UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE CIÊNCIAS E TECNOLOGIA
COORDENAÇÃO DE PÓS-GRADUAÇÃO
EM INFORMÁTICA

Dissertação de Mestrado

SEI-Com
Um Serviço de Informações Comerciais na Web com características ativas

Claudionor Alencar do Nascimento

Campina Grande - PB.
Março de 1999

UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE CIÊNCIAS E TECNOLOGIA
COORDENAÇÃO DE PÓS-GRADUAÇÃO
EM INFORMÁTICA

Claudionor Alencar do Nascimento

SEI-Com - Um Serviço De Informações Comerciais na Web Com Características Ativas

Dissertação apresentada ao Curso de Mestrado em Informática da Universidade Federal da Paraíba como requisito parcial à obtenção do título de Mestre em Informática

Área de Concentração: Ciência da Computação
Linha de Pesquisa : Sistemas de Informação e Banco de Dados
Orientador: Prof. Ulrich Schiel

Campina Grande
Universidade Federal da Paraíba
Março/99

N244s Nascimento, Claudionor Alencar do
SEI-Com : um servico de informacoes comerciais na web com caracteristicas ativas / Claudionor Alencar do Nascimento. - Campina Grande, 1999.
75 f.

Dissertacao (Mestrado em Informatica) - Universidade Federal da Paraiba, Centro de Ciencias e Tecnologia.

1. Banco de Dados Ativos 2. Sistemas de Informacao 3. Web 4. Comercio Eletronico 5. Dissertacao - Informatica I. Schiel, Ulrich II. Universidade Federal da Paraiba - Campina Grande (PB) III. Título

CDU 004.65(043)

SEI-COM – UM SERVIÇO DE INFORMAÇÕES COMERCIAIS NA WEB COM CARACTERÍSTICAS ATIVAS

CLAUDIONOR ALENCAR DO NASCIMENTO

DISSERTAÇÃO APROVADA EM 05.03.1999

PROF. ULRICH SCHIEL , Dr.
Orientador

PROF. MARCUS COSTA SAMPAIO, Dr.
Examinador

PROF. DÉCIO FONSECA, Dr
Examinador

CAMPINA GRANDE – PB

Em memória de minha mãe

Maria José Alencar do Nascimento

AGRADECIMENTOS

À Deus pelo dom da vida.

Aos meus pais Francisco e Maria que me colocaram no mundo, me deram à vida, me mostraram os caminhos e me ensinaram a traçá-los.

À minha amada esposa Lourdes pelo incentivo, companheirismo, amor e paciência.

Aos meus filhos Rafael e Rogério por tornarem a minha vida mais alegre.

Ao Prof. Dr. Ulrich Schiel pela orientação nesse trabalho, pelo apoio, pela amizade e por não me deixar fraquejar em momentos difíceis que atravessei.

Ao Prof. Albos Rodrigues pela incomensurável ajuda na concepção desse trabalho.

Ao Prof. Dr. Arturo Hernandez pelas sugestões para o engrandecimento deste trabalho.

Ao Prof. Dr. Marcus Sampaio pelos ensinamentos inestimáveis ao longo do curso.

Aos meus colegas de curso, em especial ao Jean Gonzaga, Salete, Liliane, Daniele, Marcão, Alexandre e Flávia, pelo companheirismo e amizade.

A todos os funcionários do DSC e COPIN por terem me recebido com presteza e amizade, em especial à Zeneide, Emanuela , Arnaldo ,Aninha e Vera.

À Capes pelo apoio financeiro ao longo deste trabalho.

RESUMO

Após apresentarmos algumas tecnologias utilizadas para se fazer acesso a banco de dados via Internet, discutirmos algumas arquiteturas de integração da Web com banco de dados e destacarmos a fundamental importância do desenvolvimento comercial na Web, analisando alguns serviços de informações na Web existentes, apresentamos o SEI-Com, um serviço de informações comerciais na Web, com características ativas, que visa prover empresas com subsídios básicos à tomada de decisões e clientes com serviços de informações comerciais especializados agrupadas em um único site.

ABSTRACT

After showing some technologies used to access databases through the Internet, we discuss some architectures of integration of the Web with databases and we highlight the fundamental importance of the commercial development in the Web, analising some services of commercial information in the Web. We present the SEI-Com, a service of commercial information in the Web, with active characteristics, in order to provide companies with basic subsidies to the taking of decisions and customers with specialized services of commercial information contained in an only site.

SUMÁRIO

LISTA DE FIGURAS

1. INTRODUÇÃO

1.1 MOTIVAÇÃO E OBJETIVOS

Através da história, cada salto na disponibilidade de informações, bem como no acesso a elas, tem iluminado as pessoas e expandido as oportunidades de negócios. Além disso a informação tem se mostrado uma inigualável fonte de poder, tanto para indivíduos como para empresas.

