (ebBPS) referencia a especificação do documento de negócio, indicando a sua utilização no processo.

Este mesmo mecanismo se aplica quando uma especificação de um processo de negócio reutiliza um processo de negócio já existente na biblioteca de processos de negócio do Registro (*Core Process*), ou seja, o ebBPS apenas referencia o documento do processo já existente, indicando a sua utilização pelo processo que está sendo desenvolvido.

O BPSS indica os elementos e atributos que poderão ser utilizados em um processo de negócio [ebBP,2006], ou seja, define um conjunto de vocabulários XML que podem ser utilizados para descrever as transações de negócio em uma colaboração eletrônica com a utilização do padrão ebXML.

Este esquema XML define os elementos e atributos que serão utilizados para descrever as regras, transações e identificação dos documentos de negócio que serão utilizados em um processo de negócio, o fluxo destes documentos e aspectos de segurança.

Estes elementos serão utilizados no ebBPS, o documento de instância do BPSS, onde serão definidos os conteúdos destes elementos.

Um documento de processo de negócio (ebBPS) pode ser caracterizado por conter os seguintes tipos de informação [eBP, 2006]:

- A coreografia para a troca de documentos, como a seqüência de mensagens trocadas entre os parceiros comerciais.
- As referências para um documento de negócio e para o BPSS como DTDs ou esquemas XML, que estabelecem a estrutura para os dados de negócio.
- Definição das regras que cada participante deverá executar dentro do processo de negócio.

Um documento de processo de negócio pode ser transportado entre Registros como uma mensagem ebXML e para isto possui um identificador único, para poder ser localizado e recuperado através de uma solicitação (*query)* ao Registro.

A figura 07 abaixo mostra um fragmento de um documento de processo de negócio como exemplo.

```
<ProcessSpecification>
    xmlns="http://docs.oasis-open.org/ebxml-bp/ebbp-2.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    name="PurchasingCluster" nameID="PC23"
        ...
    <BusinessDocument name="Invoice" nameID="bd-invoice">
      <Specification type="schema"
         location=" http://purchasingcluster.com/InvoiceResponse.xsd"
          name="Invoice" nameID="invoice32”/>
    </BusinessDocument>
```

```
  <BusinessDocument name="InvoiceResponse"
   nameID="bd-invoiceResponse">
     <Specification type="schema"
       location="http://purchasingcluster.com/InvoiceResponse.xsd"
         name="InvoiceResponse" nameID="invoice33"/>
  </BusinessDocument>

  <DataExchange name="Data:Invoice" nameID="data-invoice">
     <RequestingRole name="DIinitiator" nameID="DIinitiator1"/>
     <RespondingRole name="DIresponder" nameID="DIresponder1"/>
     <RequestingBusinessActivity name="ReqBA:SendInvoice"
       nameID="debareq-invoice"
         timeToAcknowledgeReceipt="PT6H"
       timeToAcknowledgeAcceptance="PT12H">
        <DocumentEnvelope name="DE:ProcessInvoice"
       nameID="data-de-invoice" businessDocumentRef="bd-invoice"/>
     </RequestingBusinessActivity>
     <RespondingBusinessActivity name="ResBA:ReceiveInvoice"
       nameID="debares-invoice">
        <DocumentEnvelope name="DE:ProcessInvoiceResponse"
       nameID="data-de-invoiceResponse"
       businessDocumentRef="bd-invoiceResponse"/>
     </RespondingBusinessActivity>
  </DataExchange>
       …
</ProcessSpecification>
```

Figura 07: Fragmento de um documento de especificação de processo de negócio[ebBP,2006]

O documento inicia com o elemento raiz *<ProcessSpecification>*, a declaração dos *namespaces* utilizados no documento e o nome (*PurchasingCluster*) e identificação desta especificação de processo de negócio.

Os documentos de negócio a serem utilizados neste processo são *Invoice* e *InvoiceResponder*, declarados no atributo *name* e identificados pelo atributo *nameId* do elemento *<BusinessDocument>*, que contém o elementos de localização dos esquemas que especificam estes documentos, assim como atributos para a identificação destes esquemas.

O elemento *<DataExchange>* permite a definição de um conjunto de transações de negócio na forma de Requisição/Resposta (*Resquest/Respondig*) para a troca de dados entre as partes, assim como a definição dos elementos para a identificação do requisitante (*RequestingRole*) e de quem deve responder (*RespondingRole*), para a execução deste processo de negócio.

Neste elemento também são definidos atributos para estabelecer os limites de tempo (no exemplo em horas) para receber uma resposta de confirmação de recebimento à mensagem de *RequestBusinessActivity* encaminhada (*timeToAcknowledgeReceipt*) , assim como uma resposta de aceite para a transação indicada na mensagem (*timeToAcknowledgeAcceptance*).

O documento de especificação do processo de negócio é referenciado por um CPP e por um CPA, conforme pode ser observado na figura 08, indicando qual processo uma empresa está disposta a executar em uma colaboração de negócios.

No CPP/CPA é especificado qual o papel (vendedor ou comprador, por exemplo) a empresa vai executar neste processo.

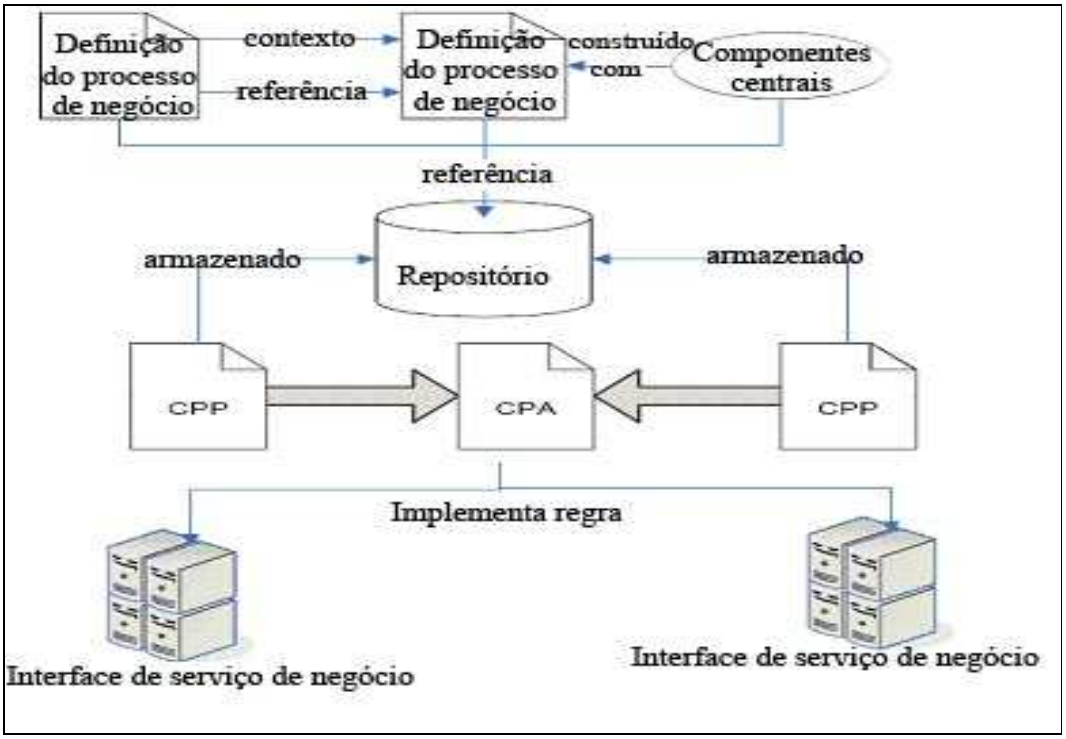

Figura 08: Definição do ebBP e relação com CPP e CPA [ebBP,2006]

Através do CPA, as regras de negócio especificadas no documento de processo de negócio serão utilizadas para configurar a interface do serviço de negócio (BSI - *Business Services Interface*) na aplicação de cada parceiro, de acordo com o papel que exercerá na execução deste processo.

2.4.6. Os Protocolos de Colaboração: CPP (*Collaboration Protocol Profile*) e CPA (*Collaboration Protocol Agreement*)

O conceito de protocolo de colaboração foi criado pela IBM, a partir da especificação de uma linguagem de marcação para o estabelecimento de acordos de colaboração entre parceiros comerciais denominada TPA (*Trading Partner Agreement*), que permitia a definição das regras de interação entre empresas engajadas em um processo de colaboração [Chiu, 2002, p.69].

No padrão ebXML o documento de Perfil do Protocolo de Colaboração – CPP, permite aos parceiros comerciais descreverem o seu processo de negócio e especificar a interface de seu serviço de negócio, de forma que estas informações possam ser entendidas e acessadas por outros parceiros comerciais.

A especificação do protocolo de colaboração [ebXML CPPA, 2002] descreve o CPP como um documento XML, governado por um DTD ou esquema, que especifica os detalhes

técnicos e informações relativas à capacidade de uma organização para suportar um determinado processo de negócio em uma colaboração eletrônica.

No CPP são especificadas informações de localização e de contato com a empresa, classificação da indústria, mensagens e protocolos de transporte utilizados pela empresa, restrições de segurança, bem como as interfaces utilizadas para viabilizar a troca de documentos de negócio, na execução de um determinado processo de negócio.

O CPP referencia um ou mais processos de negócio suportados por uma empresa, assim como as regras que esta empresa suporta em cada processo de negócio.

Um exemplo de regra é a definição se a empresa exerce o papel de comprador ou de vendedor em um processo de negócio de compra. O CPP referencia as regras que estão definidas no documento de especificação de processo de negócio (BPS) da empresa.

O Protocolo do Acordo de Colaboração - CPA, de acordo com a sua especificação, é um documento XML governado por um DTD ou um esquema, que rege a troca de dados entre parceiros de negócio, em um processo de colaboração ebXML [ebXML CPPA, 2002].

O CPA representa a intersecção dos CPP dos parceiros de negócio envolvidos em uma interação comercial, resultado do acordo mútuo de colaboração efetuado entre estes parceiros, após um processo de negociação.

As regras relativas às especificações técnicas que deverão ser seguidas pelos parceiros envolvidos no processo de colaboração devem estar descritas no CPA, como por exemplo, protocolos de transporte e mensagens de negócio a serem utilizados, recursos de segurança (criptografia e autenticação) e a definição do processo de negócio a ser executado entre as partes.

Neste contexto, o CPA pode ser utilizado por uma aplicação para configurar os detalhes técnicos que serão utilizados no processo de negócio a ser conduzido pelos parceiros comerciais, ajustando a interface do serviço de negócio e do serviço de mensagem de cada parceiro, de acordo com o conjunto de parâmetros especificados no CPA.

O processo de negociação que resulta no CPA assim como a sua elaboração, são efetuados manualmente, sendo no entanto passíveis de automatização [ebXML CPPA, 2002].

Uma organização interessada em participar de um processo de colaboração comercial utilizando o padrão ebXML deve gerar o documento CPP descrevendo o seu perfil de negócio, e publicar este documento em um Registro ebXML, para que possíveis parceiros de negócio possam consultá-lo.

Os documentos CPP e o CPA formam a base de desenvolvimento deste trabalho e serão abordados mais detalhadamente no próximo capitulo.

2.4.7. O Serviço de Mensagens ebXML (ebMS – *e-business Message Service*)

O processo de troca de mensagens entre aplicações ebXML deve ser confiável e padronizado [ebXML Requirements Specification, 2002] e para disponibilizar estas características em um ambiente descentralizado e distribuído, foi definido o Serviço de Mensagens ebXML (ebMS).

O Serviço de Mensagens do ebXML (ebMS), providencia um modo padrão para troca de mensagens eletrônicas entre parceiros comerciais e segundo a sua especificação [ebMS, 2002], é independente do protocolo de transporte, sendo normalmente utilizado com o protocolo HTTP.

A estrutura de uma mensagem ebXML é formada por duas partes: uma parte de cabeçalho (*header)*, necessário para o roteamento e entrega da mensagem a uma aplicação específica e uma parte referente à carga útil da mensagem (*payload*), onde é colocada a informação de negócio a ser transmitida [ebMS, 2002].

As informações a serem transportadas não precisam necessariamente ser baseada em XML, podendo ser mensagens formatadas em padrões tradicionais de EDI, como o EDIFACT e o ANSI X12, por exemplo, ou qualquer outro formato digital, como arquivos binários de imagens, tabelas de banco de dados, etc., pois o serviço de mensagens não coloca qualquer restrição ao conteúdo das mensagens.

O ebMS pode ser conceitualmente dividido em três partes:

1 – Uma interface de serviço abstrata, que inclui as funcionalidades de enviar, receber, notificar e solicitar o estado de uma mensagem. Um método *send* envia uma mensagem usando o valor do cabeçalho da mensagem como parâmetro, um método *receive* recebe a mensagem, *notify* notifica eventos e *inquire* questiona o estado de uma troca de mensagem específica.

2 – A camada do serviço de mensagem, onde é efetuada a verificação das regras do acordo de colaboração, usando para isto o Acordo do Protocolo de Colaboração (CPA – *Collaboration Protocol Agreement*) que define os termos de comunicação mutuamente acordados entre os parceiros comerciais envolvidos. Qualquer violação destas regras resulta em uma condição de erro, que será reportado à aplicação ebXML através da interface do serviço de mensagem.

3 – A camada de mapeamento para os protocolos de transporte, tais como o HTTP e o SMTP.

Com relação a transmissão de mensagens, o serviço de mensagens suporta trocas de mensagens do tipo notificação (em um sentido) e do tipo solicitação/resposta, no modo síncrono e assíncrono.

A figura 09 a seguir apresenta um arranjo lógico de módulos funcionais existentes dentro de uma possível implementação do ebMS, indicando o seu inter-relacionamento e dependências.

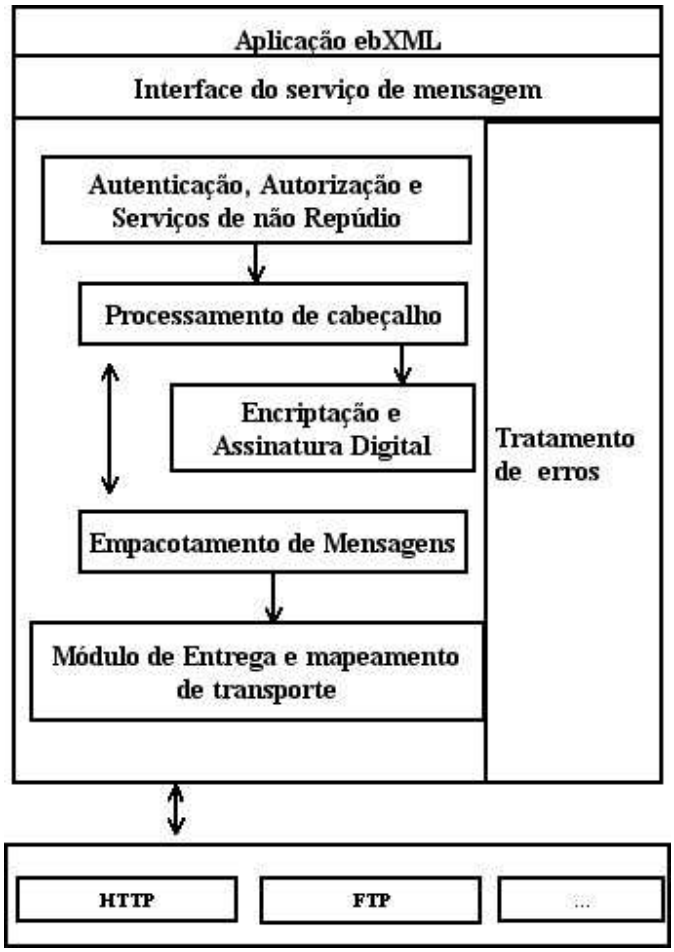

Figura 09: Módulos funcionais do ebMS [ebMS,2002]

Conforme pode ser observado na figura 09, a Interface do Serviço de Mensagem faz a intermediação do serviço de mensagem com a aplicação ebXML interna da organização, onde estão as regras de negócio.

A arquitetura do serviço de mensagem especifica os módulos de Autenticação, Autorização de acesso e Serviço de não repudio de mensagens, para verificação da segurança de acesso e identificação da mensagem.

O módulo de processamento do cabeçalho da mensagem (*Header Processing*) verifica o conteúdo do cabeçalho no recebimento de mensagens e, na transmissão, inclui o cabeçalho nas mensagens.

Este módulo é também responsável pelo roteamento da mensagem para a aplicação local ou para um destinatário remoto e deve ser configurado conforme os parâmetros descritos no acordo de colaboração (CPA) efetuado pelas partes.

Quando uma colaboração de negócio exige que a mensagem seja criptografada e assinada digitalmente, o módulo de processamento de criptografia e assinatura digital é acionado.

Caso a mensagem não necessite destes recursos de segurança, o módulo de processamento de cabeçalho (*Header Processing*) interage diretamente com o módulo de empacotamento das mensagens (*Message Packaging*), que é responsável por envelopar e extrair o conteúdo das mensagens (*payload*), utilizando o protocolo SOAP.

O tratamento de erros é efetuado pelo módulo *Error Handling*, que reporta os erros detectados durante o processamento da mensagem em qualquer um dos módulos do serviço de mensagem, inclusive notificando erros para ebMS do parceiro.

O módulo de entrega e mapeamento dos protocolos de transporte (*Delivery Module and Transport Mapping and Binding*), define as ligações para os protocolos HTTP, FTP e SMTP, a serem utilizados no transporte das mensagens, suportanto, no entanto, a configuração de ligação para outros tipos de protocolo.

O serviço de mensagem pode ser utilizado para a comunicação entre componentes ebXML distribuídos, incluindo o mecanismo do Registro e aplicações compatíveis de usuarios.

O serviço de mensagem não coloca qualquer restrição ao conteúdo das mensagens, suportanto os modos de troca de mensagem simples (em uma única direção) e requisição/resposta, ambos de forma síncrona ou assíncrona.

2.4.8 O ebMS e o SOAP

O serviço de mensagens ebXML utiliza o protocolo SOAP na troca de mensagens e para isto, especifica camadas de extensão sobre a especificação do SOAP e sobre a especificação do SOAP que suporta documentos anexados (*SOAP whit Attachments*) [ebMS, 2002], já que estas especificações não provêem serviços de segurança e de confiabilidade no transporte das mensagens.

Estas extensões são definidas dentro do elemento *header* e dentro do elemento *body* do envelope SOAP, onde são inseridos os elementos que contêm o cabeçalho e o corpo da mensagem ebXML, referenciados através de um conjunto de *namespaces* qualificados.

A figura abaixo mostra um exemplo destas extensões.

```
<SOAP:Envelope xmlns:SOAP="http://schemas.xmlsoap.org/soap/envelope/"
               xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
               xsi:schemaLocation="http://schemas.xmlsoap.org/soap/envelope/
                           http://www.oasis-open.org/committees/ebxml-msg/schema/envelope.xsd">
  <SOAP:Header
      xmlns:eb="http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd"
      xsi:schemaLocation="http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd
                          http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd">
    <eb:MessageHeader ...>
      ...
    </eb:MessageHeader>
  </SOAP:Header>
  <SOAP:Body
      xmlns:eb="http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd"
      xsi:schemaLocation="http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd
                          http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd">
    <eb:Manifest eb:version="2.0">
      ...
    </eb:Manifest>
  </SOAP:Body>
</SOAP:Envelope>
```

Figura 10: Extensões do envelope SOAP para inserção de elementos ebXML[ebMS,2002]

No exemplo mostrado na figura 10, o cabeçalho ebXML é definido dentro do elemento <eb:MessageHeader> e o corpo da mensagem no elemento <eb:Manisfest>.

A especificação [ebMS, 2002] no entanto preconiza que estas extensões no envelope SOAP dependem dos itens especificados no acordo de colaboração e não estão necessariamente sempre presentes em uma comunicação.

O ebMS adiciona ao SOAP características de segurança e não repúdio de mensagens, itens inexistentes na especificação do SOAP.

2.4.9 Os Componentes Centrais (*Core Components*)

Conforme o documento de especificação [ebXML Core Component, 2001], os Componentes Centrais são definidos como itens comuns a diversos tipos de negócios, disponibilizados no Registro para acesso compartilhado, visando promover o seu reuso.

As empresas que desejem desenvolver uma aplicação ebXML podem recuperar estes componentes de um Registro ebXML, e utilizá-los na elaboração de documentos de descrição de seus processos de negócios. Um componente representa informações relativas a um conceito de negócio do mundo real, representando dados significativos que formam um conjunto mínimo de informação de negócio.

Alguns exemplos de Componentes Centrais são as partes de uma ordem de compra como “Data do Pedido”, “Valor da Venda” e “Quantidade Total”, que podem ser utilizados em processos de negócio de diferentes domínios.

Estes componentes podem ser comparados a blocos de formato padronizado, cada qual com um identificador único, que podem ser combinados para formar componentes mais complexos, já que um componente pode conter outros componentes.

Um Componente Central é expresso em XML e deve ser armazenado e recuperado de uma Biblioteca Central no Registro ebXML.

O padrão ebXML define um conjunto inicial de Componentes Centrais que podem ser estendidos, conforme as necessidades na definição de um processo de negócio, mas o usuário pode também definir novos componentes, que poderão ser integrados posteriormente à Biblioteca Central, para reuso por ele próprio ou por outros usuários.

A especificação [ebXML Core Component, 2001] relaciona vários documentos que descrevem como podem ser desenvolvidos e utilizados os Componentes Centrais para a montagem de processos de negócios.

2.5 O funcionamento de uma interação ebXML

A interação entre empresas através da utilização de aplicações ebXML pode ser dividida em três fases básicas, de acordo com a especificação da arquitetura técnica do ebXML[ebXML T.A., 2001]:

- Fase de Implementação
- Fase de Descoberta e Negociação
- Fase de Transações

Fase de implementação

A fase de implementação abrange os procedimentos que uma empresa deve executar para o desenvolvimento de uma aplicação ebXML.

O primeiro passo é acessar um Registro ebXML para baixar e analisar o conjunto de especificações ebXML e subseqüentemente, baixar a biblioteca de Componentes Centrais, Processos de Negócios e os cenários de negócio (casos de uso e diagramas UML), que também estão armazenados no Registro e que poderão servir de modelo para a descrição do processo de negócio da empresa.

A figura 11 abaixo mostra uma interação básica que ocorre nesta fase, entre uma empresa A e a parte do repositório de um Registro ebXML.

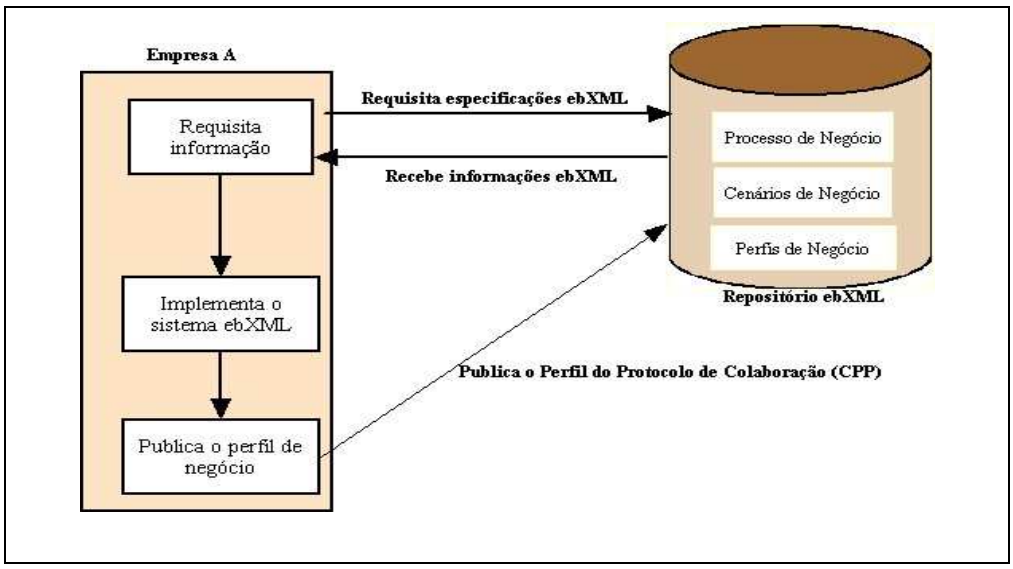

Figura 11: Fluxo relativo à fase de implementação ebXML[ebXMLTA,2002]

O repositório ebXML contém diversos processos e cenários de negocio definidos pela indústria, que são comumente utilizados em transações comerciais e que poderão ser utilizados pela empresa que está desenvolvendo uma aplicação ebXML, independentemente do segmento comercial em que atua.

Esta aplicação ebXML pode abranger o desenvolvimento do Modelo do Processo de Negócio em UML e XML, o serviço de mensagens – ebMs, a interface do serviço de negócio ebXML, que será utilizada na comunicação do ebMS com a aplicação de negócio e com o Registro, o perfil de negócio – CPP e se desejar, pode elaborar uma proposta inicial de acordo de colaboração – CPA.

As organizações podem escolher entre utilizar os modelos e processos existentes ou efetuar extensões nesses processos de negócio, adicionando seus próprios cenários de negócios (contextos específicos relativos a seus processos de negócios) ou ainda, desenvolver o seu próprio modelo de processo de negócio.

O repositório também contém os perfis de negócio (CPP) e propostas de acordos de colaboração (CPA) registrados por outras empresas que tem interesse em executar transações ebXML com parceiros comerciais através desse registro.

No lado da empresa A, conforme pode ser observado na figura 11, esta fase de implementação consiste de três passos:

1- Requisição da informação (especificações ebXML de Processos de Negocio e Cenários de Negócio),

2- Análise das especificações e verificação de quais processos de negocio a organização deseja implementar, verificando o que ela necessita para desenvolver ou adquirir um sistema ebXML.

3- Quando o sistema ebXML é construído, a empresa A está pronta para estabelecer negócios com outras organizações. Para isto deve gerar o seu perfil de negócio e publicar o Protocolo de Perfil de Colaboração – CPP, transmitindo-o para o repositório ebXML através do serviço de mensagens.

Quando registrado e publicado no Registro ebXML, o CPP da empresa possibilitará que outras empresas possam acessá-lo e verificar as capacidades de negócio da empresa A. A qualquer tempo a empresa A pode acessar o seu próprio perfil, revê-lo e efetuar as alterações que julgar necessárias.

A empresa A registra e publica também, além do CPP, o seu documento de especificação de processo de negócio (BPS) e o respectivo esquema XML (BPSS), relativo ao processo de negócio que está disposta a executar com outros parceiros.

A publicação de uma proposta inicial de CPA no Registro não é obrigatória, mas a empresa pode efetuar este procedimento para agilizar a fase posterior de negociação e elaboração do acordo de colaboração com os parceiros comerciais interessados em estabelecer uma interação com o seu processo de negócio.

Fase de descoberta da informação sobre parceiros e negociação

Nesta fase, um parceiro comercial que tenha implementado uma aplicação ebXML pode iniciar um processo de descoberta e recuperação de informações de outros parceiros no Registro ebXML e posteriormente negociar um acordo de colaboração comercial com estes parceiros.

A figura 12 exemplifica este processo. A empresa A efetua uma busca pesquisando as informações de negócio de possíveis parceiros, requisitando ao Registro os perfis de negócio (CPP) com os quais ela poderá estabelecer um processo de colaboração.