Com o advento da Internet e sua rápida disseminação em todo o mundo, a busca por informações tem crescido a cada momento. No entanto, a grande diversidade de informações existentes aliada às limitações físicas que a Internet ainda tem, faz com que a busca por informações específicas seja extremamente demorada e que o resultado de uma pesquisa geralmente traga uma infinidade de informações irrelevantes, sendo necessária a filtragem das mesmas até que as informações irrelevantes sejam desconsideradas.

O Serviço de Informações Comerciais, o SEI-Com, foi desenvolvido para que empresas e consumidores possam obter informações dos diversos ramos do comércio agrupadas em um único serviço online. Empresas poderão utilizá-lo para obter o *perfil* dos usuários de determinada localidade, além de poder acompanhar a sua posição perante a concorrência obtendo dessa forma subsídios básicos para a avaliação da presença de seus produtos no mercado, tornando-o uma ferramenta útil à *tomada de decisões*. Aos consumidores o SEI-Com fornecerá diversos serviços que possibilitarão a obtenção de orçamentos de suas listas de compras, consultas diversas e compras *online*. O serviço de *Orçamentos* dará ao cliente a possibilidade de saber onde efetuar suas compras de forma que se minimize o custo da mesma. O comportamento do cliente no sistema é registrado para que se possa oferecê-lo um interface personalizada, sugerindo opções padrões para facilitar e tornar mais rápida a navegação no sistema.

1.2 ESTRUTURA DA DISSERTAÇÃO

O restante desse documento está organizado como segue:

No Capítulo 2 apresentamos diversas tecnologias que tornam possível a integração da Web com banco de dados[Lim97, Kim96] .

No Capítulo 3 apresentamos o estado atual de desenvolvimento do Comércio Eletrônico, analisando o que vem sendo feito para prover segurança às transações comerciais[1] online e descrevendo brevemente como a Web está se tornando o meio ideal de comunicação e excelente local para a realização de negócios. Descrevemos também alguns serviços de informações comerciais, nacionais e internacionais, mostrando suas principais características para motivar o desenvolvimento de um novo serviço que aproveite algumas das melhores características, o que é o objetivo principal desse trabalho.

No Capítulo 4 descrevemos o SEI-Com, um serviço de informações na Web com características ativas.

No Capítulo 5 apresentamos os detalhes de implementação do SEI-Com bem como o seu funcionamento e interface. Além disso, descrevemos as estruturas dos dados mantidos no SEI-Com e principalmente como fizemos para que novos ramos comerciais sejam tratados pelos diversos serviços sem a necessidade de alterações na estrutura das tabelas existentes.

Finalmente no Capítulo 6 apresentamos as conclusões desta dissertação.

---

[1] *Transação Comercial, nesse contexto, significa uma operação comercial entre uma empresa e um cliente. Quando estivermos nos referindo às transações de um SGBD utilizaremos simplesmente o termo transação*

2. WEB E BANCO DE DADOS

Esse capítulo tem por objetivo apresentar os principais componentes do ambiente de integração da Web com banco de dados, mostrando as interfaces utilizadas para se fazer acesso aos servidores Web, analisando algumas das principais linguagens de programação utilizadas nesse ambiente e mostrando a arquitetura de integração da Web com banco de dados a ser utilizada [Lim96].

2.1 WORLD WIDE WEB

A *World Wide Web,* ou simplesmente Web, é um recurso de informação globalmente distribuído residindo sob a rede mundial de computadores, a *Internet* [Lim96], contendo uma grande quantidade de informações relevantes em todas as áreas do conhecimento humano.

2.1.1 Arquitetura Web

A Figura 2.1 mostra a arquitetura simplificada da Web, que é uma típica arquitetura cliente/servidor[Moh98].

Os componentes dessa arquitetura são:

- **Cliente** – Composto por *browsers* Web[BCL+94], capazes de exibir e solicitar documentos sob a rede. Caso o *browser* não consiga interpretar algum tipo de dado, ao cliente podem ser adicionados *aplicativos externos* para realizar essa função.
- ***Servidor Web*** cuja principal função é atender os pedidos dos clientes Web por documentos armazenados no *sistema de arquivos* da plataforma onde se encontra instalado. Dependendo do pedido do cliente Web, o servidor Web pode disparar uma aplicação externa como, por exemplo, a execução de um programa via interface padrão CGI *(Common Gateway Interface)* [Moh97, Win97, Gen97].