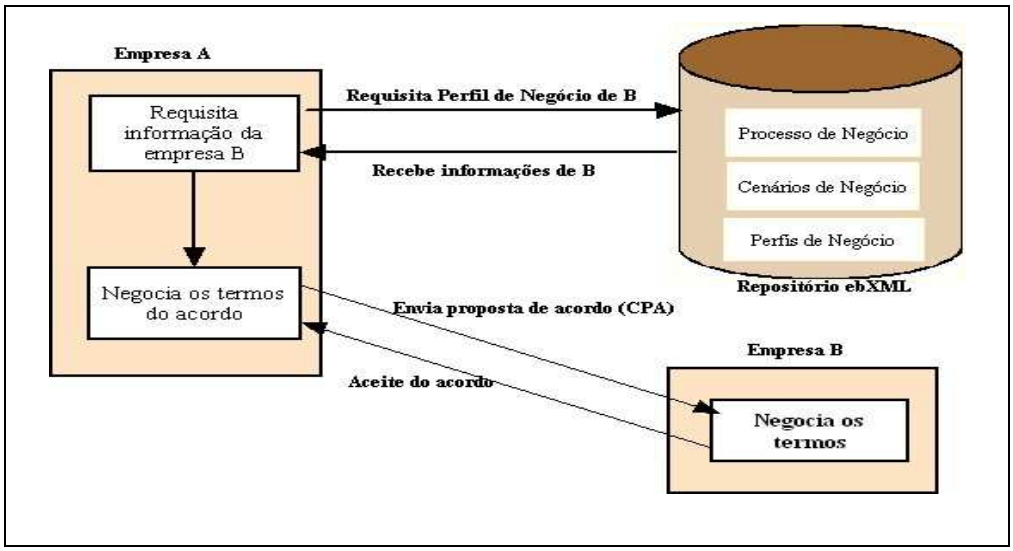

Figura 12: Fase de descoberta e Negociação [ebXMLTA,2002]

Neste cenário, a organização A fez uma pesquisa no repositório e selecionou o perfil de negócio da empresa B.

Com base nesse perfil, que é o CPP, a empresa A analisa as capacidades de negócio da empresa B para ver se está de acordo com o processo de negócio que ela está interessada em estabelecer, analisando as mensagens de negócio que podem ser trocadas, protocolos de transporte, necessidade de autenticação e outros parâmetros que a empresa B suporta.

No mundo real, empresas normalmente negociam termos e formalizam contratos de negócio antes de estabelecer uma relação comercial.

No ebXML isto não é diferente, assim o próximo passo para a empresa A é enviar uma proposta de colaboração através do Protocolo de Acordo de Colaboração - CPA, para a empresa B, estabelecendo um processo de negociação.

Neste ponto, as duas empresas devem interagir para refinar o CPA de modo que ele represente as necessidades de negócio de ambas. Este processo de negociação depende da interação humana, efetuada pelos representantes legais de cada empresa.

O CPA que resultar do processo de negociação deverá ser um reflexo dos perfís de negócio (CPP) de ambas as empresas e será utilizado para configurar e restringir o serviço de mensagem e a interface de negócio de cada um dos parceiros na fase seguinte de estabelecimento das transações de negócio.

Fase das Transações de Negócio

Nesta fase as empresas estão prontas para executar as transações de negócio referentes ao processo de negócio que foi definido para esta colaboração comercial.

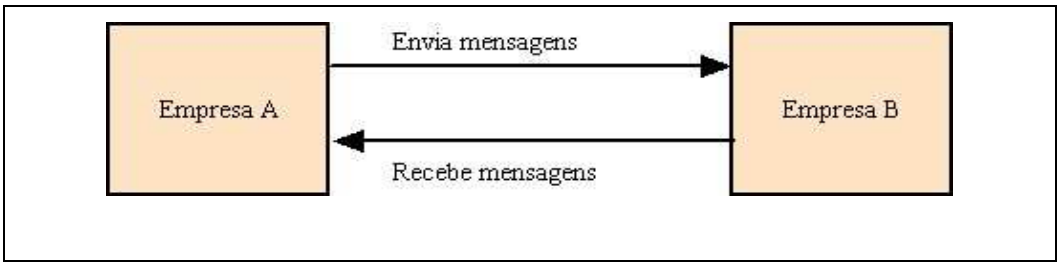

Figura 13: Fase de Transações [ebXMLTA,2002]

O CPA foi aceito na fase anterior e a interação pode ser conduzida de acordo com as regras estabelecidas no CPA, onde cada empresa executa regras pré-determinadas em cada transação de negócio.

As transações consistem de um coreografado conjunto de troca de mensagens de negócio, as quais são enviadas e controladas pelo serviço de mensagem ebXML, que por sua vez deve obedecer às regras que foram acordadas no CPA.

O CPA é o documento que contém as regras que governam estas interações, e caso ocorra alguma violação, o serviço de mensagens deve gerar uma mensagem de erro para a aplicação de negócio.

O acordo de colaboração no padrão ebXML, conforme citado na especificação do CPP/CPA [ebXML CPPA,2002], envolve uma relação binária, ou seja, o CPA é sempre estabelecido entre duas empresas, mesmo em processos de negócio que envolvam vários parceiros comerciais.

Se um determinado cenário de negócio envolve as empresas A, B e C, por exemplo, os acordos serão estabelecidos conforme o fluxo do processo de negócio, ou seja, entre as empresas que têm um relacionamento comercial direto. Se B se relaciona diretamente com A e C, então teremos um CPA entre B e A e outro entre B e C.

2.6 Considerações Finais

A geração de padrões para serviços eletrônicos continua em plena atividade, com os diversos organismos de padronização gerando periodicamente novos padrões ou novas versões de padrões existentes.

A especificação atual do serviço de mensagens do ebXML incorpora o SOAP e a partir de novas versões, poderá convergir com as especificações dos padrões para WS (*Web Services*), pois os comitês técnicos dos organismos responsáveis por estas padronizações, o W3C e o OASIS, estão trabalhando em conjunto.

Os comitês técnicos do ebXML estão examinando propostas de integração do WSDL ao CPP e analisando as sobreposições de funções entre o ebXML Registry e o UDDI, sendo que a versão atual do Registro ebXML já suporta a publicação de serviços Web [ebRS,2002].

Com relação aos componentes do padrão ebXML, o Registro, o serviço de mensagens ebMS e o ebBP, são os componentes que mais têm sido objeto de estudos, tanto na área acadêmica, quanto pelas empresas fornecedoras de software [Hofreiter e Huemer,2003].

Neste sentido, o freebXML (http://www.freebxml.org), um organismo patrocinado pela SUN e pelo governo da China para o desenvolvimento de software livre baseado no padrão ebXML, disponibiliza sistemas para uso pela comunidade, abrangendo um editor de ebBP, um ebMS e a implementação de um Registro ebXML.

Na área acadêmica, a Universidade de Hong Kong mantém um centro de pesquisas em comercio eletrônico denominado CECID (http://www.cecid.hku.hk/), onde são desenvolvidos trabalhos com o padrão ebXML, tendo sido desenvolvidos implementações do serviço de mensagens e do Registro ebXML, além de uma aplicação (*ebMail)* para edição e troca de documentos de negócio no padrão ebXML, para ser utilizado por pequenas empresas.

A Universidade de Viena (http://www.cs.univie.ac.at/) também desenvolve trabalhos com o padrão ebXML, voltados principalmente para a área de modelagem de processos de negócio e elaboração de documentos e esquemas xml, de acordo com este padrão.

A proposta do padrão ebXML, através de seu conjunto de especificações, é disponibilizar recursos para a criação de vocabulários e componentes comuns que possam substituir os vocabulários verticais, específicos de um determinado tipo de indústria.

Os componentes centrais do ebXML visam abranger variados tipos de indústria, estabelecendo uma padronização semântica que possa ser utilizada de forma horizontal por empresas de segmentos diversos.

Neste contexto, possui uma especificação bem abrangente, baseada na experiência de seus idealizadores com especificações e soluções de EDI, mas focado na colaboração entre processos de negócio.

Os padrões EDI ainda deverão continuar sendo utilizados pelas organizações, pelo menos até o ano de 2010, segundo as previsões do Gartner Group, atendendo aos mercados verticais, através de padrões específicos [Gartner, 2004].

No entanto, a adoção do ebXML vem crescendo e conforme pesquisa elaborada pelo Gartner Group [Gartner, 2004], o padrão tem sido adotado com maior ênfase, na Ásia (Japão, Corea e China), na Europa (principalmente na Alemanha) e na Austrália.

Nesses paises, os setores que mais utilizam o ebXML são a indústria automobilística, a área financeira e iniciativas de governo eletrônico, devido principalmente ao patrocínio das Nações Unidas.

Com relação a órgãos de governo, o maior impulso para a utilização do ebXML, foi a adoção deste padrão em 2004 pelo Departamento de Defesa dos Estados Unidos, para o seu portal de aquisições on-line, através da implementação das especificações do serviço de mensagens e do Registro e Repositório ebXML.

O ebXML também tem sido adotado gradativamente por grandes comunidades de comércio eletrônico, como a RosettaNet, que congrega diversas empresas do setor eletro-eletrônico, que adotou o ebXML BPSS como mecanismo padrão para expressar colaborações comerciais envolvendo seus participantes.

Um outro fator de impulso à utilização do ebXML foi a normatização pela ISO – International Standard Organization de alguns componentes do padrão ebXML sob a norma ISO 15000, de acordo com as seguintes especificações:

- ISO 15000-1: ebXML Collaborative Partner Profile Agreement
- ISO 15000-2: ebXML Messaging Service Specification
- ISO 15000-3: ebXML Registry Information Model
- ISO 15000-4: ebXML Registry Services Specification

Além de estar normatizado pela ISO, com relação aos demais padrões utilizados em serviços eletrônicos, o padrão ebXML apresenta um diferencial significativo, com a definição de componentes para a especificação de processos de negócio e componentes para serem utilizados na fase de negociação de acordos de colaboração.

A fase de negociação é uma etapa importante para o estabelecimento de um processo de colaboração eletrônica [Segev e Beam, 2001]. Nesta fase são definidos os parâmetros técnicos e comerciais (mensagens de negócio) que permitirão a interoperabilidade entre as aplicações de negócio dos parceiros comerciais.

No próximo capítulo, abordaremos o processo de negociação eletrônica e descreveremos mais detalhadamente os componentes CPP e CPA especificados no padrão ebXML para serem utilizados nesta fase.

3. PROCESSOS DE NEGOCIAÇÃO ELETRÔNICA

Neste capítulo são abordados os principais conceitos de negociação eletrônica e detalhado o processo de negociação com os componentes CPP e CPA do padrão ebXML.

3.1 Introdução

Quando necessitamos adquirir algum produto ou serviço, normalmente estabelecemos um processo de procura em vários estabelecimentos comerciais.

Algumas vezes temos que efetuar várias negociações, até decidirmos por uma proposta que melhor atenda as condições de que necessitamos, o que pode tornar demorado todo este processo.

Este cenário aborda uma situação típica do mercado entre empresas e consumidores, conhecido como B2C (*Business to Consumer*), mas no mercado B2B pode ocorrer similarmente esta mesma situação.

A negociação é uma parte importante de um processo de aquisição e a demora nos processos de negociação de uma empresa, tanto para a venda quanto para a aquisição de produtos ou serviços, pode afetar negativamente os seus processos de negócios [Byde e Piccinelli,2002].

A automatização deste processo através de um canal eletrônico como a Internet, pode contribuir para tornar esta interação mais ágil. Quanto mais rápida uma negociação for concluída, mais rápidamente uma empresa poderá iniciar o estabelecimento de um processo de negócio ou buscar um outro parceiro comercial.

O padrão ebXML foi desenvolvido visando o mercado B2B e mais especificamente a troca de documentos eletrônicos entre empresas, integrando os seus processos de negócio para o fornecimento de algum produto ou serviço.

Esta integração também é precedida de um processo de negociação, envolvendo não somente aspectos comerciais, mas também aspectos técnicos que irão suportar a comunicação e a interoperabilidade entre as aplicações das partes envolvidas.

Segundo Bartolini [Bartolini *et. al.*,2001] um processo de negociação envolve dois aspectos principais, que são o protocolo de negociação e a estratégia de negociação.

O protocolo determina o fluxo e o tipo de mensagens que será trocado entre as partes, ditando as regras que devem ser seguidas para viabilizar uma interação. O protocolo deve ser público e aberto.

Neste contexto, o protocolo de negociação envolve não somente a estrutura de mensagens, mas também toda a lógica de controle da interação entre as partes envolvidas em um processo de negociação.

A estratégia é o modo pelo qual uma determinada parte atua dentro das regras estabelecidas para tentar conseguir o melhor acordo na negociação, envolvendo, por exemplo, aspectos do que conceder, quando, como e o que não conceder, com relação às condições e parâmetros em negociação. A estratégia de cada participante é necessariamente privada.

Neste trabalho abordamos os aspectos relacionados à formatação e padronização de mensagens que possam possibilitar o desenvolvimento de um processo automatizado de negociação no contexto do padrão ebXML, não sendo objetivo aprofundarmos em aspectos de estratégia.

3.2 Negociação Eletrônica

A negociação pode ser definida como um meio através do qual dois ou mais atores chegam a um acordo ou resultado específico, utilizando alguma estratégia de interação ou efetuando decisões interdependentes [Belluci e Zeleznikow, 1999].

O processo de negociação ocorre quando um acordo não pode ser efetuado com base em uma proposta inicial, mas apresenta boas possibilidades de ser concretizado e as partes estão dispostas a discutir suas posições [Strobel, 2002].

A partir desta situação, o processo de negociação é caracterizado pelas mudanças que uma das partes efetua em sua própria proposta ou pelo esforço de convencer a outra parte a mudar as condições por ela estabelecidas.

Conforme [Segev, 2001], um processo de negociação eletrônica é definido como um processo no qual duas ou mais partes envolvidas na negociação, utilizam sistemas de software que negociam eletronicamente uma solução.

Apesar do processo de negociação entre seres humanos ser um processo complexo e difícil de ser substituído completamente, a utilização de sistemas computacionais em processos de negociação eletrônica tem sido objeto de estudo em vários trabalhos de pesquisas.

Segev cita o uso de Sistemas de Suporte a Negociação - NSS e de Agentes Inteligentes em processos de negociação eletrônica [Segev, 2001], Nunes propõe o uso de multi-agentes [Nunes et *al.*, 2003], Bartolini aborda os requisitos necessários para o desenvolvimento de um protocolo de negociação [Bartolini *et.al.*,2001], Byde apresenta um *framework* com técnicas de reconciliação automática de propostas [Byde e Piccinelli,2003] e Strobel propõe um processo de contagem de pontos para comparação e classificação de propostas visando determinar a melhor oferta, em um processo de negociação [Strobel, 2002].

3.2.1 Tipos de Negociação

Os processos de negociação podem ser classificados de acordo com os interesses envolvidos e a quantidade de participantes [Strobel, 2002].

Com relação aos interesses, uma negociação é classificada como Distributiva, quando as partes envolvidas têm interesses opostos e estão mais interessadas em tirar o máximo proveito uma da outra, e pode ser classificada como Integrativa, quando as partes envolvidas têm interesses comuns e estão interessadas em estabelecer um acordo de colaboração mutua para alcançar um objetivo comum.

Uma negociação é classificada como Bilateral quando envolve somente duas partes e tem um caráter privado a estas partes.

Multilateral é uma negociação que envolve múltiplos parceiros e tem um caráter público, por que normalmente as propostas oferecidas podem ser consultadas por todos os envolvidos. Este tipo de negociação é caracterizado como uma competição pública.

Combinações destes tipos de classificação podem ser utilizadas para o projeto de protocolos de negociação. Um protocolo de negociação define as regras através das quais as partes poderão chegar a um acordo.

Estas classificações não são persistentes quando aplicadas em negociações no mundo real, ou seja, dependendo dos interesses das partes durante o desenvolvimento do processo de negociação, estas podem alterar o seu posicionamento de distributiva para integrativa e vice-versa [Strobel, 2002].

No contexto do padrão ebXML, a princípio as negociações tendem a se caracterizar como negociações bilaterais integrativas, pois visam estabelecer um processo de colaboração comercial mútua entre dois parceiros comerciais.

No entanto, uma empresa de maior porte pode obrigar empresas menores a seguir os seus parâmetros, caso estas desejem estabelecer um processo de colaboração com ela.

3.2.2 A automatização de negociações eletrônicas

Segev cita duas grandes dificuldades para se automatizar uma negociação: a necessidade de se estabelecer uma ontologia e a necessidade de se estabelecer estratégias [Segev, 2001].

A ontologia é a forma de categorizar objetos, ou seja, utilizar uma semântica para representação dos objetos de negócio que seja de entendimento comum a todos os envolvidos, de forma que durante a negociação todos os sistemas envolvidos no processo estejam se referindo exatamente a um mesmo objeto.

Quando o objeto de negociação é um produto simples como um livro, por exemplo, não é difícil garantir uma semântica [Segev, 2001], mas quando a negociação envolve um processo de negócio com diversos itens de especificações técnicas e comerciais, tais como documentos de negócio, definição das transações comerciais, protocolos de transporte, entre outros, o correto entendimento de todos os itens torna-se crucial para o processo.

A necessidade de uma estratégia de negociação envolve a definição de quais informações deverão ser utilizadas e como combinar estas informações para se efetuar as barganhas durante o processo, visando atingir o melhor resultado.

O padrão ebXML define mecanismos que objetivam assegurar um vocabulário comum de negócios, como o BPSS (*Business Process Specification Schema*), que indica os elementos e atributos que poderão ser utilizados em um processo de negócio.

Com relação a estratégias de negociação, o ebXML não aborda este tipo de recurso, limitando-se a especificar somente o documento para elaboração do acordo de colaboração, o CPA.

Strobel [Strobel, 2002] cita que se a discussão em uma negociação envolver múltiplos atributos, a determinação do acordo de colaboração não é trivial. Por esta razão, a tarefa de

decisão em processos de barganha envolvendo negociações integrativas são mais difíceis de formalizar e a possibilidade de uma automatização completa é bastante questionável.

Neste caso, a opção mais razoável conforme [Strobel, 2002], é desenvolver um sistema de negociação para automatizar atividades centrais de um processo de negociação (comparação de propostas, por exemplo), combinados com algum processo de intervenção humana, como ocorre nos sistemas de suporte à negociação.

3.3 Sistemas de Suporte à Negociação

Segundo [Segev, 2001], um Sistema de Suporte a Negociação (NSS – *Negotiation Support System*), é um software especialmente desenvolvido para ajudar negociadores humanos a fazer a melhor decisão e negociar de uma forma mais produtiva.

Um NSS requer constante interação com negociadores humanos, que providenciam a entrada de informações durante o processo de negociação. A configuração inicial do sistema, assim como todas as decisões finais, são deixadas pelo sistema para serem efetuadas por negociadores humanos.

Os sistemas do tipo NSS suportam o processo de negociação, mas não modelam de forma satisfatória aspectos de decisão de um problema [Belluci e Zeleznikow, 1999]. Para melhoria deste aspecto, são utilizados Sistemas de Suporte a Decisão em Negociação – NDSS (*Negotiation Decision Support Systems*).

Um NDSS é similar a um NSS, mas disponibiliza recursos adicionais para o suporte à decisão, efetuando propostas de ofertas e sugestões com relação ao problema que está sendo discutido.

A Universidade de Concórdia em Montreal mantém um centro de pesquisa em negociação (http://interneg.org) que disponibiliza, entre outras ferramentas, um sistema NSS interativo chamado Inspire para uso educacional via Web, que auxilia negociadores humanos a solucionar um problema de negociação, fornecendo um estudo de caso e um papel (vendedor ou comprador) para o desenvolvimento do processo de negociação.

3.4 A colaboração de negócios no padrão ebXML: o CPP e o CPA

Strobel [Strobel, 2002] argumenta que a publicação previa de informações sobre os parceiros comerciais e sobre os produtos que tratam os seus processos de negócios são importantes para agilizar um processo de negociação, porque os possíveis parceiros já terão avaliado as possibilidades positivas de um acordo, antes de iniciar o processo de negociação.

O padrão ebXML especifica o componente CPP (Collaboration Protocol Profile), para descrever as informações sobre um ou mais processos de negócio que uma empresa está disposta a executar com outros parceiros, e o CPA (Collaboration Protocol Agreement), que abrange os itens do CPP de cada empresa, que foram negociados para serem utilizados em um processo de colaboração comercial.

Em um processo de colaboração, a comunicação entre os parceiros comerciais deve ser definida de forma que as partes envolvidas tenham um claro entendimento do que cada um pode fazer durante o desenvolvimento do processo [Gupta *et. al*., 2006].

O Perfil do Protocolo de Colaboração – CPP (*Collaboration Protocol Profile*) é um documento XML onde uma organização identifica o processo de negócio que deseja estabelecer com um parceiro comercial, o papel que executa neste processo e as definições dos canais de comunicação e protocolos de transporte (como Htttp, por exemplo) que suportam esta colaboração.

O Acordo do Protocolo de Colaboração – CPA (*Collaboration Protocol Agreement*) contém a descrição do acordo para a troca de mensagens entre dois parceiros comerciais, ou seja, o CPA define o modo como as partes deverão interagir na execução da colaboração de negócio.

Conforme a especificação destes componentes [ebXML CPPA, 2002], estão inclusos no CPP e no CPA, os detalhes de transporte, restrições de segurança e referência para o documento de Especificação de Processo de Negócio (ebBPS - *Business Process Specification*), que contém as definições dos elementos que compõe o processo de negócio (mensagens e documentos de negócio) que deverão ser utilizados na colaboração de negócios.

As partes envolvidas em um processo de colaboração devem usar a mesma cópia (versão) do CPA para configurar suas aplicações ebXML internas, independentemente de estes sistemas serem ou não de um mesmo fornecedor.

O processo de configuração deve ser automático, o que significa que uma aplicação no ambiente interno de cada parceiro dever ler o CPA e executar o processo de configuração das interfaces das aplicações de negócio (*BSI – Business Service Interface*) de cada um deles, conforme os parâmetros especificados no acordo.

Normalmente, o software que executa as trocas de mensagens e suporta a interação entre as partes é um *middleware* que pode suportar uma colaboração de negócio ebXML e repassar as mensagens para uma aplicação de negócio efetuar o processamento [ebXML CPPA, 2002]. Um dos componentes deste *middleware* pode ser o serviço de mensagens ebXML (ebMS).

O CPP deve ser armazenado em um repositório de um Registro ebXML. Com o uso de um processo de descoberta, providenciado como parte das especificações do repositório, qualquer parceiro comercial pode efetuar um processo de busca e analisar os diversos CPP armazenados no repositório para encontrar novos parceiros de negócio.

A interação entre dois parceiros pode ser efetuada através de um intermediário, como um portal ou um provedor de serviço, por exemplo. Neste caso, deverá haver um CPA entre cada parceiro e o intermediário, com definições das funcionalidades entre cada um e este intermediário.

Os processos de composição do CPP e do CPA podem ser efetuados manualmente ou automaticamente.

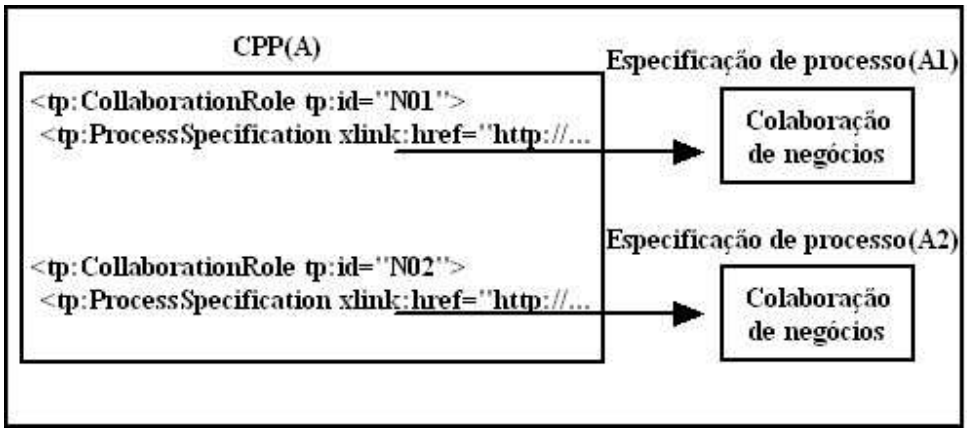

Figura 14: Estrutura de um CPP/CPA [ebXML CPPA,2002]

Conforme pode ser observado na figura 14, uma empresa pode criar vários CPP que descrevam, por exemplo, diferentes tipos de colaborações de negócio que ela suporta, referentes às suas operações em diferentes regiões do mundo ou referentes a processos de negócio de departamentos distintos de sua organização, ou referenciar estas diversas especificações de processos em um mesmo CPP.

Um CPP pode referenciar um ou mais processos de negócios (*BPS – Business Process Specification*) suportados por uma organização e as regras que a empresa é capaz de assumir dentro de cada processo de negócio.

Um exemplo de uma regra pode ser a noção de "vendedor" e de "comprador" em um processo de negócio de compra.

No CPP deve ficar clara esta regra, indicando se a empresa é a parte vendedora ou a compradora dentro deste processo de negócio.

No CPP também são descritos os detalhes de ligações que são utilizados para construir o cabeçalho de uma mensagem ebXML.

Estes detalhes abrangem, por exemplo, qual o tipo de protocolo de transporte que é utilizado (HTTP, SMTP ou FTP, entre outros) em determinado processo de negócio, que tipo de mensagem é suportado, se é do tipo notificação, em uma única direção, ou do tipo solicitação/resposta, em ambas as direções.

Estas informações são posteriormente incluídas no CPA quando um acordo é celebrado e visam garantir a interoperabilidade entre as aplicações dos parceiros.

Termos legais de um acordo de negócio estão fora do escopo das especificações e não são incluídos dentro do CPP e do CPA.

Em geral, os parceiros descritos em um CPA podem ter características tanto de cliente como de servidor. Há processos de negócio em que o relacionamento comercial tem as características de um tradicional processo cliente/servidor e outros em que é estabelecida uma relação par a par, onde cada parte pode requisitar serviços da outra parte.

Independentemente da forma de relação, o CPA deve abranger todos os papeis e regras que cada participante deve executar em um determinado processo de negócio.

3.4.1 Formando um CPA a partir de dois CPP

Uma empresa A e uma empresa B que desejem estabelecer um processo de colaboração comercial, utilizam os seus CPP para negociar a construção de um único CPA, através da intersecção das informações de seus CPP.

O CPA resultante define como os dois parceiros irão se comportar na execução de sua colaboração de negócios.

No padrão ebXML não há uma especificação para o processo de negociação e geração do CPA. As especificações do CPP/CPA[ebXML CPPA, 2002] citam uma descrição básica de um possível procedimento para o processo de geração de um CPA, conforme pode ser observado na figura 15 abaixo:

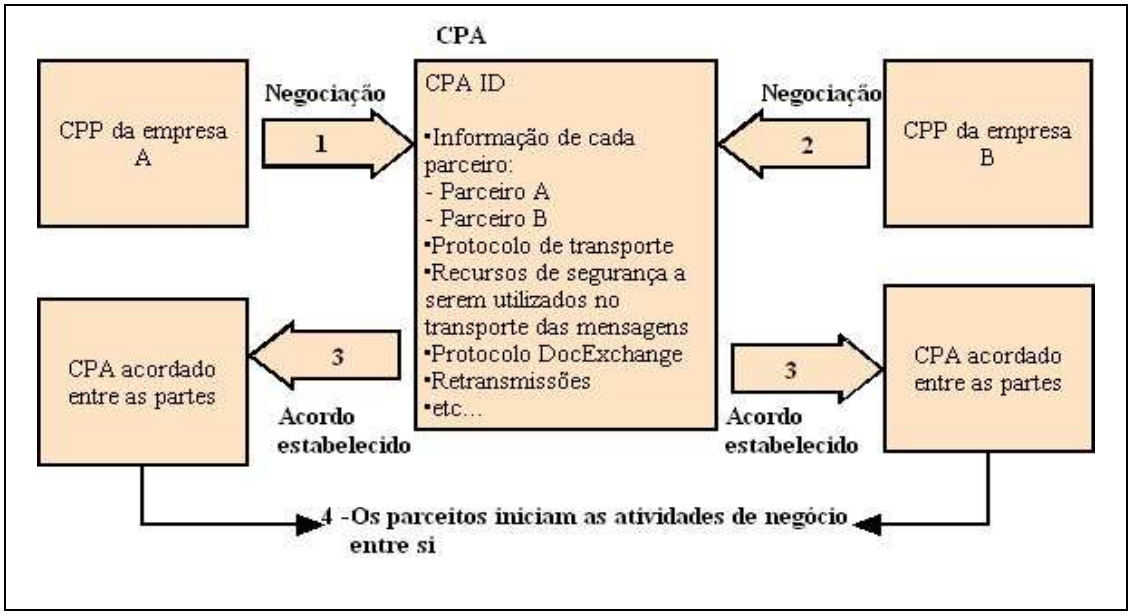

Figura 15: Procedimentos básicos para elaboração do CPA [ebXML CPPA, 2002]

Passo 1 – parte A cria a primeira versão de um CPA (A, B) e envia para a parte B.

Passo 2 – partes A e B negociam até chegar a um acordo quanto aos itens do CPA. Este processo é efetuado manualmente, mas pode ser automatizado.

Passo 3 – Partes A e B configuram seus sistemas aplicativos de acordo com as informações do CPA.

Passo 4 – Partes A e B iniciam a frase de transações de negócios, regidos pelas regras acordadas e especificadas no CPA.

Alternativamente, um CPA pode ser formado a partir de um modelo de CPA. O modelo representa uma proposta em branco a ser preenchida pela outra parte. Os elementos XML

do CPA que identificam a outra parte podem ter um valor inicial fictício que deve ser sobreposto pelos valores obtidos do CPP da outra parte.

Ao contrário do CPP que pode referenciar vários processos de negócio, um CPA deve se referir somente a um Processo de Negócio e aos requisitos necessários para executá-lo.

Este processo de negócio (ebBPS) deve estar registrado em um Registro ebXML, possibilitando que o CPP e o CPA possa referencia-lo.

Se houver alteração de qualquer parâmetro contido dentro de um CPA já executado, um novo CPA deverá ser negociado entre as partes envolvidas**.**

3.4.2 Utilização do CPA

Um CPA descreve todas as interações válidas definidas de comum acordo entre as partes envolvidas, mas é independente dos processos internos executados em cada parte.

A informação contida no CPA é usada para configurar os sistemas das partes envolvidas, habilitando-os a suportar a troca de mensagens relativa à colaboração de negócio descrita pelo acordo.

A conversação representa uma simples unidade de negócio definida pelo elemento *BinaryCollaboration* do documento de especificação de processos de negócios (ebBPS) e consiste de uma ou mais transações de negócio, cada uma das quais representando uma mensagem de requisição ou de resposta (r*equest/response*) entre as partes.

O ebBPS define, entre outras coisas, as mensagens de requisição e resposta para cada transação de negócio e a ordem em que estas transações devem ocorrer.

Uma conversação termina quando as partes completarem a unidade de negocio, conforme determinado no acordo de Colaboração de Negócios.

O sistema de execução (um middleware, por exemplo) em cada parceiro, deve providenciar uma interface através da qual a aplicação de negócio pode requisitar a iniciação ou o fim de uma conversação.

Conceitualmente, o CPA dever ser implementado em um servidor de aplicações de negócios B2B em cada parte envolvida.

O CPA descreve os requisitos necessários para a configuração da interface do serviço de mensagem de cada parceiro, assim como os detalhes de implementação mutuamente acordados, que ambos os parceiros deverão configurar para a execução do processo de colaboração.

3.4.3 Os principais elementos do CPP

Um CPP define as capacidades de uma empresa para executar uma colaboração eletrônica com outras partes, e o CPA é um subconjunto, formado a partir da intersecção destas capacidades.

Estas capacidades abrangem as capacidades tecnológicas, tais como os protocolos de mensagens e de comunicação suportados e as capacidades comerciais em termos de quais colaborações de negócio a empresa suporta.

A estrutura de um CPP consiste de um elemento raiz, denominado Collaboration ProtocolProfile, que contém elementos como *PartyInfo, Packaging, Signature* e *Comment.*

A figura 16 abaixo apresenta a estrutura de um documento CPP com estes elementos.

```
      <CollaborationProtocolProfile
xmlns="http://www.ebxml.org/namespaces/tradePartner"
xmlns:ds="http://www.w3.org/1999/xmldsig#"
xmlns:xlink="http://www.w3.org/2000/09/xlink"
      version="1.1">
        <PartyInfo>  <!--one or more-->
        ...
      <!—Required -->
        </PartyInfo>
        <Packaging id="ID"> <!--one or more-->
          ...
            <!—Required -->
        <Packaging>
  <ds:Signature>  <!--zero or one-->
          ...
      <!— Optional -->
        </ds:Signature>
        <Comment>text</Comment> <!--zero or more-->
      ...
      <!— Optional -->

      </CollaborationProtocolProfile>
```

Figura 16: Estrutura de elementos do CPP [ebXMLXPPA,2002]

A figura 16 mostra os principais elementos XML do CPP, iniciando pelo elemento raiz *CollaborationProtocolProfile,* dentro do qual estão as declarações dos *namespaces* relativos aos esquemas XML, além dos atributos e demais elementos utilizados no documento.

As declarações dos esquemas XML abrangem o esquema do CPP (cpp-cpa-2_0.xsd) que está sendo utilizado neste documento, o esquema padrão do W3C (*XMLSchema-instance*), a localização do esquema do CPP, o esquema para os elementos de segurança (*xmldsig#*) e elementos para declaração de ligações a outros documentos (*xlink*).

O atributo *version* é utilizado para declaração da versão do CPP e na seqüência, são declarados os elementos *PartyInfo* (para inclusão de informações das partes), *Packaging*, onde são declarados os elementos com informações para o empacotamento de mensagens e *Signature*, onde os elementos de segurança utilizados por uma empresa devem ser declarados.

Os elementos *PartyInfo* e *Packaging* são obrigatórios e os elementos *Signature* e *Comment* são opcionais.

No elemento PartyInfo estão especificados os principais elementos de definição de parâmetros de interoperabilidade, tais como o *ProcessSpecification, DeliveryChannel, DocExchange* e *Transport*.

Conforme a especificação [ebXML CPPA,2002], estes elementos formam uma estrutura de camadas, análoga às camadas de um modelo de comunicação.

Camada de Especificação do Processo (*ProcessSpecification*)

O elemento *ProcessSpecification* define o coração do acordo de negócios entre as partes, abrangendo os serviços (transações de negócio) e as regras de transição que determinam a ordem das requisições, e que deverão fazer parte de um acordo de negócio (CPA) .

Esta camada é definida em um documento separado, no documento de especificação de processos (ebBPS – ebXML *Business Process Specification*), que deve ser referenciado por um CPP e por um CPA.

Os processos de negócio são publicados em um registro ebXML e passam a ser utilizados como processos padrão para o estabelecimento de colaborações comerciais entre uma determinada comunidade de empresas, registradas naquele registro [ebXML CPPA,2002].

Camada de definição do Canal de Entrega (*DeliveryChannel*)

Descreve as características de recebimento e transmissão de mensagens. É onde é efetuada a definição do intercâmbio de documentos e definições de transporte das mensagens. Vários canais de entrega podem ser especificados em um CPP.

Camada de Intercâmbio de documentos (*DocExchange*)

Esta camada aceita um documento de negócio a partir da camada de especificação de processos (*ProcessSpecification*), efetua a sua encriptação, se assim for especificado, adiciona uma assinatura digital para não repúdio, se especificado, e passa o documento para a camada de transporte, para transmissão para a outra parte. No recebimento das mensagens é efetuado o processo inverso.

As opções selecionadas para a camada de intercâmbio de documentos são complementares a aquelas selecionadas para a camada de transporte.

Por exemplo: se for desejável a segurança de uma mensagem e o protocolo de transporte selecionado não providencia recursos de criptografia, então isto dever ser especificado na camada de intercâmbio de documentos.

O protocolo de transporte para a troca de mensagens entre as partes (HTTP,SMTP ou outro) é definido pela especificações do serviço de mensagens ebXML (ebMS) ou por outro serviço similar.

Camada de Transporte

É o elemento responsável pela entrega das mensagens, usando o protocolo de transporte selecionado, tais como o HTTP, SMTP, FTP ou outro.

O protocolo selecionado afeta as escolhas selecionadas para a camada de intercâmbio de documentos. Por exemplo, alguns protocolos de transporte podem providenciar recursos de criptografia e autenticação, enquanto outros não têm estes recursos, implicando que estes itens sejam definidos na camada de intercâmbio de documentos, quando necessário.

A especificação [ebXML CPPA, 2002] define diversos elementos internos para cada um destes elementos que formam a estrutura do CPP, o que permite descrever diversas características necessárias ao estabelecimento do processo de colaboração.

No Anexo I deste trabalho mostramos um exemplo de um documento CPP com todos estes elementos.

O CPP e o CPA possuem um esquema XML próprio (cpp-cpa-v1_0xsd), compatível com o ebXML *Business Process Specification Schema* (ebBPSS), que define os elementos que poderão ser utilizados nestes documentos.

3.4.4 Os principais elementos do CPA

O CPA define o acordo de colaboração em termos de sistema, para viabilizar a troca de dados entre os parceiros [Gupta *et. al.*, 2006], abrangendo um conjunto de definições técnicas que são suportadas pelas aplicações ebXML de ambas as partes.

O CPA atua como um acordo de nível de serviço que pode ser cumprido pelos parceiros de negócio, pois é elaborado com base nas capacidades técnicas descritas no CPP de cada parceiro.

A estrutura do CPA é similar a estrutura do CPP, conforme pode ser observado na figura 17.

```
<CollaborationProtocolAgreement
xmlns="http://www.ebxml.org/namespaces/tradePartner"
xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
xmlns:xlink=http://www.w3.org/1999/xlink
cpaid=”http://www.example.com/mycpa"
version="1.1">
<Status value = “proposed”/>
<Start>2006-11-20T07:21:00</Start>
<End>2007-11-20T07:21:00</End>
<ConversationConstraints invocationLimit = “250”
concurrentConversations = “5”/>

  <PartyInfo>  <!--one or more-->
    ...
      <!—Required -->
  </PartyInfo>
  <PartyInfo>  <!--one or more-->
    ...
      <!—Required -->
  </PartyInfo>
  <Packaging id="ID"> <!--one or more-->
    ...
      <!—Required -->
  <Packaging>
  <ds:Signature>  <!--zero or one-->
    ...
      <!— Optional -->
  </ds:Signature>
  <Comment>text</Comment> <!--zero or more-->
      ...
      <!— Optional -->

</CollaborationProtocolAgreement>
```

Figura 17: Estrutura de elementos do CPA [ebXMLXPPA,2002]

A figura 17 mostra os principais elementos XML do CPA, iniciando pelo elemento raiz *CollaborationProtocolAgreement,* dentro do qual estão as declarações padrão dos *namespaces* relativos aos esquemas XML utilizados no documento., além do atributo de versão, para distinguir o documento de mudanças subseqüentes, e do atributo *cpaid,* para identificação do documento.

Os elementos *PartyInfo, Packaging*, *Signature* e Comment têm o mesmo significado que eles tem no CPP, com exceção do elemento PartyInfo que é incluído duas vezes, para representar as informações das partes envolvidas no acordo.

Assim como no CPP, os elementos *PartyInfo* e *Packaging* são obrigatórios e os elementos *Signature* e *Comment* são opcionais.

O elemento Status mostra em que ponto o processo de acordo está, podendo assumir os valores de *proposed, agreed e signed.*

Os elementos *Start* e *End* representam o tempo de inicio e de fim do período de validade do CPA.
O elemento opcional *ConversationConstraints* é utilizado para especificar um valor finito de conversações que podem ser estabelecidas durante este acordo (um acordo temporário, baseado no número de conversações) e o número máximo que podem ser estabelecidas concorrentemente.

No Anexo I deste trabalho mostramos um exemplo de um documento CPA completo, abrangendo todos estes elementos.

Os papeis, documentos de negócio e demais parâmetros de interoperabilidade defnidos no CPA serão utilizados para a configuração das aplicações de negócio e do serviço de mensagens de cada parceiro.

3.5 Considerações Finais

Segundo Strobel, os custos de uma interação eletrônica, envolvendo o anúncio de informações, busca por potenciais parceiros e subseqüente processo de transação são geralmente baixos devido ao grau de automação e conectividade proporcionada pela Internet [Strobel, 2002].

Estes fatores levam a concluir que a demanda por negociações em mercados de comércio eletrônico, dentro da fase de elaboração de acordos, deverá aumentar. O suporte a negociações eletrônicas não é somente necessário, mas também pode ser um fator critico de sucesso para a agilização do processo de negócio de muitas empresas.

O ebXML está voltado para a integração de processos de negócio e surge como uma alternativa para que as empresas possam estabelecer colaborações comerciais com empresas de mercados diversos, sem ter que adquirir soluções tecnológicas especificas destes mercados.

Conforme observado por Segev [Segev, 2001] e Strobel [Strobel, 2002], negociações são um mecanismo crítico para a interação entre empresas com vistas ao futuro do comércio

eletrônico, mas disponibilizar uma automação completa do processo de negociação, sem intervenção humana, ainda não é plausível atualmente.

A atual especificação do CPP/CPA [ebXML CPPA, 2002] está mais focada em aspectos técnicos de conectividade e interoperabilidade para o estabelecimento de um processo de colaboração eletrônica entre dois parceiros comerciais e apesar de citar a hipótese de que este processo pode ser automatizado, não especifica mecanismos para esta automatização.

A necessidade de elaboração de mecanismos para a geração automatizada do acordo de colaboração (CPA) a partir da intersecção de dois CPP é também apontada por Hofreiter como um dos problemas a serem resolvidos com relação a estes documentos [Hofreiter e Huemer, 2003].

Hofreiter levanta também a necessidade de elaboração de trabalhos de pesquisa com relação a estes componentes do padrão ebXML, em razão ainda da falta de relatos de experiências envolvendo estes documentos, tanto por parte de fornecedores de software quanto pela comunidade acadêmica [Hofreiter e Huemer,2003].

A proposta deste trabalho está focada na interação entre os parceiros comerciais para a geração automatizada de documentos de acordo de colaboração com o padrão ebXML, com base em suas capacidades técnicas de interoperabilidade e nas análises e configurações efetuadas por negociadores humanos.

O uso de agentes inteligentes e outras tecnologias de inteligência artificial estão fora do escopo deste trabalho, apesar dessa área de pesquisa poder ser utilizada em futuros trabalhos complementares a esta proposta.

Neste contexto, discutiremos no próximo capítulo os aspectos de modelagem e requisitos necessários para a elaboração de uma proposta que possa automatizar determinadas atividades deste processo, auxiliando negociadores humanos e agilizando a elaboração de um acordo de colaboração.

4. A FORMAÇÃO DO CPA: REQUISITOS PARA UM SISTEMA DE NEGOCIAÇÃO ELETRÔNICA

Neste capítulo são discutidos os requisitos necessários para o desenvolvimento de um sistema que possa suportar um processo de negociação eletrônica e geração do Acordo do Protocolo de Colaboração – CPA, entre aplicações de comércio eletrônico que utilizem o padrão ebXML, assim como a modelagem inicial para a implementação deste sistema.

4.1. Introdução

Em um mundo onde os mercados estão cada vez mais globalizados e as empresas necessitam diminuir custos e ter agilidade em seus processos de negócio, utilizar um mecanismo automatizado que possa agilizar atividades de negociação que compõem estes processos, pode contribuir diretamente para acelerar a fase de produção e para a obtenção de melhores resultados.

A utilização de Sistemas de Suporte à Negociação, conforme observado por [Segev, 2001], exemplificam a utilização de sistemas que automatizam parcialmente um processo de negociação, provendo informações de modo a auxiliar um elemento humano a tomar decisões mais rápidas na condução destes processos.

A automatização de processos de negociação, mesmo parcialmente, reduz a necessidade de intervenção manual e pode contribuir para que seja avaliado um número maior de propostas, auxiliando e agilizando a elaboração de acordos de colaboração, que possam atender, da forma mais satisfatória possível, as necessidades comerciais de uma empresa.

Conforme abordado no capitulo anterior, o Protocolo do Acordo de Colaboração – CPA abrange uma série de itens de conectividade e de integração de processos que visam garantir a interoperabilidade das aplicações de negócio em um processo de colaboração eletrônica.

A geração automatizada deste acordo, assim como a automatização do processo de negociação que é desenvolvido para a elaboração deste acordo, pode envolver diversos aspectos, conforme as condições em que uma negociação é desenvolvida.

Um processo de negociação eletrônica pode ser desenvolvido diretamente entre dois parceiros ou pode haver a intermediação de um terceiro, que atue provendo serviços de negociação, buscando facilitar o estabelecimento de um acordo de colaboração entre os parceiros envolvidos no processo [ebXML CPPA, 2002].

Além deste aspecto, um processo de negociação eletrônica pode envolver também interesses de negócio, onde cada parte procura obter o máximo possível de vantagens durante o desenvolvimento do processo.

No caso do ebXML, a especificação atual não contempla aspectos estratégicos de negócio, focando nos aspectos técnicos de interoperabilidade.

Neste contexto, em uma negociação para geração do CPA, um dos principais ganhos que uma empresa buscaria obter, é o rápido estabelecimento de um processo de colaboração, preferencialmente, sem precisar implementar recursos técnicos adicionais e sem alterar a

configuração de suas aplicações ou, conforme os interesses de negócio, alterá-los o mínimo possível.

Um outro aspecto a considerar é que, para possibilitar a interação automatizada entre os parceiros comerciais na fase de negociação, há necessidade de definição de um protocolo para negociação, ou seja, uma forma de se comunicar, que possibilite que as partes envolvidas possam se entender, através da utilização de um fluxo de mensagens padronizadas.

A especificação atual dos protocolos de colaboração [ebXML CPPA, 2002] não abrange a definição de um protocolo de negociação, ou seja, deixa em aberto o processo de interação entre parceiros, assim como o processo de geração automatizada do acordo de colaboração, limitando-se a abordar algumas sugestões para que este processo possa ser automatizado.

A necessidade de especificação de um sistema automatizado para geração do acordo de colaboração (CPA) é citada por Hofreiter [Hofreiter e Huemer, 2003] e por Byde [Byde e Piccinelli, 2003], mas estes trabalhos não chegam a descrever esta especificação.

Este trabalho visa o desenvolvimento de uma proposta para atender a este problema, validando a hipótese apontada na especificação [ebXML CPPA, 2002] com relação à possibilidade de geração automatizada deste acordo.

O sentido de automatização neste contexto é auxiliar a participação humana no processo de negociação, fornecendo informações para a definição dos parâmetros envolvidos na negociação e decisão do processo.

Todo o trabalho relativo ao controle, recebimento e envio de mensagens, armazenamento de informações e geração do acordo de colaboração (CPA), a partir das informações fornecidas, deve ser efetuado por um sistema automatizado.

O desenvolvimento da proposta envolve um aspecto principal, relativo à geração do acordo de colaboração e um aspecto adicional, relativo à elaboração de um protocolo básico de mensagens de negociação, a ser utilizado durante o processo de negociação.

4.2. A formação do CPA

Para a formação do Acordo do Protocolo de Colaboração, após um dos parceiros efetuar a busca de um perfil de colaboração (CPP) em um Registro ebXML, é necessário que este elabore uma proposta para ser encaminhada à empresa detentora daquele CPP, com base nas capacidades técnicas especificadas naquele documento.

Uma análise do elemento raiz do CPP, o *CollaborationProtocolProfile* , assim como do elemento raiz do CPA, o *CollaborationProtocolAgreement*, mostra que um CPA pode ser visto como resultado da combinação de partes dos elementos *PartyInfo,* definidos dentro deste elemento raiz nos CPP de cada parceiro.

Conforme abordado no capitulo anterior, o elemento PartyInfo engloba, além de elementos de identificação da empresa, diversos elementos filhos que definem parâmetros de interoperabilidade, tais como *ProcessSpecification, DeliveryChannel, Transport, TransportSecurity* e *DocExchange,* a serem utilizados em um processo de colaboração.

A combinação de elementos dos CPPs dentro de CPAs é uma forma pelo qual parceiros comerciais podem elaborar um CPA inicial para começar um processo de negociação.

A especificação do CPP/CPA sugere a criação de um CPA modelo, que poderia ser elaborado e preenchido inicialmente por uma empresa e encaminhado para um possível parceiro [ebXMLCPPA,2002].

Este CPA-modelo poderia representar a proposta de implementação de um processo de negócio, sob o ponto de vista de uma das partes, que utilizaria valores padrão para identificar aspectos específicos da outra parte como, por exemplo, valores para os elementos *PartyId* e *TransportEndpoint* [ebXML CPPA, 2002].Conforme o esquema XML do CPP-CPA, é requerido o preenchimento destes elementos, ou seja, eles não podem ficar vazios para não invalidar o CPA.

Estes valores seriam reescritos pela outra parte, abrangendo os itens relativos à sua identificação e capacidades técnicas que ela esteja disposta a utilizar para estabelecer o processo de colaboração.

Neste CPA-modelo, constaria um conjunto mínimo de itens para garantir a interoperabilidade entre os parceiros, nesta fase de negociação. Este documento poderia ser publicado em um Registro, junto com o CPP da empresa, para ser descoberto e analisado por um eventual parceiro comercial.

A especificação também cita, como sugestão, a possível utilização de um Documento Descritor de Negociação – NDD, que poderia ser utilizado para descrever o que é negociável dentro do CPP ou de um CPA modelo, indicando à outra parte o que seria permitido alterar [ebXML CPPA, 2002].

Os documentos NDD poderiam ser trocados no início da negociação e identificariam o CPP ou o CPA modelo de cada parte, que seriam objeto de negociação, ou seja, o processo de negociação seria iniciado através destes documentos, em uma fase de pré-negociação.

A especificação técnica do CPP/CPA não descreve a estrutura destes documentos, tanto do NDD quanto do CPA modelo, limitando-se a relacionar estes documentos como sugestão para o desenvolvimento de um possível processo automatizado para gerar o CPA.

A utilização de documentos auxiliares para a geração do CPA pode tornar o processo de automatização mais complexo, gerando a necessidade de se desenvolver controles adicionais para a geração e gerenciamento das alterações efetuadas nestes documentos.

A elaboração de um CPA inicial, pela parte interessada em iniciar um processo de negociação, baseado nas capacidades descritas no CPP e focado principalmente nos itens essenciais para se estabelecer a interoperabilidade entre as aplicações dos parceiros, é uma hipótese mais plausível para estabelecer este processo.

Neste caso, ambas as partes utilizariam o mesmo documento de CPA, gerando uma nova versão ao efetuar alguma alteração durante o processo de negociação.

O controle destas versões poderá ser efetuado utilizando-se o atributo *version*, existente no elemento *CollaborationProtocolAgreement* do CPA, conforme fragmento do CPA mostrado na figura 18.

```
<?xml version="1.0" ?>
<tp:CollaborationProtocolAgreement
    xmlns:tp="http://www.ebxml.org/namespaces/tradePartner"
    xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
    xsi:schemaLocation="http://www.ebxml.org/namespaces/tradePartner
    http://ebxml.org/project_teams/trade_partner/cpp-cpa-v1_0.xsd"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"      tp:cpaid="uri:mycpa"
    tp:version="1.2">
<tp:Status tp:value="proposed" />
```

Figura 18: O atributo version do elemento CollaborationProtocolAgreement [CPPA,2002]

O elemento *Status*, que aparece na última linha do fragmento da figura 18, também pode ser utilizado durante o processo de negociação para indicar se o documento está em um estado proposto (*proposed*) ou aprovado (*aproved*).

Esta abordagem centraliza os esforços de controle e desenvolvimento do processo de negociação em um único documento, que além de servir como instrumento de negociação, seria, ao final do processo de interação, o próprio documento de acordo de colaboração.

Para ser visível a todos os parceiros comerciais, o serviço de negociação deve ser publicado no Registro, junto com o CPP da empresa, para que os possíveis parceiros de negócio possam acessar e instalar esta aplicação.

Após as duas partes estarem com a aplicação de negociação instalada, a parte interessada em estabelecer um processo de negócio inicia o processo de negociação.

Para o desenvolvimento do processo de negociação, pode-se restringir os elementos do CPA a serem negociados, àqueles elementos do CPP que são essenciais para se estabelecer a interoperabilidade entre as aplicações dos parceiros [ebXML CPPA, 2002].

Posteriormente, após a estabilidade deste processo, novas negociações poderão incluir elementos adicionais, gerando-se um novo acordo.

Com relação aos documentos elaborados durante o processo de negociação, denominaremos de CPA Inicial à primeira proposta de acordo, que um dos parceiros elabora a partir do seu CPP,do CPP do possível parceiro e do CPA Proposto (as eventuais versões posteriores do CPA Inicial, geradas durante a troca de propostas).

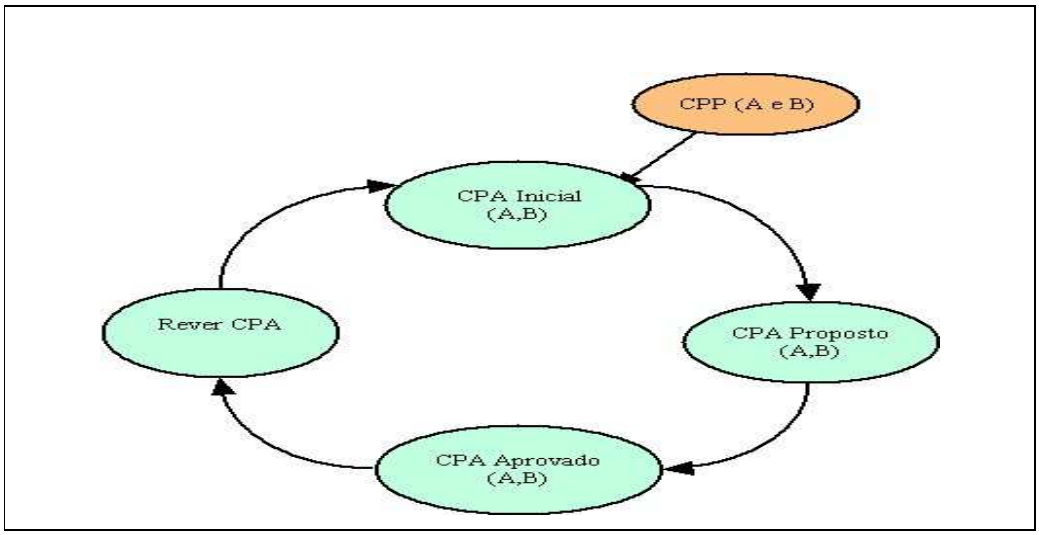

Figura 19: Ciclo de negociação do CPA

A figura 19 mostra conceitualmente o ciclo de negociação do CPA, com a evolução dos documentos que são gerados a partir do envio de um CPA inicial, por uma das partes, a geração posterior de versões intermediárias (CPA Proposto), até ser estabelecida a versão final (CPA Aprovado) do documento.

Caso uma das partes queira adicionar novos elementos, o CPA Aprovado deve ser revisto, reiniciando-se o ciclo de negociação.

4.2.1 Os elementos essenciais de interoperabilidade

Quando uma solução que satisfaça todos os itens de interoperabilidade descrita nos CPP não pode ser alcançada, mas existe o interesse mútuo em estabelecer um processo de negócio, uma das partes pode propor a implementação de um CPA que seja suficiente para satisfazer os requisitos mínimos de interoperabilidade [ebXML CPPA, 2002].

Neste caso, o usuário deve ser solicitado para determinar o que é suficiente para a implementação do acordo, efetuando a configuração dos itens essenciais para estabelecer a interoperabilidade com o outro parceiro.

- Analisando-se o documento CPP e CPA, podemos agrupar conceitualmente os elementos essenciais ao estabelecimento de um acordo de colaboração em duas camadas:Elementos relativos ao processo de negócio, abrangendo as regras e especificações definidas no elemento *ProcessSpecification* e seus elementos filhos.
- Elementos relativos aos protocolos de comunicação, envolvendo o transporte e controle das mensagens trocadas entre os parceiros, definidas nos elementos *DeliveryChannel*, *Transport/TransportSecurity* e *DocExchange*.

No CPA que está sendo negociado, o elemento *ProcessSpecification*, aponta para o Processo de Negócio (*Business Process*) que está sendo proposto.

Neste elemento, os itens essenciais para possibilitar a integração dos processos de negócio estão definidos nos atributos e no elemento filho mostrado na figura 20.

```
<tp:ProcessSpecification tp:version="1.0" tp:name="buySell" xlink:type="simple"
  xlink:href="http://www.ebxml.org/processes/buySell.xml" />


 <tp:Role tp:name="buyer" xlink:type="simple"
  xlink:href="http://ebxml.org/processes/buySell.xml#buyer" />
```

Figura 20: Itens essenciais do elemento ProcessSpecification [ebXML,2002]

Na figura 20 , o atributo *tp:name=”buySell”* indica o processo de negócio que está sendo negociado, referenciado pelo atributo *href,* como um documento externo, onde estão descritas as transações de negócio relativas a este processo.

O elemento *Role tp:name="**buyer**"* indica a regra que o detentor deste CPA está disposto a executar, neste processo de negócio de compra e venda, ou seja, o papel de comprador. O atributo *href* deste elemento indica a “url “onde está o documento relativo a esta regra.

A comparação dos elementos e atributos relativos ao processo de negócio, referenciados nos CPP ou nos CPA que estejam sendo negociados, nem sempre visa encontrar condições de igualdade, pois há itens de complementaridade, como no caso desta regra de negócio, onde um parceiro deverá constar como vendedor e o outro como comprador.

Com relação aos aspectos de comunicação, no elemento *DeliveryChannel* estão definidos itens essenciais para o estabelecimento da conectividade entre os parceiros, com relação a configuração lógica dos canais de comunicação.

```
<tp:DeliveryChannel tp:channelId="N04" tp:transportId="N05"
tp:docExchangeId="N06">
  <tp:Characteristics tp:syncReplyMode="none" tp:nonrepudiationOfOrigin="true"
    tp:nonrepudiationOfReceipt="false" tp:secureTransport="true"
    tp:confidentiality="true" tp:authenticated="true" tp:authorized="false" />
</tp:DeliveryChannel>
 <tp:DeliveryChannel tp:channelId="N07" tp:transportId="N08"
    tp:docExchangeId="N06">
  <tp:Characteristics tp:syncReplyMode="none" tp:nonrepudiationOfOrigin="false"
    tp:nonrepudiationOfReceipt="false" tp:secureTransport="false"
    tp:confidentiality="false" tp:authenticated="false" tp:authorized="false" />
</tp:DeliveryChannel>
```

Figura 21: Itens de comunicação do elemento DeliveryChnnel [ebXML,2002]

No fragmento mostrado na figura 21, o primeiro canal de entrega, identificado como “N04”, está configurado com alguns recursos de segurança, assinalados como “*true*”, enquanto o segundo canal de entrega, identificado como “N07”, não exige estas características, assinaladas como “*false*” nos respectivos elementos.

O elemento *Transport* também possui elementos filhos com definições relativas aos protocolos e respectivas versões, utilizados no transporte das mensagens e essenciais ao estabelecimento da comunicação:

```
<tp:Transport tp:transportId="N05">
  <tp:SendingProtocol tp:version="1.1">HTTP</tp:SendingProtocol>
  <tp:ReceivingProtocol tp:version="1.1">HTTP</tp:ReceivingProtocol>
     <tp:Endpoint tp:uri="https://www.example.com/servlets/ebxmlhandler"
    tp:type="allPurpose" />

  <tp:TransportSecurity>
     <tp:Protocol tp:version="3.0">SSL</tp:Protocol>
     <tp:CertificateRef tp:certId="N03" />
  </tp:TransportSecurity>
</tp:Transport>
```

Figura 22: Definições do Protocolo de transporte [ebXML CPPA,2002]

Neste elemento, opcionalmente também podem ser definidos parâmetros de segurança no elemento filho *TransportSecurity* (figura 22).

Para complementar, se o processo de negócio, que está sendo negociado, envolver a utilização de recursos de segurança, tais como certificados digitais e criptografia de mensagens, o elemento *DocExchange* e seus elementos filhos também terão que ser verificados.

O elemento DocExchange (figura 23) envolve vários outros elementos relacionados à tecnologia de assinatura digital de um documento CPA. No entanto, conforme descrito nas especificações do CPP/CPA, um CPA que está sendo negociado, não necessita ser assinado digitalmente [ebXML CPPA, 2002].

Com relação aos elementos de comunicação e transporte de mensagens, observa-se que a comparação pode ser efetuada buscando-se a igualdade e a compatibilidade de conteúdo destes elementos, quando presentes nos CPP e CPA propostos.

Os demais itens do CPP/CPA, tais como os atributos do elemento *CollaborationProtocolAgreement/* como o *cpaid, version, status, start, end* e *ConverstionConstraint/invocationLimit e concurrentConversations*, podem ter seus valores comparados para verificar a compatibilidade entre as partes.

Os atributos *version* e *status* podem ter seus valores fornecidos pelo próprio mecanismo de controle do sistema de negociação, de acordo com o andamento do processo.

```
<tp:DocExchange tp:docExchangeId="N36">
    <tp:ebXMLBinding tp:version="0.98b">
<tp:ReliableMessaging tp:deliverySemantics="OnceAndOnlyOnce"
tp:idempotency="true" tp:messageOrderSemantics="Guaranteed">
        <tp:Retries>5</tp:Retries>
        <tp:RetryInterval>30</tp:RetryInterval>
        <tp:PersistDuration>P1D</tp:PersistDuration>
</tp:ReliableMessaging>
<tp:NonRepudiation>
  <tp:Protocol>http://www.w3.org/2000/09/xmldsig#</tp:Protocol>
     <tp:HashFunction>http://www.w3.org/2000/09/xmldsig#sha1</tp:HashFunction>
     <tp:SignatureAlgorithm>http://www.w3.org/2000/09/xmldsig#dsa-
     sha1</tp:SignatureAlgorithm>
  <tp:CertificateRef tp:certId="N33" />
</tp:NonRepudiation>
    <tp:DigitalEnvelope>
        <tp:Protocol tp:version="2.0">S/MIME</tp:Protocol>
        <tp:EncryptionAlgorithm>DES-CBC</tp:EncryptionAlgorithm>
        <tp:CertificateRef tp:certId="N33" />
    </tp:DigitalEnvelope>
</tp:ebXMLBinding>
/tp:DocExchange>
```

Figura 23: Parâmetros de segurança do elemento DocExchange [ebXMLCPPA,2002]

4.2.2 A comparação das Propostas de Acordo de Colaboração

A tarefa de comparação das propostas abrange as duas camadas do CPP/CPA:

1 - A comparação do conteúdo dos elementos da camada de especificação do processo de negócio, abrangendo as regras e definições presentes no elemento *ProcessSpecification*, verificando a compatibilidade entre estas especificações.

2- A comparação do conteúdo dos elementos da camada de transporte de mensagens, abrangendo os protocolos e parâmetros de comunicação definidos em cada parte, verificando se há compatibilidade entre estes itens.

Para comparação das propostas, o sistema de negociação deve ser configurado com base no CPP da empresa proprietária do sistema e nas informações fornecidas por um usuário, abrangendo os itens relativos à capacidade técnica e ao processo de negócio que a empresa está disposta a estabelecer com um outro parceiro.

O arquivo de configuração gerado com estas informações passa a ser utilizado internamente pelo sistema de negociação como o CPA atual da empresa.

Neste processo de comparação temos duas situações:

- Efetuar a comparação deste CPA interno com uma proposta (CPA) recebida de um possível parceiro (recebimento de um CPA Inicial ou de um CPA Proposto).
- Efetuar a comparação do CPP de um possível parceiro com os itens definidos neste arquivo de configuração (CPA interno) para gerar uma proposta de acordo a ser encaminhada a este parceiro (envio de um CPA Inicial ou de um CPA Proposto).

Para o desenvolvimento de um sistema que possa efetuar, de forma automatizada, esta comparação de CPA's propostos e gerar um CPA Aprovado ao final do processo de negociação, foram adotados, como premissas, os seguintes aspectos:

- As empresas envolvidas no processo de negociação estão pretendendo desenvolver um processo de colaboração eletrônica, com a integração de seus processos de negócio, conforme as capacidades técnicas descritas em seus CPP's, que estão publicados em um Registro.
- O processo de negociação estará sendo desenvolvido diretamente entre dois parceiros comerciais, sem intermediação de um terceiro.
- O processo de negócio (ebBP) que está sendo negociado é um processo padronizado por uma comunidade eletrônica de empresas, e está publicado e registrado em um Registro ebXML, portanto, acessível a todos os participantes desta comunidade eletrônica.
- A análise de aspectos estratégicos e de interesse de negócio, envolvendo o processo de negociação, será efetuado por um negociador humano, auxiliado pelas informações fornecidas pelo sistema automatizado.

Considerando-se estas premissas, podem ser estabelecidos os requisitos necessários para que um sistema automatizado possa auxiliar no processo de formação de um CPA.

4.2.3 Requisitos Funcionais

Os requisitos necessários para a automatização do processo de geração do CPA, de acordo com os aspectos de interoperabilidade que devem ser contemplados em um processo de colaboração, abrangem o seguinte conjunto de funcionalidades básicas:

- Para a elaboração de propostas, o sistema deve ler o documento de perfil de colaboração (CPP) de cada empresa, para gerar um CPA inicial que será utilizado no processo de negociação.
- A partir do CPP da própria empresa, gerar um CPA local com as capacidades da empresa e apresentá-lo ao usuário para ajustes e configuração. Este CPA local ou interno, representado pelos parâmetros que forem configurados, será utilizado para comparação com as propostas (CPA) recebidas.

- Permitir que o usuário configure os parâmetros de negociação relativos aos itens a serem incluídos ou alterados no CPA, estabelecendo preferências quando houver mais de um valor possível para um mesmo elemento.
- Suportar o papel de iniciador do processo de negociação, através da elaboração e encaminhamento de uma proposta (CPA inicial), ou de receptor, através do recebimento de uma proposta encaminhada por um possível parceiro, iniciador do processo.
- Receber e enviar propostas de acordo de colaboração via Web a outro parceiro comercial, conforme as regras definidas no protocolo de negociação.
- Comparar as propostas de acordo recebidas com os parâmetros do CPA gerado localmente, para verificar as incompatibilidades e apresentá-las ao usuário.
- Controlar o tempo de resposta relativo a uma mensagem de oferta ou de contra-oferta encaminhada a um parceiro, conforme os parâmetros de espera configurados pelo usuário.
- Registrar o tempo de conclusão do processo de negociação com um determinado parceiro.
- Registrar as exceções ocorridas durante o processo, como indisponibilidade de comunicação com o parceiro, por exemplo.
- Gerar propostas e contrapropostas, independentemente do papel (cliente ou provedor) que esteja representando em uma determinada negociação.
- Controlar a versão dos CPA propostos, conforme o andamento das negociações.
- Informar ao usuário o estado atual de uma negociação, quando solicitado ou quando for necessária a intervenção do usuário para a definição de uma determinada negociação.
- O sistema deve manter um histórico das negociações para consulta.
- O sistema deverá disponibilizar o CPA resultante do processo de negociação para uma aplicação de negócio, que controlará a interação comercial (fase de transações), após a finalização com sucesso de um processo de negociação.

Além destas funcionalidades, é necessária a definição de um protocolo básico de negociação, abrangendo as mensagens a serem utilizadas na interação entre os parceiros durante o processo de negociação.

4.2.4 Protocolo de Mensagens para Negociação

Para que os parceiros possam interagir, encaminhando as propostas de acordo durante o processo de negociação, é necessário que seja definido um conjunto básico de mensagens padronizadas, que possam ser utilizadas para possibilitar as trocas de CPA e o próprio processo de negociação.

Para a especificação de um protocolo de negociação, Law propõe uma abordagem na qual o protocolo deve definir as atividades permitidas em um certo estado do processo de negociação e as regras de como mudar este estado, dependendo das atividades executadas [Law *et.al*, 2005].

Estas atividades envolvem o envio de mensagens de ofertas e contra-ofertas, e o respectivo processamento destas mensagens por um sistema de negociação, que entende e executa estas atividades conforme regras explícitas que devem ser seguidas por ambas as partes.

Conforme Strobel, a comunicação utilizada em processos de Negociações Eletrônicas, deve assegurar que ambas as partes tenham o mesmo entendimento com relação aos itens que estão sendo negociados, especificando um vocabulário comum que facilite a interação entre as partes [Strobel, 2001.

Neste contexto, a partir de uma perspectiva sintática e semântica, Strobel propõe a utilização de esquemas XML como o mecanismo ideal para representação de um conjunto de regras e mensagens padronizadas, a serem utilizadas durante o processo de negociação.

Conforme o tipo de mensagem enviada ou recebida, o sistema de negociação deve executar uma lógica apropriada para o tratamento desta mensagem.

Com base nestas definições e analisando-se as interações necessárias para geração de um CPA, um conjunto de mensagens básicas foi elaborado para suportar o processo de negociação, abrangendo os seguintes tipos de mensagens:

- Tipo 1 - Oferta
- Tipo 2 – Contra-Oferta
- Tipo 3 – Confirmação de Recebimento
- Tipo 4 – Recusa de Oferta
- Tipo 5 – Aceite de Oferta
- Tipo 6 – Finalização da Negociação

A troca de mensagens deve seguir um fluxo lógico de mensagens, como por exemplo, a seqüência mostrada na figura 24.

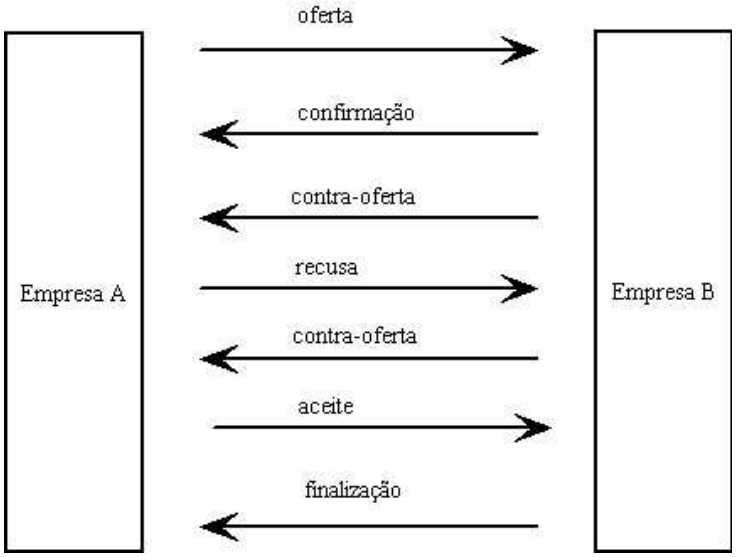

Figura 24: Fluxo de mensagens do protocolo de negociação

Para simplificar o desenho do fluxo de mensagens, não mostramos todas as mensagens de confirmação de recebimento, que são enviadas como resposta a cada envio de uma das mensagens descritas na figura 24.

Este protocolo de mensagens utiliza um modelo de mensagens análogo ao conjunto de mensagens utilizado no ebBP para a fase de transações de negócio [eBP,2006], onde temos, por exemplo, um fluxo de mensagens de *Requesting, Response, AcknowledgeReceipt e AcknowledgeAcceptance* para o desenvolvimento das atividades de negócio.

Com relação ao fluxo da figura 24, a mensagem de oferta deve possuir, como anexo, um CPA Inicial, indicando o estabelecimento do estado inicial do processo de negociação.

Esta mensagem deve ser seguida por uma mensagem de confirmação de recebimento de mensagem: uma mensagem de contra oferta pode ser enviada na sequência, tendo como anexo um CPA Proposto, indicando que o outro parceiro está disposto a negociar.

Neste ponto, o processo de negociação entra em um estado intermediário para elaboração do acordo, onde as partes podem vir a trocar várias mensagens de contra-oferta.

A mensagem de recusa de oferta indica que um parceiro não aceitou uma oferta, mas não deseja efetuar uma contra-oferta e nem finalizar a negociação, entrando em um estado de espera por uma nova proposta.

Neste caso o tempo de espera deverá ser controlado pela aplicação de negociação, até um limite máximo estipulado pelo usuário, podendo a negociação ser encerrada com o envio de uma mensagem de finalização, após a expiração do tempo estipulado.

As mensagens de aceite de oferta e finalização da negociação não possuem anexo e indicam respectivamente que o processo de negociação entrou em um estado de acordo, com o CPA Aprovado, e em um estado de finalização, com o encerramento do processo.

Para definir a estrutura e as regras de formação destas mensagens e possibilitar o controle durante o processo de interação, foi elaborado o esquema XML representado graficamente na figura 25.

Neste esquema, estão definidos os elementos e atributos que podem ser utilizados na elaboração das mensagens de oferta, contra-oferta, confirmaRecebimento, recusaOferta, aceitaOferta e finalizaNegociacao, que compreendem o conjunto básico de mensagens a ser utilizadas no processo de interação.

O elemento raiz *msgsNegociacao* possui o atributo *nrNegociacao* que deve receber um numero seqüencial, a ser utilizado pelo sistema para controle da negociação.

O elemento filho mensagem possui o atributo *nrSequencia*, para controle das mensagens e permite a escolha de um tipo de mensagem a ser enviada.

Cada uma das mensagens definidas possui o atributo *tipo*, com um valor definido para possibilitar que o sistema identifique uma mensagem por este atributo.

A estrutura definida para todas as mensagens abrange os elementos *origem* e *destino*, que devem ter como elementos filhos os elementos *empresa, endereco e url*, para possibilitar o controle e a identificação do emitente e do recebedor da mensagem.

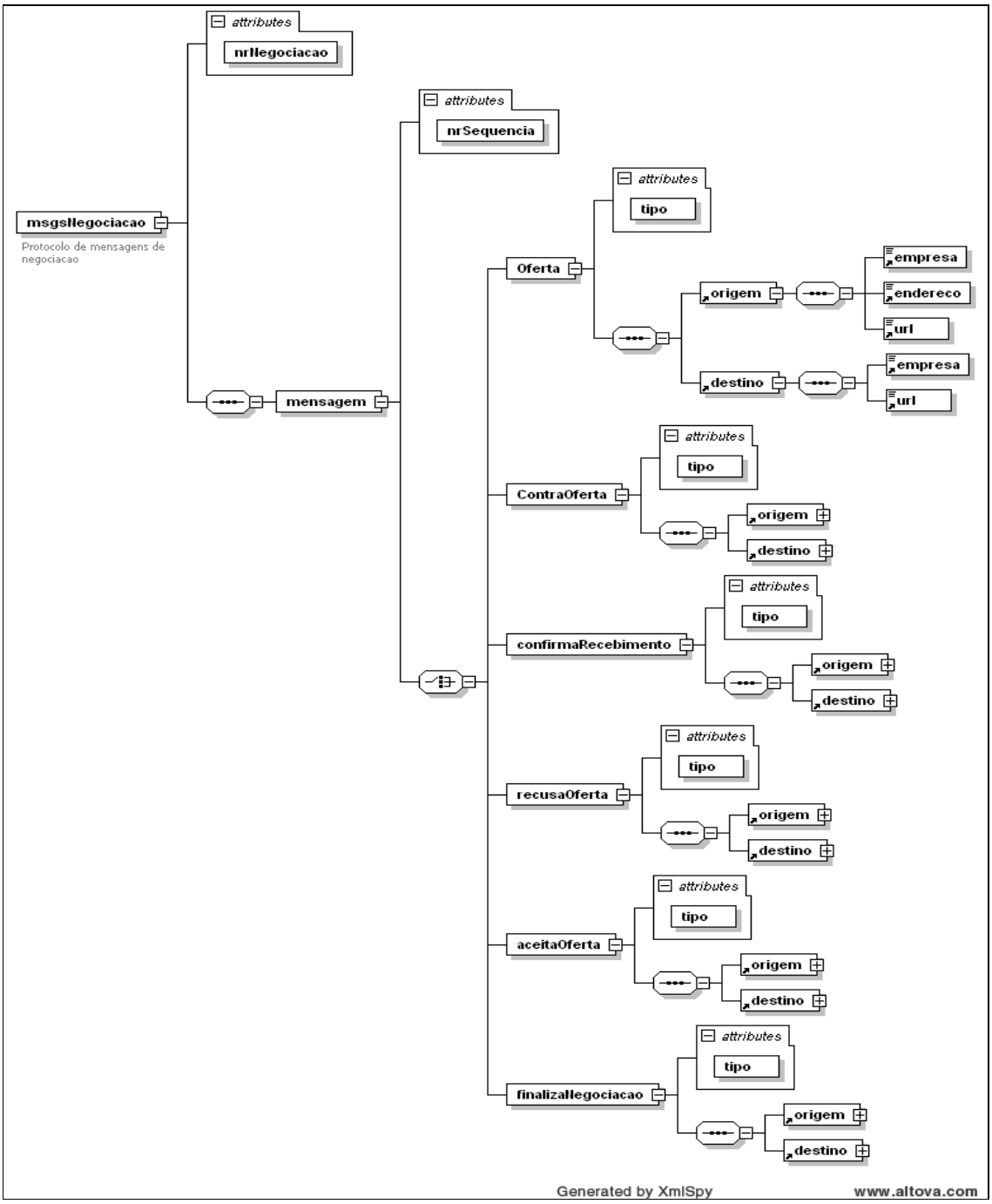

Figura 25: Modelo gráfico do esquema XML do protocolo de mensagens

Os documentos de mensagem utilizados durante o processo de negociação deverão ser construídos e validados de acordo com as regras definidas neste esquema.

No próximo capítulo, abordamos a implementação e a codificação deste esquema em XML.

4.2.5 O Sistema de Negociação ebXML

O sistema de negociação eletrônica deverá efetuar a negociação via web das possíveis condições de um acordo de colaboração, conforme o perfil de negócio (CPP) dos parceiros comerciais, gerando de forma automatizada o acordo de colaboração (CPA), para ser avaliado por um negociador humano.

A decisão referente ao desenvolvimento de um determinado processo de negociação, assim como a configuração inicial do sistema com relação aos parâmetros negociáveis, deverá ser efetuada por este negociador humano, subsidiado pelas informações disponibilizadas pelo sistema.

O sistema poderá ser disponibilizado para os parceiros comerciais a partir de um Registro, quando os parceiros efetuarem a baixa dos CPP's de eventuais parceiros comerciais.

Neste sentido, o sistema seria similar a um processo de negócio de negociação, que estaria disponível no Registro ebXML para que qualquer empresa pudesse baixá-lo e utilizá-lo para iniciar processos de negociação com os demais parceiros.

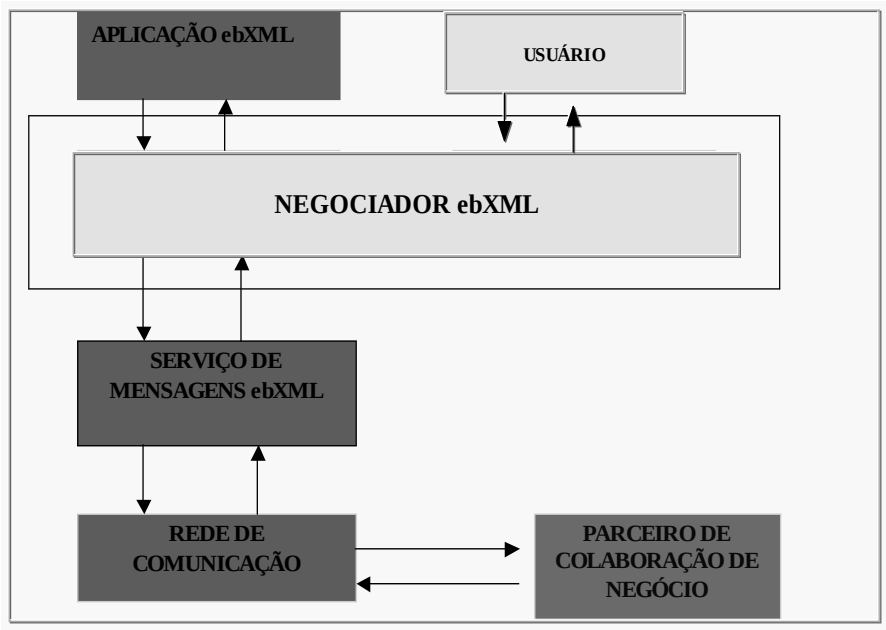

Figura 26: Fluxo de interação do sistema com o ambiente ebXML

A figura 26 mostra o fluxo de interação do sistema de negociação em um ambiente de aplicações ebXML.

O sistema deve efetuar o controle do processo de negociação, atuando entre o serviço de mensagens e a aplicação ebXML de negócios, durante o desenvolvimento do processo de negociação do CPA, de acordo com as configurações efetuadas pelo usuário.

Na elaboração de uma proposta, a aplicação de negócio ebXML disponibilizará o CPP do possível parceiro ao sistema de negociação, gerando uma mensagem para uma interface do sistema, após disponibilizar este documento em um diretório configurado pelo usuário, de onde o sistema efetuará a leitura deste documento.

Após a extração das informações do CPP, é efetuada a comparação com as capacidades da empresa local configuradas pelo usuário e gerado um CPA inicial a ser encaminhado ao possível parceiro, através de uma mensagem de oferta.

Esta mensagem é disponibilizada para o serviço de mensagens, juntamente com o CPA gerado, para ser encapsulada em uma mensagem SOAP e transportada através da rede de comunicação até o parceiro a que se destina.

No sentido inverso, uma mensagem é encaminhada por um parceiro e entregue ao sistema de negociação pelo serviço de mensagens para ser processada.

O Negociador ebXML efetuará a análise da mensagem e, no caso de ser uma proposta ou uma contra-proposta, efetuará a comparação com as condições da empresa local e apresentará o resultado ao usuário, para que este tome uma decisão com relação à negociação em curso.

Após o fechamento de um acordo de colaboração com um parceiro comercial, o sistema disponibilizará o CPA acordado à aplicação de negócio ebXML, gerando uma mensagem para esta aplicação.

O usuário deverá configurar o sistema com relação a informar ou não à aplicação de negócio quando uma negociação for finalizada sem a elaboração de um acordo.

4.3 A Modelagem do Sistema

Para a modelagem do sistema de negociação foi utilizado o padrão UML (*Unified Modeling Language*), envolvendo a elaboração dos diagramas de Caso de Uso e de Seqüência.

Os diagramas de Seqüência e de Classes são abordados no próximo capítulo, juntamente com os aspectos de implementação do protótipo.

4.3.1 O Diagrama de Casos de Uso

Conforme [Flower, 2004 p.104] o diagrama de caso de uso é uma técnica utilizada para descrever as interações entre os usuários de um sistema e o próprio sistema, fornecendo uma descrição do comportamento do sistema do ponto de vista do usuário.

A notação básica do diagrama utiliza os Atores, para representar os usuários do sistema, e para representar os casos de uso utiliza elipses, com o nome do caso de uso.

Com base nos requisitos funcionais levantados, foram identificados os principais casos de uso do sistema, através da interação com os atores Parceiro Comercial, Usuário e Aplicação Comercial ebXML(figura 27).

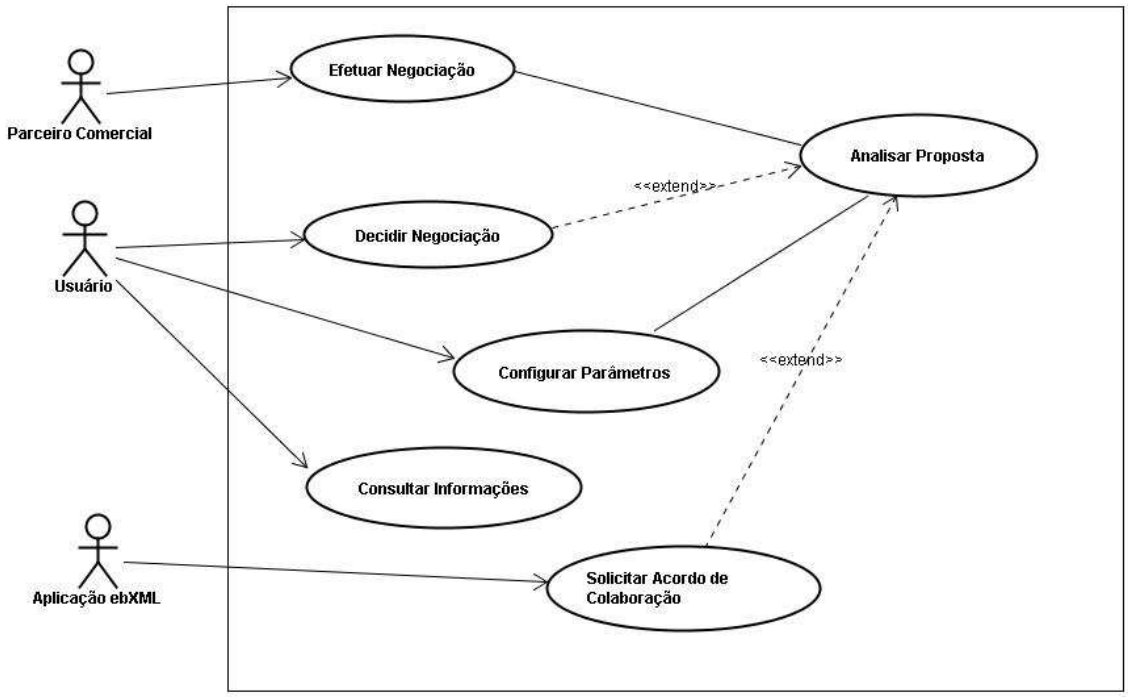

Figura 27: Diagrama de Casos de Uso

Este diagrama de caso de uso, mostra uma visão geral do sistema proposto.

1. **O caso de uso Efetuar Negociação**:
   - Este caso de uso abrange a parte de interação de um parceiro comercial com o sistema, e inicia-se quando um parceiro comercial envia uma mensagem de oferta, com uma proposta de acordo (CPA) para ser avaliada.
   - O sistema deve receber a mensagem e efetuar o processamento da proposta, acionando o caso de uso Elaborar Proposta.
   - Neste processamento, o sistema pode aceitar, recusar, finalizar ou efetuar uma contra proposta.
   - Após o processamento, o sistema deve enviar uma mensagem de volta ao parceiro comercial, com o posicionamento em relação à proposta recebida.
   - O parceiro comercial pode enviar uma outra mensagem, continuando o processo de negociação ou pode encerrar o processo, encerrando este caso de uso.
2. **O caso de uso Analisar Proposta**:
   - É iniciado pelo caso de uso Efetuar Negociação, quando é recebido pelo sistema uma proposta (CPA) para análise.
   - Também pode ser iniciado pelo caso de uso Solicitar Acordo de Colaboração, a partir da solicitação da aplicação de negócio, que encaminha um CPP para elaboração de uma proposta (CPA) a um possível parceiro.

- Abrange funcionalidades internas do sistema para análise de CPP e CPA recebidos e não tem interação com atores externos.

3. Caso de uso Decidir Negociação

- Este caso de uso é uma extensão do caso de uso Analisar Proposta e inicia-se somente quando há uma proposta para ser apresentada ao usuário.
- O usuário efetua a análise da proposta e toma a decisão, configurando itens locais para a continuação ou optando pela finalização do processo de negociação.
- O sistema recebe esta decisão e efetua o processamento da proposta de acordo com a configuração recebida.

4. Caso de uso Configurar Parâmetros

- Inicia-se quando o usuário efetua a configuração inicial do sistema, com as condições iniciais que a empresa está disposta a suportar em um processo de colaboração.
- O sistema deve utilizar esta configuração inicial para analisar eventuais propostas (CPP e CPA) recebidas.

5. Caso de Uso Consultar Informações

- Este caso de uso se incia quando o usuário solicitar informações sobre algum processo de negociação
- O sistema deve apresentar para o usuário opções de consulta ao histórico de negociações efetuadas
- O usuário escolhe qual tipo de consulta deseja efetuar
- O sistema busca as informações em um banco de dados e as apresenta ao usuário.
- O usuário pode efetuar uma nova solicitação ou encerrar a consulta.

6. Caso de uso Solicitar Acordo de Colaboração

- Este caso de uso é uma extensão do caso de uso Analisar Proposta e somente é iniciado quando o sistema recebe da aplicação de negócio o documento XML de perfil (CPP) de um possível parceiro comercial, para o qual a empresa deseja encaminhar uma proposta (CPA), visando estabelecer um processo de colaboração comercial.
- Com base neste CPP o sistema deve elaborar uma proposta (CPA) para ser negociada com o possível parceiro.
- Após o processo de negociação se encerrar, o sistema deve enviar à aplicação de negócio o resultado da negociação, disponibilizando o documento de acordo (CPA) caso este tenha sido elaborado.

4.4 A Arquitetura do Sistema de Negociação

A arquitetura do Sistema de Negociação ebXML, elaborada com base nos requistos e casos de uso do sistema, pode ser visualizada na figura 28, onde cada bloco representa um conjunto de funcionalidades a serem executadas pelo sistema:

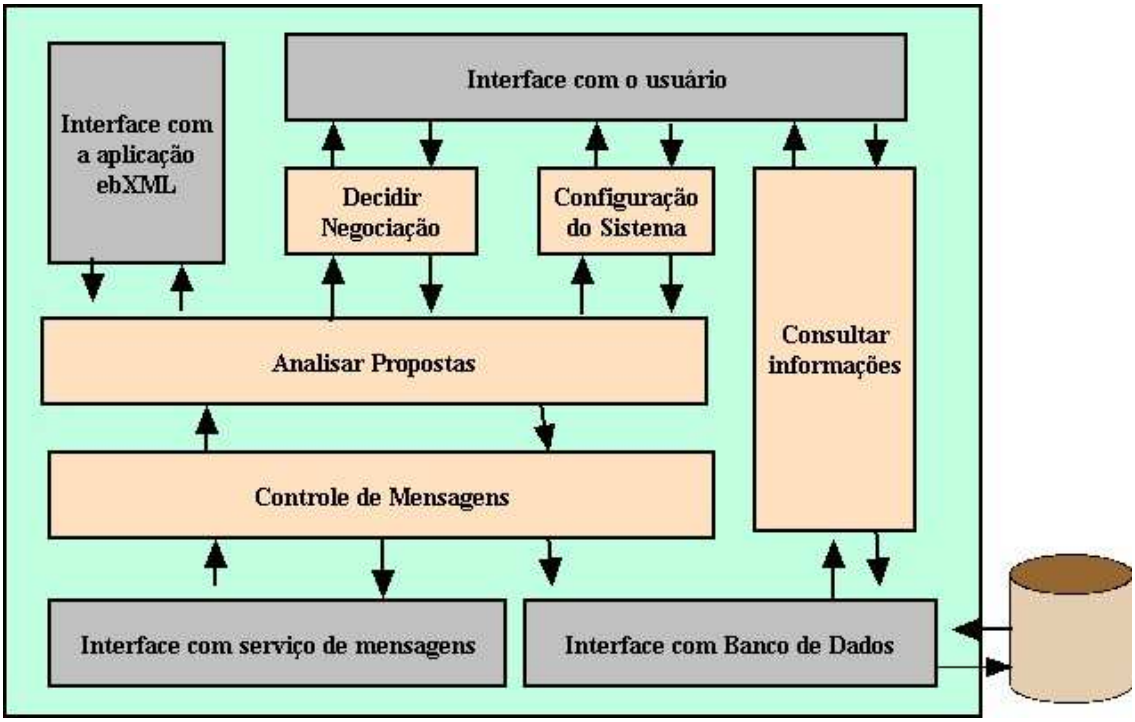

Figura 28: Arquitetura do Sistema

Os módulos de interface do sistema abrangem interfaces com a aplicação ebXML, com o usuário, com o serviço de mensagens e com o sistema gerenciador do Banco de Dados.

Os módulos de processamento interno do sistema consistem do módulo de controle de mensagens, analisar propostas, decidir negociação, configuração do sistema e o módulo de consultar informações.

O módulo de controle de mensagens efetua o controle do processo de interação, de acordo com as mensagens definidas no protocolo de negociação, utilizando a interface com o serviço de mensagens, que pode ser o ebMS, por exemplo, para controlar o envio e recebimento de mensagens com os parceiros de negócio.

Este módulo também tem interação com o banco de dados para gravação de mensagens e informações relativas ao histórico de controle de uma negociação.

A configuração de parâmetros do sistema é efetuada pelo módulo de Configuração do Sistema, que pode ser acionado tanto pelo módulo de análise de propostas, quanto pelo usuário.

O módulo de análise de propostas efetua a leitura e o controle das propostas, na forma de CPA ou de CPP, efetuando o controle das versões dos documentos gerados e dos documentos recebidos para análise.

A decisão sobre as mudanças a serem efetuadas em um CPA, visando estabelecer um acordo de colaboração, assim como a decisão final sobre a finalização de um processo de negociação, deve ser efetuada por um usuário, através do módulo Decidir Negociação.

Neste módulo são verificadas as incompatibilidades entre as propostas de acordo, podendo ser geradas novas propostas, conforme a decisão do usuário.

A consulta às informações de histórico das negociações, poderão ser consultadas pelo usuário através do módulo de consulta às informações, que interage com a interface do banco de dados para disponibilizar as informacões solicitadas.

A estas funções básicas poderão ser agregadas posteriormente outras funções complementares, conforme as necessidades de gerenciamento de um processo de negociação.

4.5 Considerações Finais

Neste capitulo, efetuamos a discussão dos requisitos funcionais para o desenvolvimento de um sistema automatizado que possa auxiliar um usuário a gerar o CPA e possibilitar também a interação entre os parceiros comerciais, durante a fase de negociação.

Para isto, foram definidos a arquitetura do sistema, um protocolo básico de mensagens de negociação e a modelagem dos aspectos funcionais que um sistema de negociação necessita para a geração automatizada do acordo de colaboração (CPA), através do diagrama de caso de uso.

No próximo capítulo, descreveremos a implementação de um protótipo, desenvolvido com base nos requisitos discutidos neste capítulo.

5. A IMPLEMENTAÇÃO DO PROTÓTIPO

Neste capítulo, será abordada a implementação do protótipo baseado nos requisitos levantados no capitulo anterior, assim como a descrição do cenário de testes realizados e os respectivos resultados obtidos.

5.1. O sistema Negociador ebXML

Para a validação prática da modelagem e considerações técnicas discutidas no capitulo anterior, foi desenvolvido um protótipo para a implementação de um protocolo de negociação para a automatização do processo de interação e a implementação de funcionalidades básicas de análise e geração de um CPA.

Para o desenvovimento deste protótipo, consideramos inicialmente somente os requisitos básicos, para possibilitar a avaliação prática da proposta.

Estas funcionalidades básicas abrangem a transmissão e recebimento de mensagens e a comparação dos CPA com a geração de um novo CPA (contra-oferta).

Para o processamento dos documentos XML (CPA e CPP) foi utilizado a API JAXP (*Java API for XML Processing*) da SUN que implementa a especificação do padrão DOM [W3C DOM, 2004].

Esta API foi escolhida por ser uma API de uso livre, fornecida com o conjunto de ferramentas de desenvolvimento Java (JDK), e possuir diversos métodos para manipulação de elementos e atributos de um documento XML.

Para o serviço de mensagens, foi utilizado o software livre padrão ebXML, ebMS Hermes, disponibilizado pelo freebxml.org.

5.1.1 A API DOM

A especificação DOM (*Document Object Model*) definida pelo W3C é uma interface independente de linguagem e plataforma, que especifica um modelo de objetos em árvore, para acesso e manipulação de dados em documentos XML e HTML [W3C, DOM, 2004].

As API que implementam este modelo criam em memória uma hierarquia de objetos de um documento (figura 29), possibilitando a navegação nessa estrutura, para adicionar, modificar ou apagar elementos e seu conteúdo.

Para isto, a API disponibiliza diversos métodos que podem ser utilizados para acessar os nós, tais como *getFirstChild, getLastChild, getNextSibling, getPreviousSIbling e getParentNode*, que possibilitam navegar por uma árvore DOM, a partir de uma determinada localização na árvore.

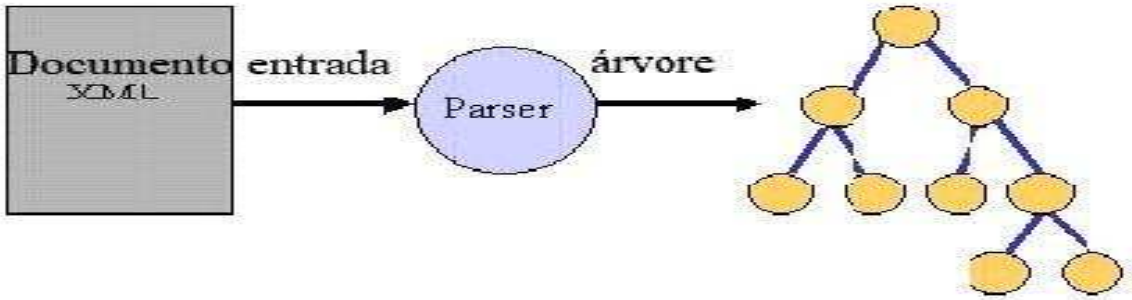

Figura 29: O modelo DOM [W3C DOM, 2004f]

A API JAXP (*Java API for XML Processing*) é disponibilizada pela SUN, podendo ser acessada em www.sun.com.

5.2 O Serviço de Mensagens

Para o transporte das mensagens neste protótipo, utilizamos o produto ebMS Hermes, um software livre que implementa o serviço de mensagens ebXML, disponibilizado à comunidade pelo freebXML.org (http://www.freebxml.org).

A organização freebXML é composta pelo governo da China e pela SUN, com a participação de algumas outras empresas e tem como finalidade desenvolver aplicações baseadas nas especificações do ebXML.

O sistema foi desenvolvido pela Universidade de Hong Kong, que mantém um centro de pesquisa e desenvolvimento em comércio eletrônico (CECID – http://www.cecid.hku.hk).

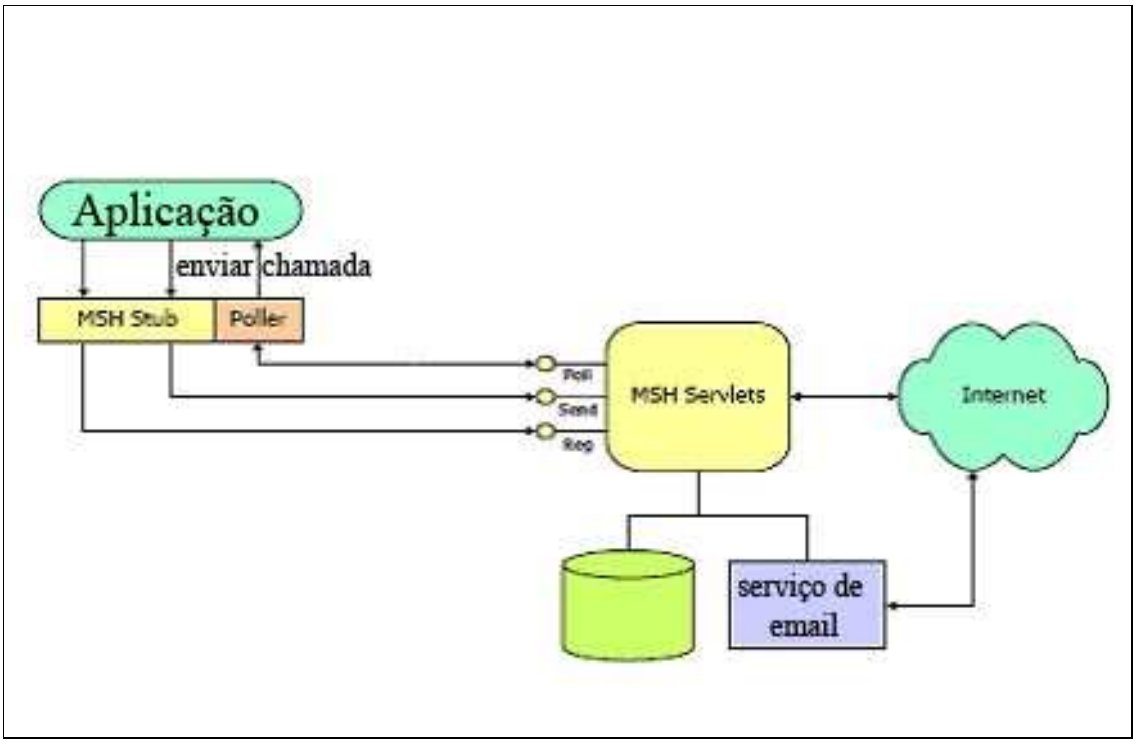

Figura 30: Arquitetura do ebMS Hermes [CECID, 2002]

A documentação do produto, que é baixada juntamente com o arquivo compactado do próprio software, esclarece as ações necessárias de configuração e instalação, assim como a forma de integração com sistemas do ambiente onde será instalado.

O ebMS Hermes utiliza um módulo cliente, o qual é formado por um componente MSH Stub e MSH Poller, e um módulo servidor, que consiste de um servlet (MSH Servlet) para tratar as requisições e interações com o cliente ebMS, conforme descrito na figura 30.

A aplicação cliente, através do MSH Stub, inicialmente efetua um registro no MSH Servlet, informando sua url, para poder habilitar a transmissão e recebimento de mensagens.

Para a transmissão de mensagens, a aplicação efetua chamadas ao método *send( )* do objeto *Request* do MSH Stub, que efetua o processo de empacotamento da mensagem no protocolo SOAP e transfere a mensagem para o MSH servlet, que efetua a transmissão para o destino.

Na recepção de mensagens, o MSH servlet pode ser configurado para transferir a mensagem para um local específico em disco. Periodicamente, o componente MSH Poler, efetua uma solicitação (*polling*) a este servlet, para verificação de chegada de mensagens e posterior chamada ao método da aplicação, para transferência da mensagem recebida.

A classe Controle de Mensagens do Negociador ebXML é quem efetua a interação com esta interface do serviço de mensagens, através dos métodos lerMensagem e receberMensagem.

Na recepção das mensagens, o ebMS foi configurado para gravar a mensagem recebida em disco, na pasta CpaRx, gerando uma mensagem para a aplicação de negociação.

5.3 A implementação do Protocolo de Negociação

Conforme abordado no capitulo anterior, o protocolo de mensagens de negociação abrange as mensagens de oferta, confirmação de recebimento, contra-oferta, recusa de oferta, aceite de oferta e finalização da negociação.

O esquema XML que define a estrutura destas mensagens será utilizado pelo módulo de controle de mensagens do sistema de negociação, para verificar a validade das mensagens trocadas entre os parceiros, durante o processo de negociação.

Conforme as definições abordadas no capitulo anterior, foi criado o esquema XML da figura 31, para a implementação do protocolo de mensagens de negociação.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XMLSpy v2006 3 sp2 (http://www.altova.com) by Adilson Silva (AJS) -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified" attributeFormDefault="unqualified">
      <xs:element name="msgsNegociacao">
        <xs:annotation>
          <xs:documentation>Protocolo de mensagens de negociacao</xs:documentation>
        </xs:annotation>
        <xs:complexType>
          <xs:sequence>
            <xs:element name="mensagem">
                  <xs:complexType>
                        <xs:choice>
                              <xs:element name="Oferta">
                                <xs:complexType>
                                    <xs:sequence>
                                      <xs:element ref="origem"/>
                                      <xs:element ref="destino"/>
                                    </xs:sequence>
                                    <xs:attribute name="tipo"
                                    type="xs:positiveInteger" use="required"
                                    fixed="1"/>
                                </xs:complexType>
                              </xs:element>
                              <xs:element name="ContraOferta">
                                <xs:complexType>
                                    <xs:sequence>
                                      <xs:element ref="origem"/>
                                      <xs:element ref="destino"/>
                                    </xs:sequence>
                                      <xs:attribute name="tipo"
                                    type="xs:positiveInteger" use="required"
                                    fixed="2"/>
                                </xs:complexType>
                              </xs:element>
                              <xs:element name="confirmaRecebimento">
                                <xs:complexType>
                                    <xs:sequence>
                                      <xs:element ref="origem"/>
                                      <xs:element ref="destino"/>
                                    </xs:sequence>
                                      <xs:attribute name="tipo"
                                      type="xs:positiveInteger" use="required"
                                      fixed="3"/>
                                </xs:complexType>
                              </xs:element>
                              <xs:element name="recusaOferta">
```

```
                    <xs:complexType>
                     <xs:sequence>
                        <xs:element ref="origem"/>
                        <xs:element ref="destino"/>
                     </xs:sequence>
                     <xs:attribute name="tipo"
                    type="xs:positiveInteger" use="required"
                    fixed="4"/>
                    </xs:complexType>
                </xs:element>
                <xs:element name="aceitaOferta">
                        <xs:complexType>
                           <xs:sequence>
                                <xs:element ref="origem"/>
                                <xs:element ref="destino"/>
                           </xs:sequence>
                           <xs:attribute name="tipo"
                           type="xs:positiveInteger" use="required"
                           fixed="5"/>
                        </xs:complexType>
                </xs:element>
                <xs:element name="finalizaNegociacao">
                        <xs:complexType>
                           <xs:sequence>
                                <xs:element ref="origem"/>
                                <xs:element ref="destino"/>
                           </xs:sequence>
                           <xs:attribute name="tipo"
                           type="xs:positiveInteger" use="required"
                           fixed="6"/>
                        </xs:complexType>
                           </xs:element>
            </xs:choice>
             <xs:attribute name="nrSequencia"
             type="xs:positiveInteger" use="required"/>
        </xs:complexType>
    </xs:element>
  </xs:sequence>
  <xs:attribute name="nrNegociacao" type="xs:positiveInteger"
  use="required"/>
 </xs:complexType>
</xs:element>
<xs:element name="origem">
   <xs:complexType>
     <xs:sequence>
          <xs:element ref="empresa"/>
          <xs:element ref="endereco"/>
```

```
                    <xs:element ref="url"/>
                </xs:sequence>
            </xs:complexType>
        </xs:element>
        <xs:element name="empresa" type="xs:string"/>
        <xs:element name="endereco" type="xs:string"/>
        <xs:element name="url" type="xs:anyURI"/>
        <xs:element name="destino">
            <xs:complexType>
                <xs:sequence>
                    <xs:element ref="empresa"/>
                    <xs:element ref="url"/>
                </xs:sequence>
            </xs:complexType>
    </xs:element>
</xs:schema>
```

Figura 31: Documento do Esquema XML do Protocolo de Mensagens

O esquema XML do protocolo de mensagens (figura 31) possui dentro do elemento raiz msgsNegociacao, a definição do elemento Mensagem, abrangendo a definição dos elementos filhos Oferta, ContraOferta, ConfirmaRecebimento, RecusaOferta, AceitaOferta e FinalizaNegociacao.

O elemento Mensagem contém o atributo *choice*, indicando que deve ser escolhido apenas um dos elementos filho definidos abaixo dele, para compor uma determinada mensagem, ou seja, em uma mesma mensagem não é permitido ter um elemento Oferta e outro elemento ContraOferta, por exemplo.

A definição das mensagens abrange a referência aos elementos origem e destino, definidos no final do esquema, e o atributo tipo, que é utilizado pelo sistema de negociação, para identificar o tipo de mensagem.

No elemento Mensagem está definida o atributo nrsequencia, para controle do número de seqüência das mensagens pelo sistema de negociação.

O atributo nrNegociacao, que está definido dentro do elemento raiz (msgsNegociacao), será utilizado pelo sistema de negociaçã,o para que possa efetuar o controle de uma determinada negociação.

No final do esquema, estão definidos os elementos empresa, endereço e url, que compõem o elemento origem e o elemento destino, que são utilizados por referência, em cada tipo de mensagem.

A classe Mensagens do sistema de negociação foi elaborada conforme os elementos descritos neste esquema, para a implementação dos objetos relativos a cada uma destas mensagens.

As mensagens devem ser construídas de acordo com este esquema, que deve ser referenciado nas declarações dos elementos XML da mensagem, como no exemplo da mensagem de contra-oferta da figura 32.

```
<?xml version="1.0" encoding="UTF-8"?>
<msgsNegociacao xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation=
"C:\EsquemaXMLtese06\esquema_msgsNegociacao_V5.xsd" nrNegociacao="001">
    <mensagem nrSequencia="002">
        <ContraOferta tipo="2">
            <origem>
                <empresa>Empresa B</empresa>
                <endereco>Rua Inteiro s/n</endereco>
                <url>http://www.empresab.com.br</url>
            </origem>
            <destino>
                <empresa>Empresa A</empresa>
                <url>http://www.empresaa.com.br</url>
            </destino>
        </ContraOferta>
    </mensagem>
</msgsNegociacao>
```

Figura 32 : Mensagem de Contra-Oferta

Esta mensagem é transferida juntamente com o CPA ao serviço de mensagens, para transmissão à url de destino indicada na mensagem.

5.4 O Diagrama de Sequência

Para identificação dos objetos que deverão compor o sistema e detalhamento da seqüência de interação entre estes objetos, elaboramos o diagrama de seqüência para o cenário de recebimento de uma proposta (CPA), o principal cenário envolvendo os casos de uso levantados.

Este cenário descreve o recebimento de uma mensagem de oferta de um parceiro comercial e envio de uma proposta da empresa para um parceiro comercial.

O diagrama de seqüência da figura 33 descreve a seqüência de mensagens trocadas pelos objetos, durante o desenvolvimento deste cenário.

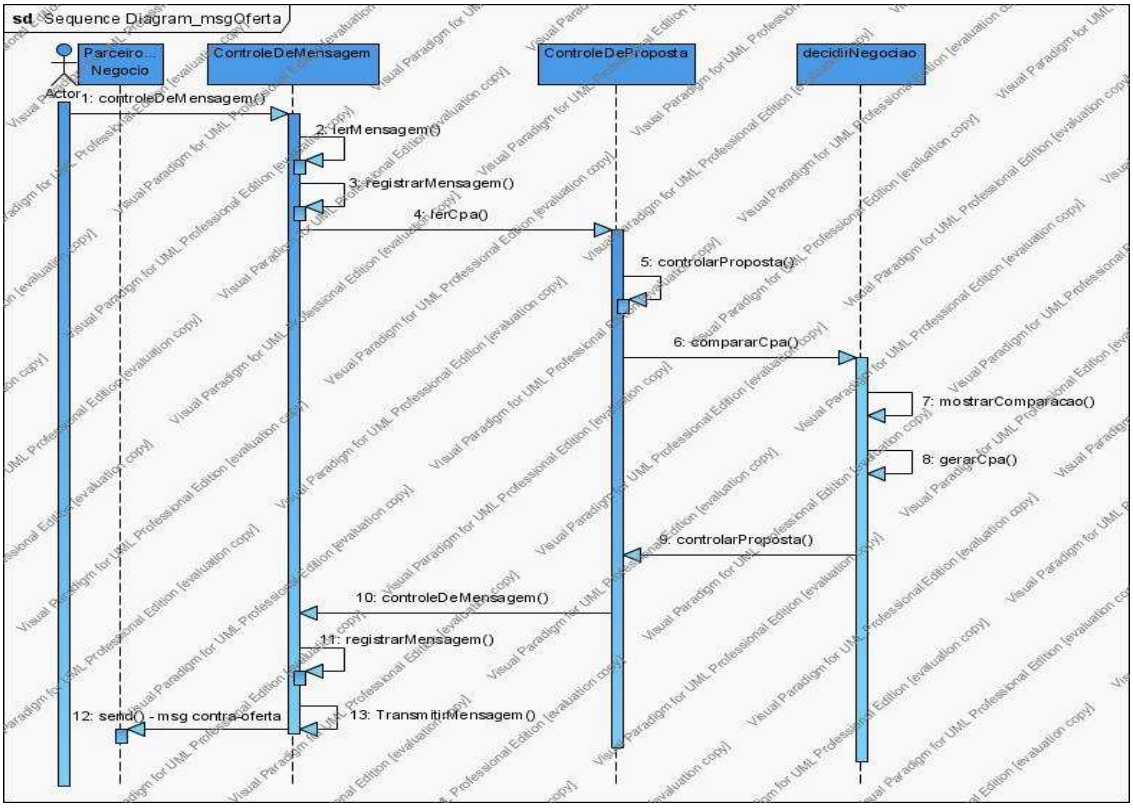

Figura 33: O diagrama de seqüência para a mensagem de oferta recebida

Neste diagrama, uma mensagem de oferta é recebida pelo sistema, através do método controleDeMensagem, da classe ControleDeMensagem.

O método lerMensagem efetua o "parsing" da mensagem, efetuando sua validação de acordo com o esquema do protocolo de mensagens, retornando a mensagem como parâmetro, para o método controleDeMensagem.

O método controleDeMensagem identifica o tipo de mensagem como uma mensagem de oferta e efetua a chamada ao método lerCpa da classe ControleDeProposta, que implementa as funcionalidades do módulo de Análise de Propostas, conforme a arquitetura do sistema (figura 28).

Este método efeturá a leitura do CPA recebido utilizando a API JAXP, gerando o documento DOM relativo a este CPA, que será utilizado para comparação com a configuração interna do sistema, que é o CPA interno gerado quando o usuário efetua a configuração inicial do sistema.

Na classe DecidirNegociacao, o método compararCpa vai ser acionado para ler este documento e comparar os elementos e atributos desta proposta com os elementos e atributos do CPA interno da empresa, acionando o método mostrarComparacao para apresentar estes dados ao usuário.

Os dados são apresentados ao usuário, com o resultado da comparação de cada elemento da proposta recebida com o respectivo elemento interno do sistema, indicando os elementos com conteúdo divergente.

Após a avaliação e as alterações efetuadas pelo usuário, o método gerarCpa é acionado, para que um novo CPA seja gerado, para ser enviado ao parceiro comercial juntamente com uma mensagem de contra-oferta.

O método controlarProposta é chamado, para que efetue o registro e o respectivo controle da nova versão do CPA que foi gerada.

Este método acionará o método controleDeMensagem passando como parâmetro o documento gerado e a informação do número de identificação da negociação, para que o controleDeMensagem possa identificar o parceiro de destino da mensagem.

Após montar a mensagem, controleDeMensagem aciona o método registrarMensagem para registrar o envio desta mensagem de oferta, e chama o método transmitirMensagem, para que este possa chamar o método *send* do serviço de mensagens, para a transmissão da mensgem de oferta e o CPA como anexo.

Com base nesta seqüência de relacionamento entre os objetos, elabora-se o diagrama de classes, para a definição da estrutura do sistema.

5.5 Diagrama de Classes

O diagrama de classes da figura 34 abrange as classes básicas do sistema, essenciais para a implementação do protótipo.

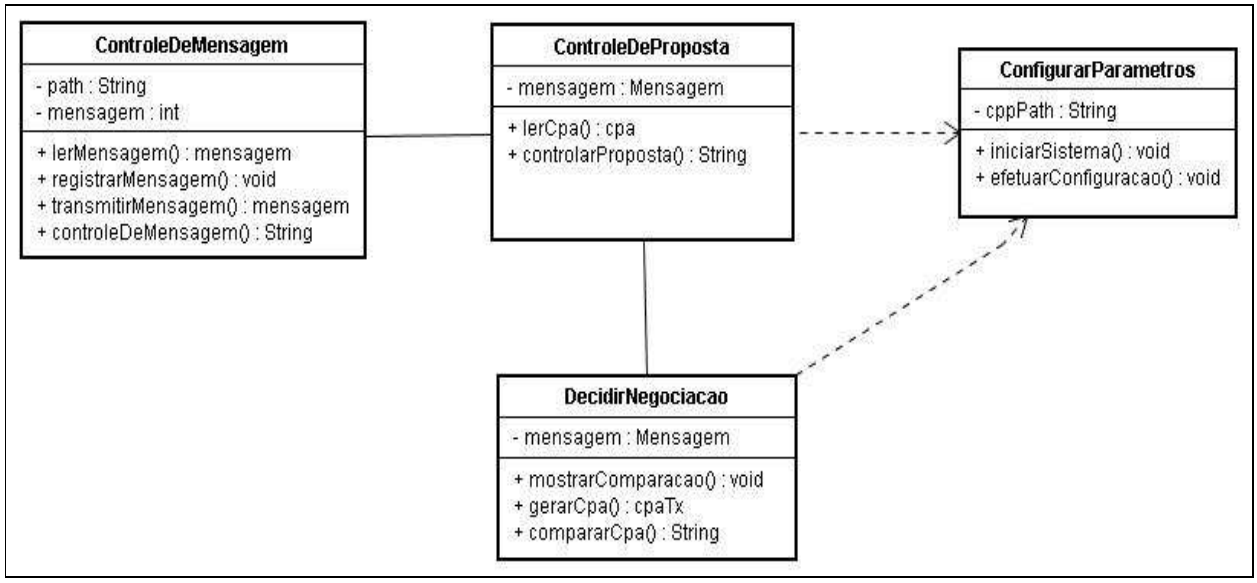

Figura 34 : O diagrama de classes

Neste diagrama, as classes ControleDeMensagem, ControleDeProposta e DecidirNegociacao estão vinculadas através de um relacionamento de associação, para a execução das operações de recebimento de mensagens e propostas.

A classe ControleDeProposta e a classe DecidirNegociacao dependem que a classe ConfigurarParametros efetue as operações de iniciar e efetuar a configuração do sistema, para que suas atividades possam ser realizadas.

Os próprios nomes das classes indicam a responsabilidade de cada uma na realização das operações do sistema.

Uma mensagem recebida é passada pela interface do serviço de mensagem para a classe ControleDeMensagens, que efetuará as operações de controle e validação da mensagem, identificando, conforme o tipo de mensagem, se há um anexo (CPA) a ser processado.

A classe ControleDePropostas é responsável por controlar as versões de CPA que são geradas e por executar o processo de validação e geração do documento (DOM) relativo ao CPA recebido, passando este documento para a classe DecidirNegociacao.

A classe DecidirNegociacao executa as operações de comparação com o CPA da empresa e apresentação dos resultados ao usuário.

O documento gerado (DOM) também será passado como parâmetro na chamada ao objeto registrarMensagem da classe de ControleDeMensagens, que efetuará a persistência deste documento no banco de dados, para controle.

No Anexo I deste trabalho são abordadas algumas considerações adicionais, abrangendo o código desenvolvido para a implementação do sistema.

5.6 Cenário de uso e Ambiente de testes

Para testar a aplicação desenvolvida, foi utilizado um cenário de teste, baseado nos CPP descritos na especificação do CPP/CPA [ebXML CPPA, 2002], com utilização do processo de negócio Compra/Venda (*Buy/Sell*) pois, conforme descrito nesta especificação, este é um dos processos de negócio mais comuns em processos de colaboração entre empresas, sendo portanto aplicável a diversos casos reais de uso.

Os CPP's utilizados nestes testes, para a empresa A, assim como para a empresa B, estão descritos no Anexo I deste trabalho.

O ambiente de testes, que será descrito logo a seguir, envolveu a utilização de duas máquinas, uma atuando como empresa A e a outra como empresa B.

Cenário : Recebimento de proposta de um possível parceiro comercial

Neste cenário foram testadas as funcionalidades de recebimento de uma mensagem de oferta encaminhada por um parceiro, extração das informações desta mensagem para processamento pelo sistema, assim como extração das informações contidas no documento CPA encaminhado anexo a esta mensagem.

Considerando que foi efetuado o mesmo processo de avaliação de propostas, tanto na empresa A, quanto na empresa B, o cenário é apresentado sob a visão da empresa A.

O cenário envolve o desenvolvimento de um processo de negociação entre as duas empresas para estabelecimento de um processo de negocio de compra e venda.

O CPP/CPA abrange itens de interoperabilidade e não abrange itens comerciais específicos de um determinado produto, por exemplo, e, portanto, não é necessário o detalhamento de qual serviço ou produto é o objeto de compra e venda.

A figura 35 mostra a tela inicial do sistema negociador, com as opções disponíveis para o usuário.

Figura 35: Tela inicial do sistema

Após iniciar o sistema, o usuário efetua a configuração, de acordo com as capacidades definidas no CPP da empresa. Uma vez configurado, o sistema está apto a receber e enviar mensagens.

Quando o sistema recebe uma mensagem de oferta da empresa B, efetua internamente o processo de leitura e extração de informações desta mensagem, executa o processo de comparação com a sua configuração interna e apresenta a tela com os resultados da comparação ao usuário (figura 36).

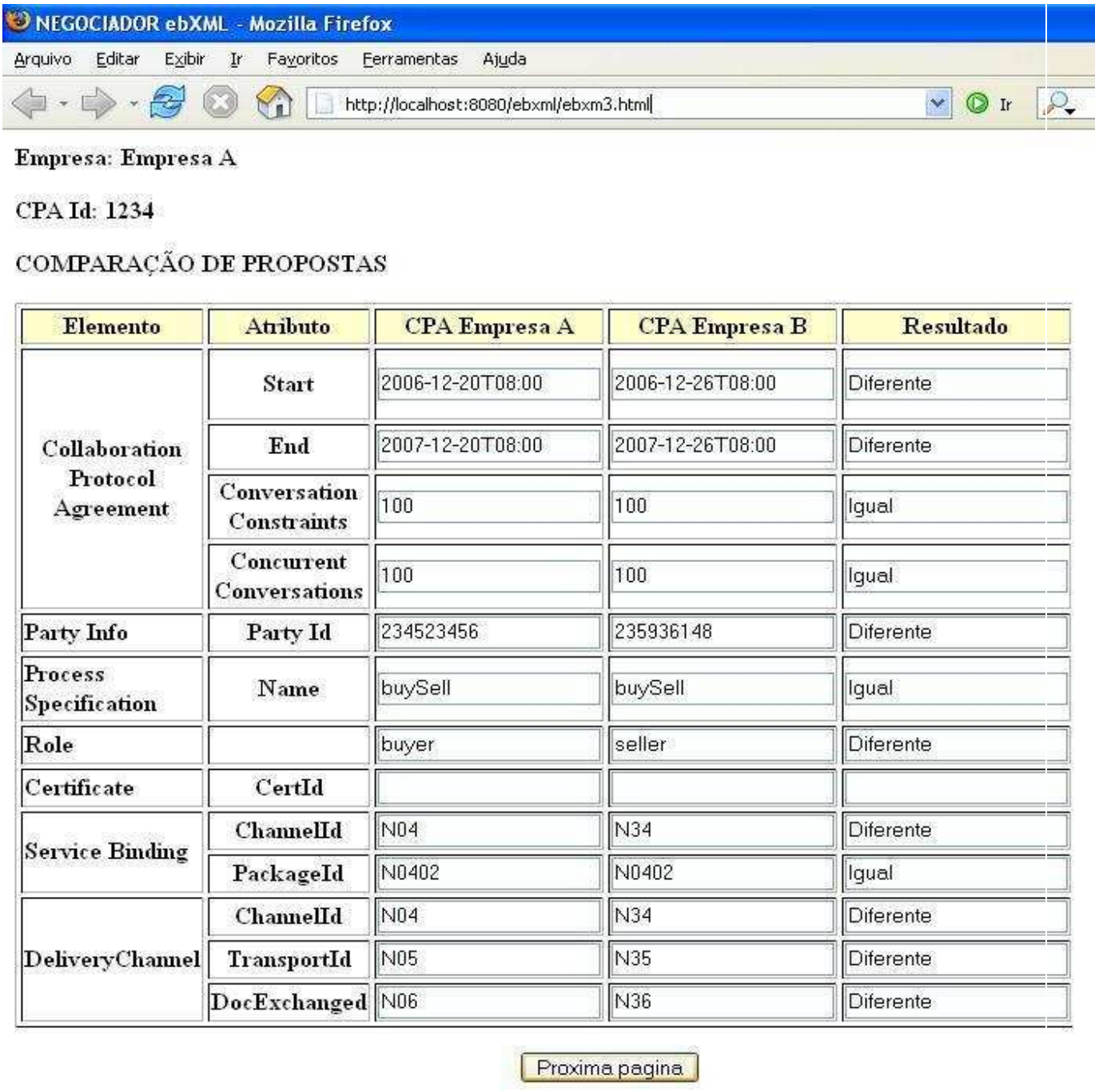

Figura 36: Tela de Comparação de propostas

O usuário efetuará a análise dos resultados e conforme a opção que solicitar, o sistema efetuará uma determinada ação: Editar - para elaboração de contra-proposta, Aceite, Recusar ou Finalizar a negociação (figura 37).

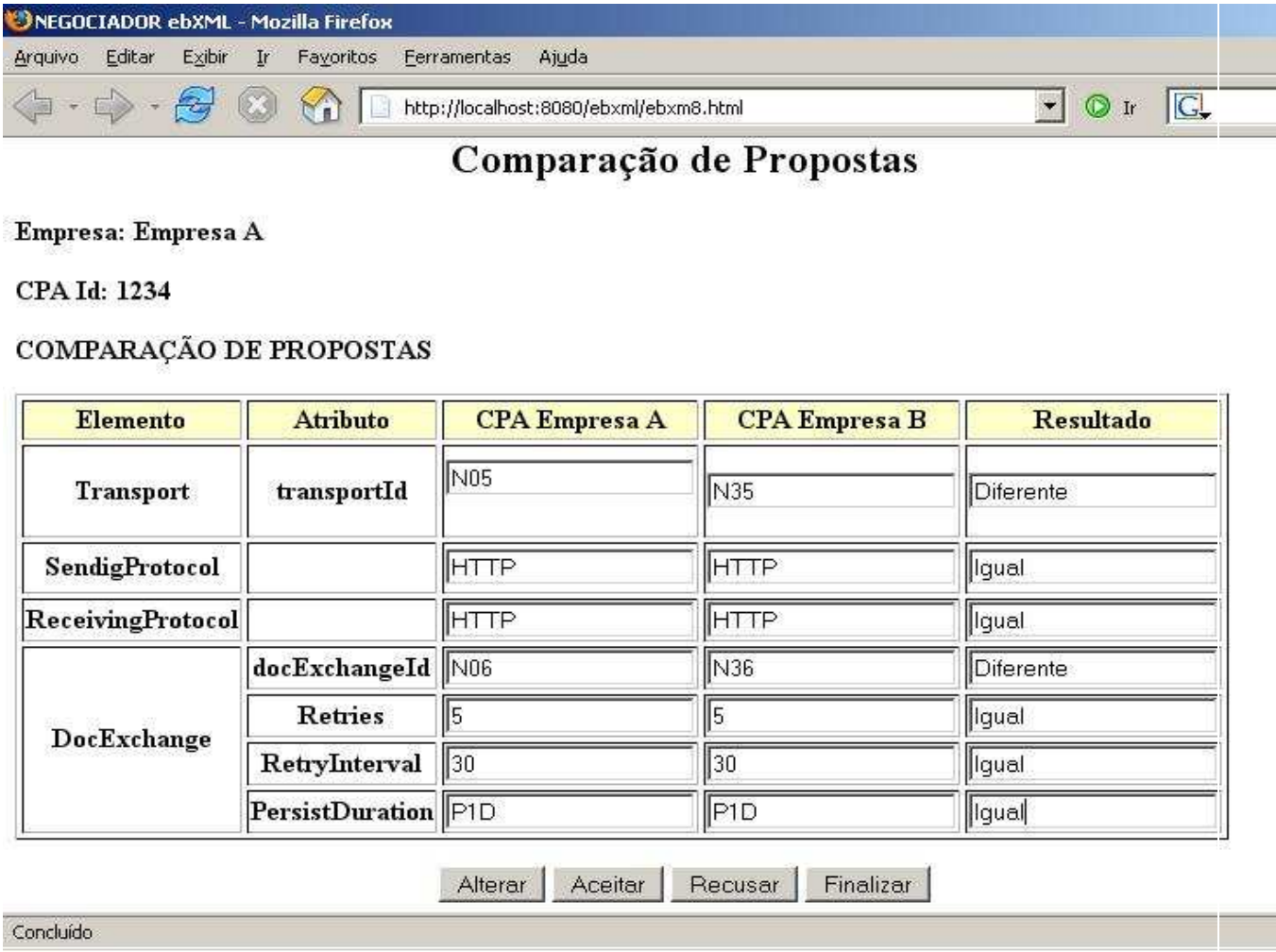

Figura 37: Tela de comparação com opção de decisão

Se o usuário decidir editar a proposta para se adequar à proposta recebida, o sistema possibilitará a alteração dos campos relativos a cada elemento na coluna da empresa A, possibilitando a edição conforme a seleção do usuário (figura 38).

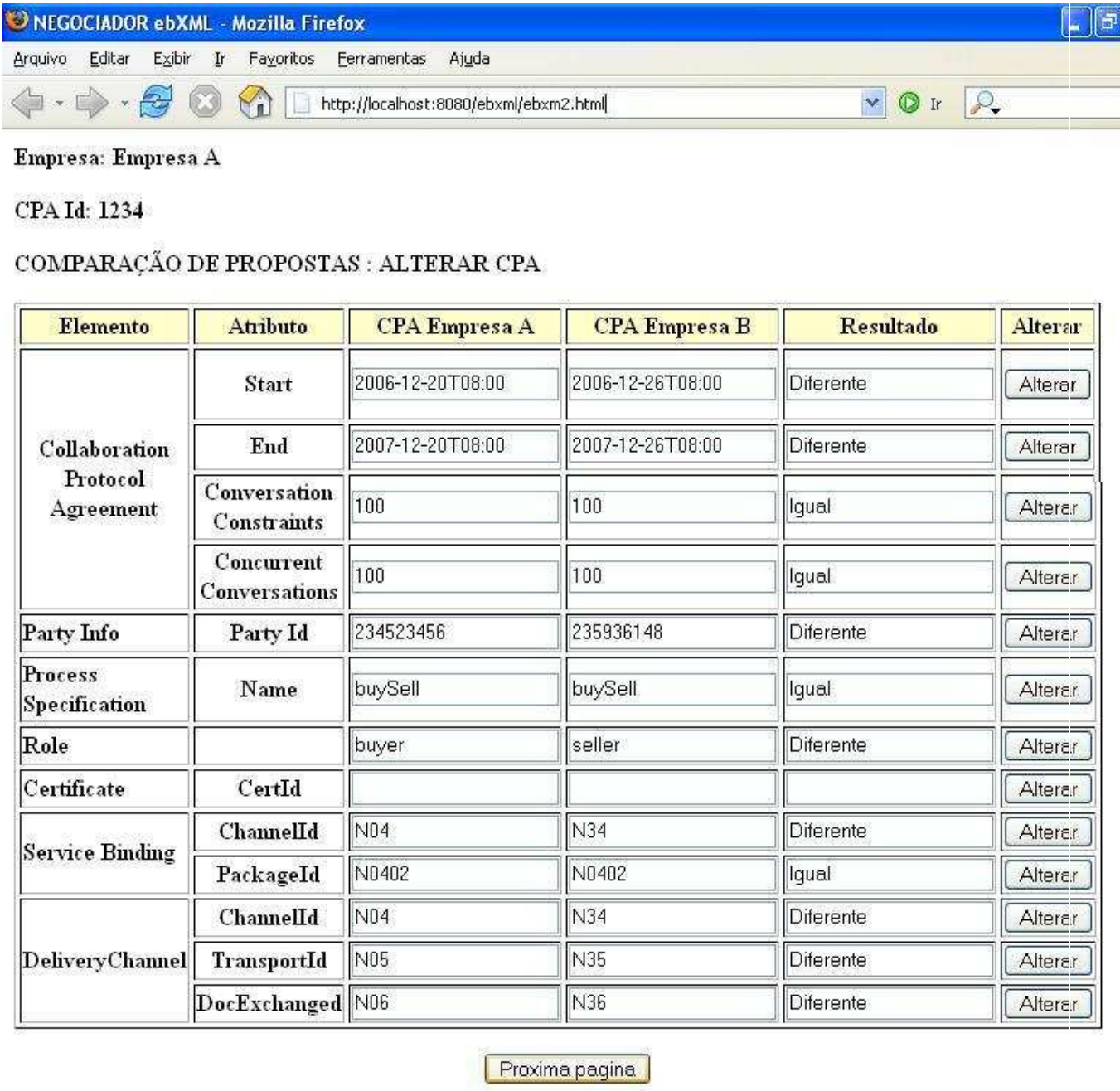

Figura 38: Tela de Alteração de CPA

Este processo repete-se com a troca de algumas mensagens de contra-oferta para teste de funcionamento do protótipo.

Quando ocorre o recebimento de uma mensagem de aceite da empresa B, o sistema na empresa A emite um aviso ao usuário, informando que a contra-proposta foi aceita pelo parceiro (figura 39).

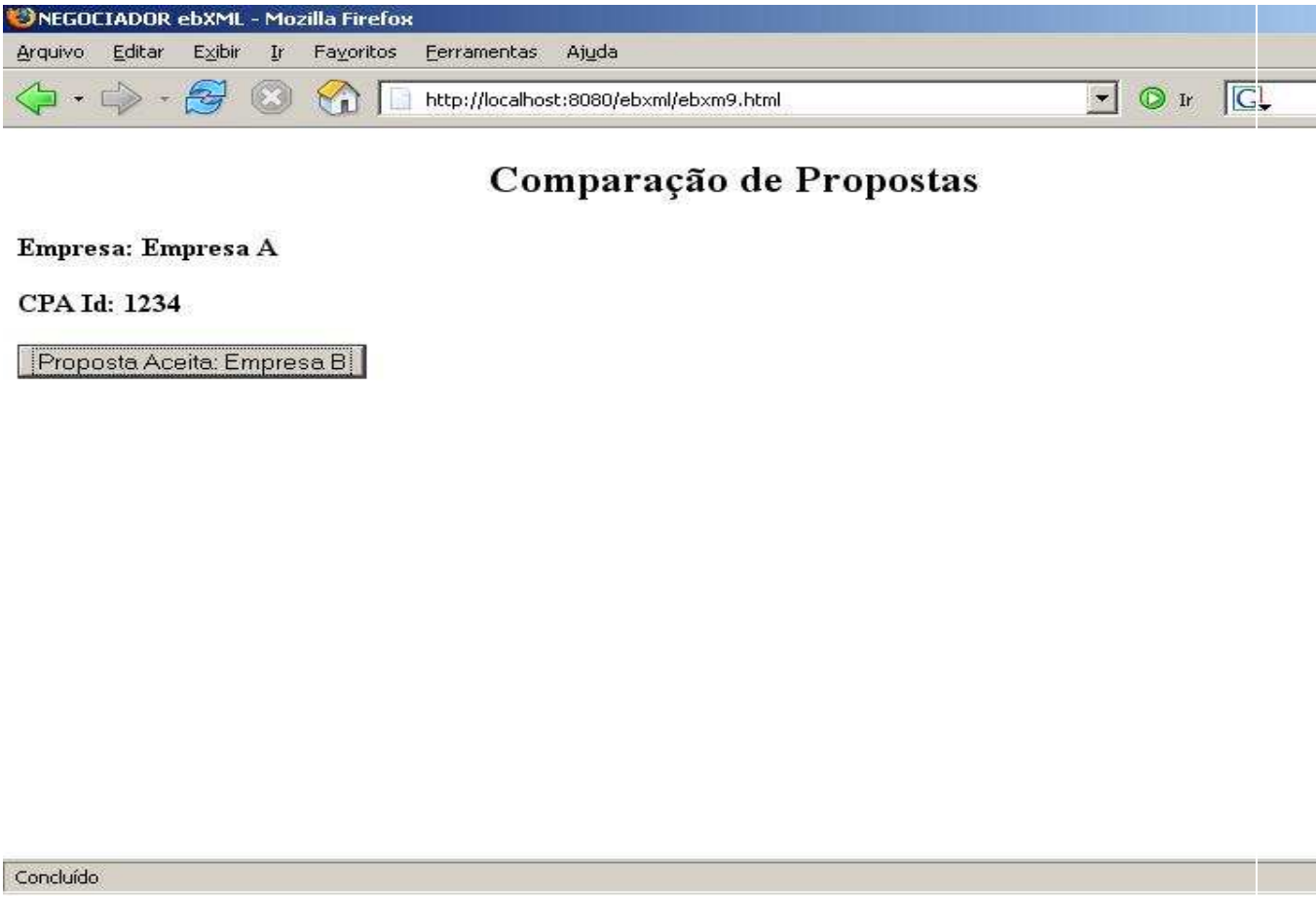

Figura 39: Mensagem de Aceite

Esta tela informa ao usuário o recebimento de uma mensagem de Aceite da empresa B e possiblita que ao ser pressionado o botão com a mensagem, um relatório com os itens que foram acordados seja apresentado ao usuário (não implementado nesta versão do protótipo).

5.7 Ambiente de Testes e Desenvolvimento

Para o desenvolvimento e testes com o protótipo foi utilizado o seguinte ambiente operacional:

- JAVA SDK 5.0
- Servidor Web/J2EE TomCat 5.0
- IDE NetBeans 5.5 e Eclipse WTP 3.2
- XMLSPY 2006 versão Home Edition (*freeware*) para a criação de documentos e esquemas XML (www.altova.com)
- Visual Paradigm versão Professional Edition (*trial*), para a modelagem UML (http://www.visual-paradigm.com).

- Jude versão Comunity Edition (http://jude.change-vision.com/jude-web/index.html), para modelagem UML.
- Dois equipamentos x86, executando Windows XP, com 512 Mb de memória

A figura 40 ilustra o ambiente de desenvolvimento e testes do protótipo.

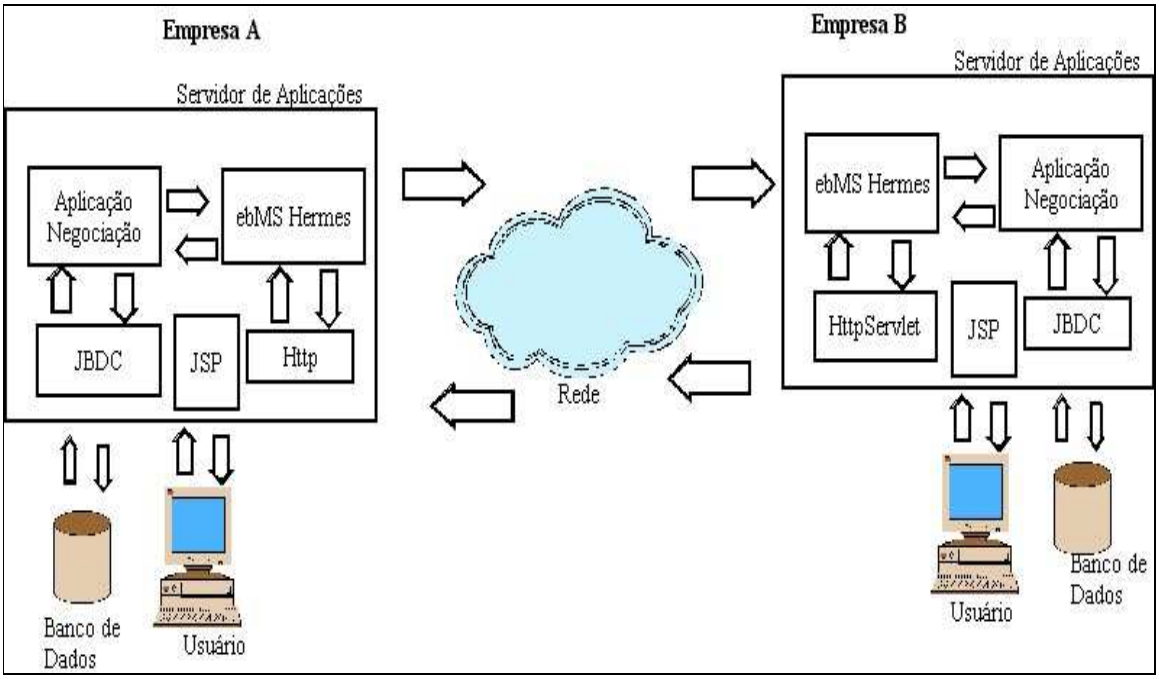

Figura 40: Ambiente de desenvolvimento e testes

5.8 Resultados Obtidos

Nestes testes, verificou-se a funcionalidade prática do sistema proposto, com relação a utilizar as informações fornecidas por um negociador humano de modo a suportar um processo de negociação e a geração automatizada do CPA.

Com relação ao protocolo de mensagens, foi identificada a necessidade de se acrescentar a este protocolo um tipo de mensagem sinalizando o estabelecimento do acordo, para ser enviada como resposta a uma mensagem de envio de aceite, quando o processo de negociação é finalizado. Esta resposta sinalizaria que a fase de negociação foi encerrada e que a fase de transação poderá ser iniciada pelas aplicações de negócio de cada parceiro.

O CPA gerado a partir destes testes encontra-se no Anexo I deste trabalho.

5.9 Dificuldades encontradas

Houve grande dificuldade para a configuração e utilização do serviço de mensagens ebXML (ebMS Hermes), considerando-se que a documentação do software não abrange as mensagens e situações de erro que podem ocorrer durante a implementação.

Uma alternativa a ser verificada em versões posteriores do protótipo para a implementação do serviço de mensagens é utilizar o software AXIS, um processador de mensagens SOAP, disponibilizado à comunidade pela Apache Software Foudation (www.apache.org/axis).

A utilização de um conjunto abrangente de tecnologias, como XML, ebXML, SOAP, JAVA EE, também implicaram na necessidade de uma fase de estudos mais longa, para possibilitar a elaboração deste trabalho.

5.10 Considerações Finais

O sistema foi testado dentro de um cenário de envio e recebimento de propostas, atuando tanto no recebimento de proposta, no papel de provedor de serviço, quanto no encaminhamento de proposta, no papel de cliente de serviços.

Utilizamos nestes testes o processo de negócio padrão de compra e venda, definido na especificação do CPP/CPA [ebXML CPPA, 2002], onde uma empresa A executa as regras de vendedor e uma empresa B o papel de compradora.

O protótipo é funcional, considerando que foram implementadas nesta primeira versão, os requisitos básicos para possibilitar a interação entre os parceiros e a geração do CPA.

O sistema teve um comportamento de acordo com o esperado, mas ainda há a necessidade de desenvolvimento complementar de funcionalidades, não contempladas nesta versão do protótipo, como por exemplo, os aspectos de controle de negociação e persistência em banco de dados, além do módulo de consulta de informações.

O protocolo de mensagens também é funcional, mas necessita que sejam definidas mensagens complementares de sinalização entre os parceiros, como, por exemplo, a definição de um elemento dentro da mensagem de finalização, para indicar se a negociação foi finalizada com sucesso ou com insucesso.

No caso de finalização com sucesso, esta mensagem de sinalização indicaria para os parceiros o ponto onde deve ser iniciado o processo de transações de negócio entre as suas aplicações comerciais.

O serviço de mensagens também necessita ser revisto, considerando-se as dificuldades de configuração e operacionalização do ebMS utilizado.

No próximo capitulo, efetuaremos as considerações finais com relação a este trabalho.

6. CONCLUSÃO

Neste capítulo são apresentadas as conclusões e contribuições desta dissertação, assim como sugestões de trabalhos futuros que poderão ser desenvolvidos.

6.1. Considerações Finais

Empresas inevitavelmente necessitam interagir entre si para o desenvolvimento de seus negócios, algumas vezes atuando como compradoras e em outras, atuando como vendedoras.

Algumas vezes, esta interação é efetuada através do telefone e do envio de documentos em papel, em um processo totalmente dependente da intervenção humana.

A execução deste processo através de comunicação eletrônica pode evitar a necessidade da troca de documentos em papel e reduzir o tempo gasto com atividades manuais.

Uma alternativa utilizada ainda hoje pelas empresas são os sistemas EDI, mas o seu custo de implementação e manutenção inviabiliza a participação de muitas empresas, principalmente aquelas de pequeno porte.

Neste contexto, o padrão ebXML surge como uma das principais alternativas para o estabelecimento de um processo de colaboração eletrônica através da troca de documentos e integração de processos de negócio via Internet, com a especificação de diversos componentes visando facilitar a implementação deste processo.

Os componentes CPP e CPA, utilizados nos processos de negociação eletrônica, facilitam o estabelecimento de acordos comerciais entre as empresas, abrangendo diversos itens técnicos de comunicação e de regra de processo de negócio, para possibilitar a interoperabilidade entre as aplicações dos parceiros.

Com o presente trabalho, verificou-se que a hipótese de geração automatizada deste acordo, citada na especificação destes componentes [CPPA, 2002] e em trabalhos relacionados [Hofreiter e Hummer, 2003], pode ser efetuada, mas ainda há a necessidade da intervenção humana para condução do processo de negociação, principalmente para a tomada de decisão e na avaliação de aspectos estratégicos, como por exemplo, a configuração dinâmica de parâmetros do acordo, em função do interesse de negócio em se estabelecer uma colaboração com um determinado parceiro.

Neste contexto, foi elaborada como parte do trabalho, uma proposta de automatização das atividades de geração do acordo de colaboração e de interação entre as partes durante o processo de negociação, para auxiliar os negociadores humanos durante este processo.

A implementação da proposta foi efetuada através do desenvolvimento de um protótipo, abrangendo um protocolo básico de mensagens para automatização do processo de interação entre os parceiros e funcionalidades de configuração, controle, comparação de propostas e apresentação dos resultados a um usuário, para a definição do processo.

Este trabalho não tem a pretensão de esgotar todas as possibilidades para tratamento da referida hipótese, mas sim de deixar um alicerce que poderá ser explorado para a

construção de outras propostas, visando elevar a utilização de sistemas automatizados em processos de negociação eletrônica entre empresas.

6.2. Contribuições

A utilização de um sistema que suporte o processo de geração do CPA pode reduzir a necessidade de intervenção humana e contribuir para agilizar o processo de negociação eletrônica entre empresas que utilizem o padrão ebXML, e conseqüentemente, agilizar o processo de produção destas empresas.

Nesse sentido, o sistema proposto neste trabalho pode auxiliar os negociadores humanos que atuam em processos de negociação ebXML, comparando e fornecendo informações sobre os parâmetros que estão sendo oferecidos, evitando que esta tarefa seja efetuada manualmente e possibilitando assim que decisões sejam tomadas de forma mais rápida.

A agilização desta fase de negociação pode também possibilitar a uma empresa estender o processo de negociação, abrangendo um universo maior de possíveis parceiros de negócio e conseqüentemente, comparar e obter melhores condições técnicas de colaboração.

O trabalho também contribui para a divulgação de conhecimentos referentes ao padrão ebXML, facilitando o desenvolvimento de outros trabalhos utilizando este padrão.

6.3. Trabalhos Futuros

Com relação aos trabalhos que poderão ser desenvolvidos para complementar ou evoluir o presente trabalho, podemos citar os seguintes itens:

- Implementação de recursos adicionais no protótipo para suporte à decisão, visando passar o sistema do atual tipo NSS – Sistema de Suporte a Negociação para ser um NDSS – Sistema de Suporte a Decisão em Negociação.
- Dotar o protótipo de módulo que possibilite a avaliação de estratégias de negociação.
- Definir o protocolo de mensagens de negociação como um ebBP, de forma que ele possa ser publicado em um Registro ebXML como um processo de negócio de negociação, adaptando-o às especificações ebXML [ebBP, 2006]. Para isto pode ser utilizado o documento *ebBP Deployment Profile Template v0.95 For OASIS ebXML ebBP v2.0.*, que especifica um modelo para a definição de processos de negócio e de documentos de negócio no contexto do padrão ebXML [ebBP DPT,2006], abrangendo também a definição de mensagens de negócio.
- Utilizar um banco de dados XML, juntamente com a XSLT (*XML Stylesheet Language Transformation)* no sistema proposto, para otimizar o processamento e manipulação dos documentos XML CPP e CPA. Há algumas opções disponíveis na Web, como por exemplo, a versão gratuita do banco de dados DB2 (DB2 v9) que suporta o armazenamento de dados XML, disponível através do endereço eletrônico http://www-306.ibm.com/software/data/db2/express/download.html.

7. REFERÊNCIAS BIBLIOGRAFICAS

[Bartolini *et. al.*,2001] Bartolini, Cláudio. Preist, Chris. Kuno, Harumi. Requirements for Automated Negotiation. Zeleznikow, John. A Comparative Study of Negotiation Decision Support Systems. Database Research Laboratory. Australia. IEEE 2001.

[Belluci e Zeleznikow, 1999] Belucci, Emilia. Zeleznikow, John. A Comparative Study of Negotiation Decision Support Systems. Database Research Laboratory. Austrália. IEEE 1999.

[Byde e Piccinelli,2002] Byde, Andrew. Piccinelli, Giacomo. Automating Negotiation over B2B Processes. Proceedings of the 13th International Workshop on Database and Expert Systems.IEEE.2002.

[Chiu, 2002] Chiu, Eric. ebXML Simplified – A Guide to the New Standard for Global E-Commerce. Editora John Wiley & Sons, Inc. 2002. p.23, 24 e 60.

[Clements, 2002] Clements, Tom. Overview of SOAP. SUN Developer, Janeiro de 2002. http://java.sun.com/developer/technicalArticles/xml/webservices/ acessado em Fevereiro de 2005.

[Deitel *et al*. 2003, p. 06].Deitel, H.M, Deitel P. J., DuWaldt, B., Trees, L.K. Web Services A Technical Introduction. Prentice Hall, 2003. p. 06, 17, 34, 39, 106 e 145.

[ebBP, 2006] ebXML Business Process Specification Schema – Technical Specification v2.0.3. UN/CEFACT e OASIS, Julho 2006. www.ebxml.org/especs/ebBPSS.pdf

[ebBP DPT,2006] ebXML Business Process Deployment Profile Template v0.95 For OASIS ebXML ebBP v2.0. UN/CEFACT e OASIS, Junho 2006. www.ebxml.org/especs/ebBPSS

[ebXML Core Component, 2001] ebXML Core Component Overview v1.05 UN/CEFACT e OASIS, Maio 2001. www.ebxml.org/especs/ccover.pdf

[ebXML CPPA, 2002] ebXML Collaboration Protocol Profile and Agreement Specification v. 2.0. UN/CEFACT e OASIS, Abril 2002. www.ebxml.org/especs/ebcpp2.pdf

[ebMS, 2002] ebXML Message Service v2.0. UN/CEFACT e OASIS, Abril 2002. www.ebxml.org/especs/ebMS2.pdf

[ebRS,2002] ebXML Registry Services Specification V2.0. UN/CEFACT e OASIS, Abril 2002. www.ebxml.org/especs/ebRS2.pdf

[ebrim,2002] ebXML Registry Information Model V2.0. UN/CEFACT e OASIS, Abril 2002. www.ebxml.org/especs/ebrim2.pdf

ebXML[ebXML Requirements Specification,2002] ebXML Requirements Specification v1.06 UN/CEFACT e OASIS, Abril 2002. www.ebxml.org/especs/ebREQ.pdf

[ebXML T.A., 2002] ebXML Tecnical Arquitecture - UN/CEFACT e OASIS, Maio 2002. www.ebxml.org/especs/ebTA.pdf

[Gartner,2003] Gartner, Group. Different Goals Mean ebXML is not a Rival to Web Services.Gartner Group. Agosto 2003.

[Gartner,2004] Gartner, Group. ebXML´s Future Appears Brihtest in Messaging. Gartner Group. Agosto 2004.

[Gupta *et. al*., 2006] Gupta, Varun. Fernandes, Leo. Parikh, Ash. Enable real-world trading partner Collaborations. Java World. Agosto 2006. www.javaworld.com

[Harold e Means,2004] Harold, Rusty. Means, Scott. XML In a Nutshell, A Desktop Quick Reference –Editora O'Reilly- 2004 p. 60.

[Hendricks *et al*., 2002, p.39] Hendricks, Mack. Irani, Romim. Toussaint, Galbkaith. Professional Java Web Services. Editora Wrox, 2002.p.39 e 41.

[Hofreiter e Humer,2006] Hofreiter, Birgit. Huemer, Christian. ebXML Business Process: Defined both in UMM and BPSS. Department of Computer Science and Business Informatics, Univervity of Viena. Viena, Austria.2006

[Hofreiter e Humer,2003] Hofreiter, Birgit. Huemer,Christian. ebXML: Status, Research Issues, and Obstacles. *IEEE* Proceedings of the 12th Int.l Workshop on Research Issues in Data Engineering: Engineering e-Commerce/ e-Business Systems. 2003

[Irani, 2001] Irani, Romin. An Introduction to ebXML. Web Services Architect, Maio 2001. www.websercearchitect.com

[Lhotka,2005] Rocky, Lhotka. The Intersection of Objects and Services. TheServerSide.Net, Fevereiro 2005. http://www.serverside.com acessado em Junho de 2005

[Liang *et al*.,2004], Liang, Pen. Keqing, He. Bing, Li. Liu, Jin. Interoperability Test of ebXML Solutions. IEEE - Proceedings of the Fourth International Conference on Computer and Information Tecnology (CIT'04).2004

[Newcomer,2002,p2] Newcomer, Eric. Understanding Web Services. Boston, Editora Addison-Wesley,2002.

[Nunes et al.,2003] Nunes, Ricardo *et. al.* An Architecture and a Decision Making Model to Web-Based Electronic Commerce Negotiation. The 3o. IFIP Conference on e-Commerce, e-Business and e-Government. September 2003. Guarujá. São Paulo.

[Niemeyer e Knudsen, 2004] Niemeyer, Patrick e Knudsen, Jonathan. An Introduction to XML. O´Reilly Book Excerpts: Learning Java, 2nd Edition, 2004. http://www.onjava.com.pub/q/excerpts acessado em Fevereiro de 2005.

[Segev, 2001] Segev, Arie. Automated Negotiations: A Survey of the State of the Art. University of California, Berkeley. Berkeley, CA , Fevereiro 2001.

[Schmidt e Vinoski, 2002] Schmidt, Douglas C. e Vinoski, Steve. Object Interconnnections: CORBA and XML Part 3: SOAP and Web Services. C/C++ Users Journal, 2002 http://www.cuj.com/experts/1910/vinoski.htm

[Strobel,2002] Strobel, Michael. Effects of Electronic Markets on Negotiation Processes. IBM Research Laboratory. Zurich. Alemanha, 2002.

[Tidwell, 2002a] Doug Tidwell. Introduction to XML. IBM developerWorks, Agosto 2002. http://www-106.ibm.com/developerworks/xml

[Tidwell, 2002b] Doug Tidwell. Web Services – The Web´s next revolution. IBM.com/developerWorks, Setembro 2002. http://www-06.ibm.com/developerworks/xml

[Tizzo N., *et. al*,2004]. Tizzo, N. Mendes, Manuel de Jesus. Service Composition Applied to E-Government. IEEE IFIP WCC2004, Toulouse 2004.

[Vasudevan, 2001] Vasudevan, Venu. A Web Services Primer.XML.com, Abril 2001, http://webservices.xml.com/pub/a/ws/2001/04/04/webservices/indes.html

[Willaert, 2001] Willaert, Frederik. XML-Based Frameworks and Standards for B2B E-Commerce. Katholieke Universiteit Leuven. Leuven, Alemanha, Maio 2001.

[W3C, 2004a] World Wide Web Consortium. Extensible Markup Language (XML) 1.0 Terceira edição, Fevereiro 2004, http://www.w3.org/TR/2004/REC-xml-20040204

[W3C, 2004b] World Wide Web Consortium. XML Schema Part 0: Primer Segunda edição, Outubro 2004, http://www.w3.org/TR/2004/REC-xmlschema-0-20041028/

[W3C, 2004c] World Wide Web Consortium.Web Services Architecture. Fevereiro 2004,

http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/

[W3C, 2004d] World Wide Web Consortium. SOAP Version 1.2 Part 1: Messaging Framework. Junho 2003, http://www.w3.org/TR/2003/REC-soap12-part1-20030624/

[W3C, 2002e] World Wide Web Consortium.Web Services Description Language (WSDL) Version 2.0 Part1 Core Language.Agosto 2005, http://www.w3.org/TR/wsdl20/

[W3C DOM, 2004f] World Web Consortium. Document Object Model (DOM) Level 3 Core Specification 1.1 Version 1.0. Abril 2004.

ANEXO I

Neste anexo são listados exemplos de documentos CPP e CPA com os respectivos comentários relativos aos elementos que compõem estes documentos, assim como os CPP utilizados no teste do protótipo e algumas considerações sobre o código das classes do sistema e mensagens geradas conforme o protocolo de mensagens.

1 Exemplo de um documento CPP

O exemplo abaixo, retirado das especificações do CPP/CPA [ebXML CPPA, 2002] mostra um documento CPP:

```
<?xml version="1.0" encoding="UTF-8"?>
<tp:CollaborationProtocolProfile
        xmlns:tp="http://www.ebxml.org/namespaces/tradePartner"
        xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
        xsi:schemaLocation="http://www.ebxml.org/namespaces/tradePartner
        http://ebxml.org/project_teams/trade_partner/cpp-cpa-v1_0.xsd"
        tp:version="1.1">
        <tp:PartyInfo>
           <tp:PartyId>123456789</tp:PartyId>
           <tp:CollaborationRole tp:id="N00">
                <tp:ProcessSpecification tp:version="1.0" tp:name="buySell"
                xlink:type="simple"
                xlink:href="http://www.ebxml.org/processes/buySell.xml"/>
                <tp:Role tp:name="buyer" xlink:type="simple"
                xlink:href="http://ebxml.org/processes/buySell.xml#buyer"/>
                <tp:CertificateRef tp:certId="N03"/>
                <tp:ServiceBinding tp:channelId="N04" tp:packageId="N0402">
                     <tp:Service
                     tp:type="uriReference">uri:example.com/services/buyerService</tp:Ser
                     vice>
                </tp:ServiceBinding>
           </tp:CollaborationRole>
           <tp:Certificate tp:certId="N03">
                <ds:KeyInfo/>
           </tp:Certificate>
                <tp:DeliveryChannel tp:channelId="N04" tp:transportId="N05"
                tp:docExchangeId="N06">
```

```
            <tp:Characteristics tp:syncReplyMode="none"
tp:nonrepudiationOfOrigin="true" tp:nonrepudiationOfReceipt="false"
tp:secureTransport="true" tp:confidentiality="true" tp:authenticated="true"
tp:authorized="false"/>
        </tp:DeliveryChannel>
        <tp:DeliveryChannel tp:channelId="N07" tp:transportId="N08"
        tp:docExchangeId="N06">
             <tp:Characteristics tp:syncReplyMode="none"
                tp:nonrepudiationOfOrigin="true"
                tp:nonrepudiationOfReceipt="false" tp:secureTransport="false"
                tp:confidentiality="true" tp:authenticated="true"
                tp:authorized="false"/>
        </tp:DeliveryChannel>
        <tp:Transport tp:transportId="N05">
                <tp:SendingProtocol tp:version="1.1">HTTP</tp:SendingProtocol>
                <tp:ReceivingProtocol
                     tp:version="1.1">HTTP</tp:ReceivingProtocol>
                <tp:Endpoint
                     tp:uri="https://www.example.com/servlets/ebxmlhandler"/>
                <tp:TransportSecurity>
                        <tp:Protocol tp:version="3.0">SSL</tp:Protocol>
                        <tp:CertificateRef tp:certId="N03"/>
                </tp:TransportSecurity>
        </tp:Transport>
        <tp:Transport tp:transportId="N08">
                <tp:SendingProtocol tp:version="1.1">HTTP</tp:SendingProtocol>
                <tp:ReceivingProtocol
                        tp:version="1.1">SMTP</tp:ReceivingProtocol>
                <tp:Endpoint tp:uri="mailto:ebxmlhandler@example.com"/>
        </tp:Transport>
        <tp:DocExchange tp:docExchangeId="N06">
                <tp:ebXMLBinding tp:version="0.98b">
                <tp:ReliableMessaging tp:deliverySemantics="OnceAndOnlyOnce"
                tp:idempotency="true" tp:messageOrderSemantics="Guaranteed">
                                <tp:Retries>5</tp:Retries>
                                <tp:RetryInterval>30</tp:RetryInterval>
                                <tp:PersistDuration>P1D</tp:PersistDuration>
                </tp:ReliableMessaging>
                <tp:NonRepudiation>
                   <tp:Protocol>http://www.w3.org/2000/09/xmldsig#</tp:Protocol>
<tp:HashFunction>http://www.w3.org/2000/09/xmldsig#sha1</tp:HashFunction>
<tp:SignatureAlgorithm>http://www.w3.org/2000/09/xmldsig#dsa-
sha1</tp:SignatureAlgorithm>
                    <tp:CertificateRef tp:certId="N03"/>
                </tp:NonRepudiation>
                <tp:DigitalEnvelope>
```

```
                                <tp:Protocol tp:version="2.0">S/MIME</tp:Protocol>
                                    <tp:EncryptionAlgorithm>DES-
                                      CBC</tp:EncryptionAlgorithm>
                                      <tp:CertificateRef tp:certId="N03"/>
                                </tp:DigitalEnvelope>
                            </tp:ebXMLBinding>
                        </tp:DocExchange>
                </tp:PartyInfo>
                <tp:Packaging tp:id="N0402">
                    <tp:ProcessingCapabilities tp:parse="true" tp:generate="true"/>
                    <tp:SimplePart tp:id="N40" tp:mimetype="text/xml">
                        <tp:NamespaceSupported
                        tp:location="http://ebxml.org/project_teams/transport/messageService.xsd"
                        tp:version="0.98b">http://www.ebxml.org/namespaces/messageService</tp:
                        NamespaceSupported>
                        <tp:NamespaceSupported
                        tp:location="http://ebxml.org/project_teams/transport/xmldsig-core-
                        schema.xsd"
                        tp:version="1.0">http://www.w3.org/2000/09/xmldsig</tp:NamespaceSupp
                        orted>
                    </tp:SimplePart>
                    <tp:SimplePart tp:id="N41" tp:mimetype="text/xml">
                        <tp:NamespaceSupported
                        tp:location="http://ebxml.org/processes/buysell.xsd"
                        tp:version="1.0">http://ebxml.org/processes/buysell.xsd</tp:NamespaceSup
                        ported>
                    </tp:SimplePart>
                    <tp:CompositeList>
                            <tp:Composite tp:id="N42" tp:mimetype="multipart/related"
                                tp:mimeparameters="type=text/xml">
                                        <tp:Constituent tp:idref="N40"/>
                                        <tp:Constituent tp:idref="N41"/>
                             </tp:Composite>
                    </tp:CompositeList>
                </tp:Packaging>
                <tp:Comment tp:xml_lang="en-us">buy/sell agreement between example.com and
contrived-example.com</tp:Comment>
</tp:CollaborationProtocolProfile>
```

O documento inicia com as declarações dos *namespaces* padrão e respectivos prefixos utilizados no documento.

O elemento *PartyInfo* providencia informações sobre a empresa detentora do CPP e possui os seguintes elementos filhos:

- *PartyId*, para identificação lógica da organização através de um código padrão ou qualquer outro identificador especifico definido pela comunidade de negócio onde a empresa atua.

- *PartyRef*, aponta para um recurso externo onde podem ser obtidas mais informações sobre a organização.

- *CollaborationRole*, é o principal elemento do CPP, onde são declaradas as informações do Processo de Negócio a ser utilizado na colaboração, no elemento ProcessSpecification, e as regras que a empresa executa neste processo, no elemento Role. Este elemento referencia o ebBPS que está armazenado no registro.

-*Certificate*, identifica o certificado digital utilizado pela empresa para autenticar o envio de mensagens.Este elemento não é obrigatório.

-*DeliveryChannel*, define os canais através dos quais a empresa pode receber mensagens, incluindo referencias para o protocolo de mensagens e para o protocolo de transporte descrito no elemento transport (HTTP, FTP ou SMTP, por exemplo).

Podem ser definidas varias especificações lógicas de canais, com configurações especificas de protocolos para cada canal, conforme as capacidades de comunicação que a empresa possui.

- *Transport*, especifica detalhes do protocolo de transporte a ser utilizado como o tipo de versão, por exemplo, e o endereço (url) de acesso à aplicação de negócio da empresa.

A definição do protocolo de mensagens a ser utilizado, como o ebMS por exemplo, também é efetuada neste elemento.

- *DocExchange*, define detalhes de utilização de recursos de segurança para a troca de documentos de negócio, tais como algoritmo de criptografia e de assinatura digital, por exemplo e parâmetros de retransmissão de mensagens (*Retries*).

Conforme pode ser observado neste documento CPP, cada um destes elementos possui elementos filhos para a definição destes detalhes de configuração.

O elemento *Packaging* define como o cabeçalho e o conteúdo (*payload)* das mensagens serão empacotados para a transmissão.

As mensagens são processadas utilizando-se o protocolo *SOAP with Attachments* e os elementos filhos deste elemento providenciam informações de como estas mensagens estão organizadas.

O elemento *Packaging* possui três principais elementos filhos:

- *ProcessingCapabilities*, que possui dois atributos, *parse e generate*, que asseguram se o sistema da empresa é capaz de efetuar a análise (*parsing*) e geração das mensagens.

- *SimplePart*, que possui um atributo de identificação (*id*) e um atributo *mimetype* para definição de que as partes da mensagem são do tipo texto XML (*mimetype=”text/xml*).

Possui também um elemento *NamespaceSupported*, com o atributo *location*, para indicar a localização dos *namespaces* dos esquemas XML utilizados nas mensagens, como o esquema XML do serviço de mensagens, por exemplo.

As partes simples de uma mensagem podem ser referenciadas pelo seu atributo de identificação, para composição de uma mensagem, dentro do elemento *CompositeList,*.

- *CompositeList,* que possui um atributo de identificação (*id*) e um atributo *mimetype* que indica se as mensagens são compostas de elementos *SimplePart* (*mimetype=”multipart/related*”). Se as partes de uma mensagem forem transmitidas individualmente este elemento não é utilizado.

O elemento opcional *Signature* é utilizado para assinatura do CPP, de acordo com a especificação XML (XMLDSIG) para assinatura digital.

O elemento Comment pode ser utilizado opcionalmente pelo autor do CPP para definir uma nota de texto no CPP.

1.1 Exemplo de um documento CPA

O exemplo de um CPA, mostrado abaixo, é baseado no CPP anterior:

```
<?xml version="1.0"?>
<tp:CollaborationProtocolAgreement
        xmlns:tp="http://www.ebxml.org/namespaces/tradePartner"
        xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
        xsi:schemaLocation="http://www.ebxml.org/namespaces/tradePartner
        http://ebxml.org/project_teams/trade_partner/cpp-cpa-v1_0.xsd"
        xmlns:xlink="http://www.w3.org/1999/xlink"
        xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
        tp:cpaid="uri:yoursandmycpa"
        tp:version="1.2">
        <tp:Status tp:value="proposed"/>
        <tp:Start>2001-05-20T07:21:00Z</tp:Start>
        <tp:End>2002-05-20T07:21:00Z</tp:End>
        <tp:ConversationConstraints tp:invocationLimit="100"
tp:concurrentConversations="100"/>

        <-- Inicio das definições do primeiro parceiro – empresa A -->
        <tp:PartyInfo>
                <tp:PartyId >123456789</tp:PartyId>
                <tp:PartyRef xlink:href="http://example.com/about.html"/>
                <tp:CollaborationRole tp:id="N00">
                    <tp:ProcessSpecification tp:version="1.0" tp:name="buySell"
                    xlink:type="simple"
                    xlink:href="http://www.ebxml.org/processes/buySell.xml"/>
                    <tp:Role tp:name="buyer" xlink:type="simple"
                    xlink:href="http://ebxml.org/processes/buySell.xml#buyer"/>
                    <tp:CertificateRef tp:certId="N03"/>
                    <tp:ServiceBinding tp:channelId="N04" tp:packageId="N0402">
                                …
                </tp:CollaborationRole>
```

```
                <tp:Certificate tp:certId="N03">
                        <ds:KeyInfo/>
                </tp:Certificate>
                <tp:DeliveryChannel tp:channelId="N04" tp:transportId="N05"
                     tp:docExchangeId="N06">
                                ….
                </tp:DeliveryChannel>
                <tp:Transport tp:transportId="N05">
                   <tp:SendingProtocol tp:version="1.1">HTTP</tp:SendingProtocol>
                      <tp:ReceivingProtocol
                        tp:version="1.1">HTTP</tp:ReceivingProtocol>
                                …
                </tp:Transport>

                <tp:DocExchange tp:docExchangeId="N06">
                        <tp:ebXMLBinding tp:version="0.98b">
                        <tp:ReliableMessaging tp:deliverySemantics="OnceAndOnlyOnce"
                        tp:idempotency="true" tp:messageOrderSemantics="Guaranteed">
                                        <tp:Retries>5</tp:Retries>
                                        <tp:RetryInterval>30</tp:RetryInterval>
                                        <tp:PersistDuration>P1D</tp:PersistDuration>
                        </tp:ReliableMessaging>
                                                …
                        </tp:ebXMLBinding>
                </tp:DocExchange>
        </tp:PartyInfo>

<-- Inicio das definições da outra parte – empresa B -->
        <tp:PartyInfo>
                <tp:PartyId tp:type="DUNS">987654321</tp:PartyId>
                <tp:PartyRef xlink:type="simple" xlink:href="http://contrived-
                example.com/about.html"/>
                <tp:CollaborationRole tp:id="N30">
                        <tp:ProcessSpecification tp:version="1.0" tp:name="buySell"
                        xlink:type="simple"
                        xlink:href="http://www.ebxml.org/processes/buySell.xml"/>
                        <tp:Role tp:name="seller" xlink:type="simple"
                        xlink:href="http://ebxml.org/processes/buySell.xml#seller"/>
                        <tp:CertificateRef tp:certId="N33"/>
                        <tp:ServiceBinding tp:channelId="N34" tp:packageId="N0402">
                                        …
                        </tp:ServiceBinding>
                </tp:CollaborationRole>
                <tp:Certificate tp:certId="N33">
                        <ds:KeyInfo/>
                </tp:Certificate>
```

```
            <tp:DeliveryChannel tp:channelId="N34" tp:transportId="N35"
               tp:docExchangeId="N36">
                                    …
            </tp:DeliveryChannel>
            <tp:Transport tp:transportId="N35">
                  <tp:SendingProtocol tp:version="1.1">HTTP</tp:SendingProtocol>
                  <tp:ReceivingProtocol
                  tp:version="1.1">HTTP</tp:ReceivingProtocol>
                  <tp:Endpoint tp:uri="https://www.contrived-
                  example.com/servlets/ebxmlhandler" tp:type="allPurpose"/>
                                    …
            </tp:Transport>
            <tp:DocExchange tp:docExchangeId="N36">
                  <tp:ebXMLBinding tp:version="0.98b">
                                          …
                  </tp:ebXMLBinding>
            </tp:DocExchange>
      </tp:PartyInfo>
      <tp:Packaging tp:id="N0402">
            <tp:ProcessingCapabilities tp:parse="true" tp:generate="true"/>
            <tp:SimplePart tp:id="N40" tp:mimetype="text/xml">
                        …
            </tp:SimplePart>
            <tp:CompositeList>
                  <tp:Composite tp:id="N42" tp:mimetype="multipart/related"
                  tp:mimeparameters="type=text/xml;">
                        <tp:Constituent tp:idref="N40"/>
                        <tp:Constituent tp:idref="N41"/>
                  </tp:Composite>
            </tp:CompositeList>
      </tp:Packaging>
      <tp:Comment xml:lang="en-us">buy/sell agreement between example.com and
      contrived-example.com</tp:Comment>
</tp:CollaborationProtocolAgreement>
```

Conforme pode ser observado neste exemplo, os elementos que compõem o documento do CPA são os mesmos elementos descritos anteriormente no exemplo do CPP.

Para ressaltar a definição dos parâmetros a serem seguidos por cada parte, incluímos um comentário no documento, antes da definição dos elementos de cada empresa.

O processo de negócio que deverá ser utilizado nesta colaboração é o processo *buySell* (compra e venda), conforme estabelecido no elemento *ProcessSpecification*, dentro do elemento *PartyInfo* de ambas as partes.

Este elemento também define o papel que cada parceiro vai executar neste processo de negócio, sendo que a empresa ‘’A’’ vai executar o papel de comprador (*Role tp:name="buyer"*) e a empresa ‘’B’’ o papel de vendedor (*Role tp:name="seller"*).

Estas regras estão definidas no documento referenciado pelo atributo *href=http://ebxml.org/processes/buySell.xml#buyer* ,com relação à especificação do papel de comprador, e pelo atributo *href="http://ebxml.org/processes/buySell.xml#seller*, com relação a especificação do papel de vendedor.

2 Os documentos CPP e CPA utilizados nos testes

Estão listados abaixo os CPP utilizados no cenário de testes do sistema, assim como o CPA gerado.

2.1 O CPP utilizado pela empresa A:

```
<?xml version="1.0" encoding="UTF-8" ?>
- <tp:CollaborationProtocolProfile
    xmlns:tp="http://www.ebxml.org/namespaces/tradePartner"
    xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
    xsi:schemaLocation="http://www.ebxml.org/namespaces/tradePartner
    http://ebxml.org/project_teams/trade_partner/cpp-cpa-v1_0.xsd"
    tp:version="1.1">
- <tp:PartyInfo>
  <tp:PartyId tp:type="empresaA">23452356</tp:PartyId>
  <tp:PartyRef tp:href="http://example.com/about.html" />
- <tp:CollaborationRole tp:id="N00">
   <tp:ProcessSpecification tp:version="1.0" tp:name="buySell" xlink:type="simple"
    xlink:href="http://www.ebxml.org/processes/buySell.xml" />
            <tp:Role tp:name="buyer" xlink:type="simple"
    xlink:href="http://ebxml.org/processes/buySell.xml#buyer" />
  - <tp:ServiceBinding tp:channelId="N04" tp:packageId="N0402">
  <tp:Service
    tp:type="uriReference">uri:example.com/services/buyerService</tp:Service>
    </tp:ServiceBinding>
    </tp:CollaborationRole>
- <tp:DeliveryChannel tp:channelId="N04" tp:transportId="N05"
    tp:docExchangeId="N06">
  </tp:DeliveryChannel>
```

```
- <tp:Transport  tp:transportId="N05">
  <tp:SendingProtocol tp:version="1.1">HTTP</tp:SendingProtocol>
  <tp:ReceivingProtocol tp:version="1.1">HTTP</tp:ReceivingProtocol>
     <tp:Endpoint      tp:uri="https://www.empresaA.com.br/ebxml/ebxmlhandler"
    tp:type="allPurpose" />
   </tp:Transport>
- <tp:Transport tp:transportId="N08">
  <tp:SendingProtocol tp:version="1.1">HTTP</tp:SendingProtocol>
  <tp:ReceivingProtocol tp:version="1.1">SMTP</tp:ReceivingProtocol>
  <tp:Endpoint tp:uri="mailto:ebxmlhandler@example.com" tp:type="allPurpose" />
     </tp:Transport>
- <tp:DocExchange tp:docExchangeId="N06">
- <tp:ebXMLBinding tp:version="0.98b">
-              <tp:ReliableMessaging              tp:deliverySemantics="OnceAndOnlyOnce"
    tp:idempotency="true" tp:messageOrderSemantics="Guaranteed">
  <tp:Retries>5</tp:Retries>
  <tp:RetryInterval>30</tp:RetryInterval>
  <tp:PersistDuration>P1D</tp:PersistDuration>
     </tp:ReliableMessaging>
   - </tp:ebXMLBinding>
    </tp:DocExchange>
    </tp:PartyInfo>
- <tp:Packaging tp:id="N0402">
  <tp:ProcessingCapabilities tp:parse="true" tp:generate="true" />
- <tp:SimplePart tp:id="N40" tp:mimetype="text/xml">
 <tp:NamespaceSupported
    tp:location="http://ebxml.org/project_teams/transport/messageService.xsd"
    tp:version="0.98b">http://www.ebxml.org/namespaces/messageService</tp
    :NamespaceSupported>
 <tp:NamespaceSupported
    tp:location="http://ebxml.org/project_teams/transport/xmldsig-core-
    schema.xsd"
    tp:version="1.0">http://www.w3.org/2000/09/xmldsig</tp:NamespaceSuppor
    ted>
     </tp:SimplePart>
- <tp:SimplePart tp:id="N41" tp:mimetype="text/xml">
```

<tp:NamespaceSupported tp:location="**http://ebxml.org/processes/buysell.xsd**" tp:version="**1.0**">**http://ebxml.org/processes/buysell.xsd**</tp:NamespaceSupported>

</tp:SimplePart>

- <tp:CompositeList>

- <tp:Composite tp:id="**N42**" tp:mimetype="**multipart/related**" tp:mimeparameters="**type=text/xml**">

<tp:Constituent tp:idref="**N40**" />

<tp:Constituent tp:idref="**N41**" />

</tp:Composite>

</tp:CompositeList>

</tp:Packaging>

<tp:Comment tp:xml_lang="**en-us**">**buy/sell agreement between example.com and contrived-example.com**</tp:Comment>

</tp:CollaborationProtocolProfile>

2.2 O CPP utilizado pela empresa B:

<?xml version="1.0" encoding="UTF-8" ?>

-<tp:CollaborationProtocolProfile
xmlns:tp="**http://www.ebxml.org/namespaces/tradePartner**"
xmlns:xsi="**http://www.w3.org/2000/10/XMLSchema-instance**"
xmlns:xlink="**http://www.w3.org/1999/xlink**"
xmlns:ds="**http://www.w3.org/2000/09/xmldsig#**"
xsi:schemaLocation="**http://www.ebxml.org/namespaces/tradePartner http://ebxml.org/project_teams/trade_partner/cpp-cpa-v1_0.xsd**"
tp:version="**1.1**">

- <tp:PartyInfo>

<tp:PartyId tp:type="**empresaB**">**235936148**</tp:PartyId>

<tp:PartyRef tp:href="**http://example.com/about.html**" />

- <tp:CollaborationRole tp:id="**N00**">

<tp:ProcessSpecification tp:version="**1.0**" tp:name="**buySell**" xlink:type="**simple**" xlink:href="**http://www.ebxml.org/processes/buySell.xml**" />

<tp:Role tp:name="**seller**" xlink:type="**simple**" xlink:href="**http://ebxml.org/processes/buySell.xml#buyer**" />

- <tp:ServiceBinding tp:channelId="**N34**" tp:packageId="**N0402**">

<tp:Service tp:type="**uriReference**">**uri:example.com/services/buyerService**</tp:Service>

</tp:ServiceBinding>

</tp:CollaborationRole>

```
- <tp:DeliveryChannel tp:channelId="N34" tp:transportId="N35"
    tp:docExchangeId="N36">
  </tp:DeliveryChannel>
- <tp:Transport tp:transportId="N35">
  <tp:SendingProtocol tp:version="1.1">HTTP</tp:SendingProtocol>
  <tp:ReceivingProtocol tp:version="1.1">HTTP</tp:ReceivingProtocol>
    <tp:Endpoint tp:uri="https://www.empresaB.com.br/ebxml/ebxmlhandler"
    tp:type="allPurpose" />
  </tp:Transport>
- <tp:DocExchange tp:docExchangeId="N36">
- <tp:ebXMLBinding tp:version="0.98b">
- <tp:ReliableMessaging tp:deliverySemantics="OnceAndOnlyOnce"
    tp:idempotency="true" tp:messageOrderSemantics="Guaranteed">
  <tp:Retries>5</tp:Retries>
  <tp:RetryInterval>30</tp:RetryInterval>
  <tp:PersistDuration>P1D</tp:PersistDuration>
    </tp:ReliableMessaging>
  - </tp:ebXMLBinding>
   </tp:DocExchange>
   </tp:PartyInfo>
- <tp:Packaging tp:id="N0402">
  <tp:ProcessingCapabilities tp:parse="true" tp:generate="true" />
- <tp:SimplePart tp:id="N40" tp:mimetype="text/xml">
 <tp:NamespaceSupported
    tp:location="http://ebxml.org/project_teams/transport/messageService.xsd"
    tp:version="0.98b">http://www.ebxml.org/namespaces/messageService</tp
    :NamespaceSupported>
 <tp:NamespaceSupported
    tp:location="http://ebxml.org/project_teams/transport/xmldsig-core-
    schema.xsd"
    tp:version="1.0">http://www.w3.org/2000/09/xmldsig</tp:NamespaceSuppor
    ted>
    </tp:SimplePart>
- <tp:SimplePart tp:id="N41" tp:mimetype="text/xml">
   <tp:NamespaceSupported tp:location="http://ebxml.org/processes/buysell.xsd"
    tp:version="1.0">http://ebxml.org/processes/buysell.xsd</tp:NamespaceSupp
    orted>
    </tp:SimplePart>
```

```
- <tp:CompositeList>
- <tp:Composite tp:id="N42" tp:mimetype="multipart/related"
  tp:mimeparameters="type=text/xml">
  <tp:Constituent tp:idref="N40" />
  <tp:Constituent tp:idref="N41" />
  </tp:Composite>
  </tp:CompositeList>
  </tp:Packaging>
  </tp:CollaborationProtocolProfile>
```

2.3 O CPA gerado

O CPA gerado, a partir das interações efetuadas através do protótipo, para estabelecer um acordo de colaboração utilizando o processo de negócio padrão, de compra e venda (*buy/sell*):

```
<?xml version="1.0" ?>
-<tp:CollaborationProtocolAgreement
  xmlns:tp="http://www.ebxml.org/namespaces/tradePartner"
  xmlns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
  xsi:schemaLocation="http://www.ebxml.org/namespaces/tradePartner
  http://ebxml.org/project_teams/trade_partner/cpp-cpa-v1_0.xsd"
  xmlns:xlink="http://www.w3.org/1999/xlink"
  xmlns:ds="http://www.w3.org/2000/09/xmldsig#"
  tp:cpaid="uri:yoursandmycpa" tp:version="1.2">
  <tp:Status tp:value="aproved" />
  <tp:Start>2006-12-26T08:00</tp:Start>
  <tp:End>2007-12-26T08:00</tp:End>
  <tp:ConversationConstraints tp:invocationLimit="100"
  tp:concurrentConversations="100" />
- <tp:PartyInfo>
  <tp:PartyId tp:type="empresaA">234523456</tp:PartyId>
  <tp:PartyRef xlink:href="http://example.com/about.html" />
- <tp:CollaborationRole tp:id="N00">
  <tp:ProcessSpecification tp:version="1.0" tp:name="buySell" xlink:type="simple"
  xlink:href="http://www.ebxml.org/processes/buySell.xml" />
  <tp:Role tp:name="buyer" xlink:type="simple"
  xlink:href="http://ebxml.org/processes/buySell.xml#buyer" />
- <tp:ServiceBinding tp:channelId="N04" tp:packageId="N0402">
  <tp:Service
  tp:type="uriReference">uri:example.com/services/buyerService</tp:Service>
```

```
    </tp:ServiceBinding>
    </tp:CollaborationRole>
- <tp:DeliveryChannel tp:channelId="N04" tp:transportId="N05"
    tp:docExchangeId="N06">
  </tp:DeliveryChannel>
- <tp:Transport tp:transportId="N05">
  <tp:SendingProtocol tp:version="1.1">HTTP</tp:SendingProtocol>
  <tp:ReceivingProtocol tp:version="1.1">HTTP</tp:ReceivingProtocol>
      <tp:Endpoint tp:uri="https://www.example.com/servlets/ebxmlhandler"
    tp:type="allPurpose" />
   </tp:Transport>
- <tp:Transport tp:transportId="N08">
  <tp:SendingProtocol tp:version="1.1">HTTP</tp:SendingProtocol>
  <tp:ReceivingProtocol tp:version="1.1">SMTP</tp:ReceivingProtocol>
  <tp:Endpoint tp:uri="mailto:ebxmlhandler@example.com" tp:type="allPurpose" />
    </tp:Transport>
- <tp:DocExchange tp:docExchangeId="N06">
- <tp:ebXMLBinding tp:version="0.98b">
-       <tp:ReliableMessaging tp:deliverySemantics="OnceAndOnlyOnce"
    tp:idempotency="true" tp:messageOrderSemantics="Guaranteed">
  <tp:Retries>5</tp:Retries>
  <tp:RetryInterval>30</tp:RetryInterval>
  <tp:PersistDuration>P1D</tp:PersistDuration>
    </tp:ReliableMessaging>
  - </tp:ebXMLBinding>
   </tp:DocExchange>
   </tp:PartyInfo>
- <tp:PartyInfo>
  <tp:PartyId tp:type="empresaB">235936148</tp:PartyId>
      <tp:PartyRef xlink:type="simple" xlink:href="http://contrived-
    example.com/about.html" />
  - <tp:CollaborationRole tp:id="N30">
    <tp:ProcessSpecification tp:version="1.0" tp:name="buySell" xlink:type="simple"
    xlink:href="http://www.ebxml.org/processes/buySell.xml" />
        <tp:Role tp:name="seller" xlink:type="simple"
    xlink:href="http://ebxml.org/processes/buySell.xml#seller" />
```

<tp:ServiceBinding tp:channelId="**N34**" tp:packageId="**N0402**">

<tp:Service tp:type="**uriReference**">**uri:example.com/services/buyerService**</tp:Service>

</tp:ServiceBinding>

</tp:CollaborationRole>

<tp:DeliveryChannel tp:channelId="**N34**" tp:transportId="**N35**" tp:docExchangeId="**N36**">

</tp:DeliveryChannel>

<tp:Transport tp:transportId="**N35**">

<tp:SendingProtocol tp:version="**1.1**">**HTTP**</tp:SendingProtocol>

<tp:ReceivingProtocol tp:version="**1.1**">**HTTP**</tp:ReceivingProtocol>

<tp:Endpoint tp:uri="**https://www.example.com/servlets/ebxmlhandler**" tp:type="**allPurpose**" />

</tp:Transport>

<tp:DocExchange tp:docExchangeId="**N36**">

<tp:ebXMLBinding tp:version="**0.98b**">

<tp:ReliableMessaging tp:deliverySemantics="**OnceAndOnlyOnce**" tp:idempotency="**true**" tp:messageOrderSemantics="**Guaranteed**">

<tp:Retries>**5**</tp:Retries>

<tp:RetryInterval>**30**</tp:RetryInterval>

<tp:PersistDuration>**P1D**</tp:PersistDuration>

</tp:ReliableMessaging>

</tp:ebXMLBinding>

</tp:DocExchange>

</tp:PartyInfo>

<tp:Packaging tp:id="**N0402**">

<tp:ProcessingCapabilities tp:parse="**true**" tp:generate="**true**" />

<tp:SimplePart tp:id="**N40**" tp:mimetype="**text/xml**">

<tp:NamespaceSupported tp:location="**http://ebxml.org/project_teams/transport/messageService.xsd**" tp:version="**0.98b**">**http://www.ebxml.org/namespaces/messageService**</tp:NamespaceSupported>

<tp:NamespaceSupported tp:location="**http://ebxml.org/project_teams/transport/xmldsig-core-schema.xsd**" tp:version="**1.0**">**http://www.w3.org/2000/09/xmldsig**</tp:NamespaceSupported>

</tp:SimplePart>

```
- <tp:SimplePart tp:id="N41" tp:mimetype="text/xml">
    <tp:NamespaceSupported tp:location="http://ebxml.org/processes/buysell.xsd"
     tp:version="1.0">http://ebxml.org/processes/buysell.xsd</tp:NamespaceSupp
     orted>
    </tp:SimplePart>
- <tp:CompositeList>
-       <tp:Composite tp:id="N42" tp:mimetype="multipart/related"
     tp:mimeparameters="type=text/xml">
   <tp:Constituent tp:idref="N40" />
   <tp:Constituent tp:idref="N41" />
    </tp:Composite>
    </tp:CompositeList>
    </tp:Packaging>
  <tp:Comment xml:lang="en-us">buy/sell agreement between example.com and
     contrived-example.com</tp:Comment>
    </tp:CollaborationProtocolAgreement
```

3 Considerações sobre o código

Para o desenvolvimento das classes de tratamento dos documentos XML, utilizamos os pacotes de classes abaixo, onde os dois primeiros pacotes estão incluídos no conjunto padrão da API JAXP (*Java API for XML Processing*):

```
import javax.xml.parsers.*;
import org.w3c.dom.*;
import java.io.*;
import java.net.*;
import java.util.* ;
```

A implementação do método lerMensagem da classe ControleDeMensagem foi efetuada conforme o código abaixo, onde é efetuado o *parsing* da mensagem recebida e a geração do documento DOM:

```
package controlarMensagens;
public class ControleDeMensagem {
private String path;
public ControleDeMensagem(String xmlPath) {
        path = xmlPath;
 }
```

```
public Mensagem lerMensagem throws Exception {
        // Cria um DocumentBuilderFactory
        DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
        // cria um DocumentBuilder
        DocumentBuilder db = dbf.newDocumentBuilder();
        Document doc = db.parse(path);
        return doc;
 }
```

A geração do documento relativo ao CPA recebido é implementado através do método lerCpa da classe ControleDeProposta. A criação do documento DOM é efetuada de forma similar a criação do documento de mensagem, mas neste caso a árvore com os objetos permanece em memória, para ser manipulada principalmente no processo de comparação de elementos e atributos do documento recebido da empresa B, com elementos e atributos da empresa A.

```
DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
DocumentBuilder db = dbf.newDocumentBuilder();
// faz o parser do arquivo - Cpa recebido
Document cpa = db.parse(CpaRx));
```

Após a criação o documento está pronto para sofrer as operações de extração de informação de seus elementos.

Conforme a especificação DOM, todas as partes do documento são representadas por um objeto do tipo Node. Utilizando-se o Node do tipo Element e outras operações disponibilizadas pela API, acessa-se um elemento ou atributo no documento, assim como o seu conteúdo:

```
Element docB = cpa.getDocumentElement();
String tagBps = docB.getElementByTagName("ProcessSpecification");
String tagRole =  docB.getAttribute(Role, "name");
```

Para recuperar os atributos de um determinado elemento utilizamos:

```
NamedNodeMap map = docB.getAttibutes("ProcessSpecification");
Node db = map.getNamedItem("name") // atributo name do elemento ProcessSpecification
String conteúdo = db.getNodeValue();  // recupera valor para utilizar na comparação
```

Para a comparação de elementos foi utilizada a classe NodeComparator e o valor de retorno é utilizado para popular a tela de comparação de elementos, conforme o fragmento de codigo a seguir:

```
NodeComparator comparator = new NodeComparator();
if (comparator.compare (docB.getElementByTagName(Retries),
docA.getElementByTagName(Retries) ) == 0 ) {
```

```
    return 1;  // elementos são iguais
}
  else
        { return 0; } // elementos são diferentes
```

Com relação à apresentação, as telas do sistema foram elaboradas em HTML, com relação à parte estática e JSP (Java Server Pages), com relação à parte dinâmica das informações.

4 Mensagens do Protocolo de Mensagens

Na seqüência estão listadas alguns exemplos de mensagens trocadas entre empresa A e empresa B, durante o processo de negociação:
- mensagem de oferta da empresa B

```
<?xml version="1.0" encoding="UTF-8"?>
<msgsNegociacao xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="E:\EsquemaXML\esquema_msgsNegociacao_V5.xml
" nrNegociacao="001">
        <mensagem nrSequencia="001">
                <Oferta tipo="1">
                        <origem>
                                <empresa>Empresa B</empresa>
                                <endereco>Rua Inteiro s/n</endereco>
                                <url>http://www.empresab.com</url>
                        </origem>
                        <destino>
                                <empresa>Empresa A</empresa>
                                <url>http://www.empresaa.com</url>
                        </destino>
                </Oferta>
        </mensagem>
</msgsNegociacao>
```

- Mensagem de contra-oferta:

```
<?xml version="1.0" encoding="UTF-8"?>
<msgsNegociacao xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\Dissertação 2005_2006\Esquema XML da Tese
2006 \esquema_msgsNegociacao_V5.xml" nrNegociacao="001">
        <mensagem nrSequencia="001">
                <ContraOferta tipo="2">
                        <origem>
                                <empresa>Empresa A</empresa>
                                <endereco>Rua String s/n</endereco>
```

```
                    <url>http://www.empresaa.com</url>
                </origem>
                <destino>
                    <empresa>Empresa B</empresa>
                    <url>http://www.empresab.com</url>
                </destino>
            </ContraOferta>
        </mensagem>
</msgsNegociacao>
```

- Mensagem de aceite:

```
<?xml version="1.0" encoding="UTF-8"?>
<msgsNegociacao xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation=" C:\Dissertação 2005_2006\Esquema XML da Tese
2006 \esquema_msgsNegociacao_V5.xml " nrNegociacao="001">
        <mensagem nrSequencia="006">
            <aceitaOferta tipo="5">
                <origem>
                    <empresa>Empresa A</empresa>
                    <endereco>Rua String s/n</endereco>
                    <url>http://www.empresaa.com</url>
                </origem>
                <destino>
                    <empresa>Empresa B</empresa>
                    <url>http://www.empresab.com</url>
                </destino>
            </aceitaOferta>
        </mensagem>
</msgsNegociacao>
```