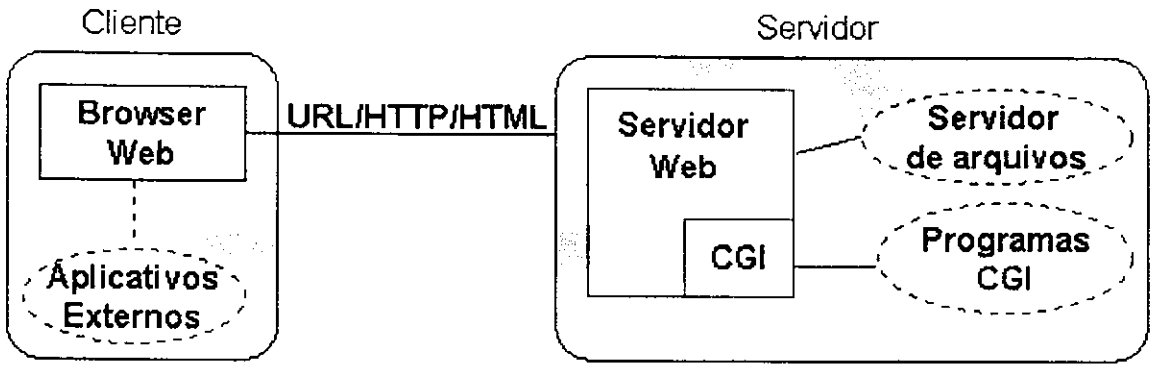

Figura 2.1 – Arquitetura Web simplificada.

Uma importante característica da Web é que ela foi projetada para funcionar no "topo" da *Internet*, que é composta por uma grande variedade de computadores e redes que interagem entre si. Dessa forma a Web incorpora naturalmente a característica de ser um ambiente distribuído e multi-plataforma.

Uma Segunda característica importante é que a Web foi projetada para encapsular vários protocolos *Internet* incluindo, entre outros, *FTP(File Transfer Protocol)*, GOPHER, WAIS(Wide Area Information Server), NNTP e TELNET[Lim96,Moh97]. Assim é eliminada a necessidade de se usar um programa separado para cada serviço *Internet* diferente, unificando-os num único serviço, no caso a Web.

Uma terceira característica importante da Web é que ela é baseada em três padrões abertos que permitem **localizar** a informação, **transferir** a informação e **descrever** a formatação da informação distribuída pela rede :

1. ***Localizar a informação:*** Para a identificação e localização de documentos Web distribuídos pela *Internet* foi proposta a utilização de um formato para endereçamento de documentos Web denominado URL *(Uniform Resource Locator)* [BMM94, Moh98].

2. ***Transferir a informação:*** Para o transporte da informação entre o servidor e o cliente Web foi proposto um protocolo de comunicação denominado HTTP *(HyperText Transfer Protocol)* [W3C97d]. Sua principal característica é ser um protocolo aberto e especializado na transmissão de documentos Web sob a

Internet. O protocolo HTTP é um protocolo *stateless*, ao contrário de diversos outros protocolos *Internet* que são *stateful*. Isto significa que o cliente Web envia o pedido de uma operação ao servidor Web, este atende o pedido e logo em seguida é encerrada a conexão. Nenhuma informação sobre a solicitação do usuário é mantida no servidor Web. A característica *stateless* do protocolo HTTP proporciona eficiência e velocidade necessárias a sistemas de informação *hipermídia* distribuídos como a Web, mas origina vários problemas para o desenvolvimento de aplicações que exijam, por exemplo, a identificação do usuário.

3. ***Descrever a formatação da informação:*** Para a apresentação e formatação de documentos na Web é utilizada a linguagem padrão HTML *(HyperText Markup Language)* [W3C97e, Moh98], que permite que a estrutura dos documentos Web, bem como os *links* a outros documentos e recursos da *Internet,* sejam incorporados diretamente no formato texto. Documentos codificados em HTML podem ser interpretados pelos *browsers* e formatados segundo as características de cada plataforma em que são exibidos.

A comunicação entre cliente e servidor Web é sempre feita na forma de pares requisição/resposta e é sempre iniciada pelo cliente que envia uma mensagem de ao servidor pedindo abertura de conexão. O servidor responde aceitando ou não este pedido. Se aceitar, a conexão é estabelecida.

Uma vez estabelecida a conexão, o cliente envia uma mensagem com a requisição feita pelo usuário, representada por uma URL. Esta mensagem é enviada através da rede para o servidor, que busca a informação no disco local ou dispara a execução de uma aplicação externa. Após o processamento do pedido, uma mensagem de resposta é enviada ao cliente que, ao recebê-la, retorna um pedido de fechamento de conexão.

Quando o cliente recebe a informação solicitada, é necessário verificar se os dados são válidos, através de informações contidas em campos do cabeçalho da mensagem. Somente após esta verificação o cliente pode mostrar o documento para

o usuário. Para tanto, ele precisa saber qual o tipo e o formato do arquivo. Esta informação também é obtida no cabeçalho da mensagem. Todo este processamento impõe um atraso no cliente, que é refletido no tempo de resposta observado pelo usuário.

2.2 INTERFACES DE ACESSO A SERVIDORES WEB

Nessa seção apresentamos algumas interfaces de acesso a um servidor Web atualmente utilizadas.

2.2.1 A interface CGI

Antes mesmo do surgimento da Web já existiam muitos serviços de informações *on line,* como catálogos de bibliotecas, serviços de recuperação de informações e acesso a Banco de dados. Para proporcionar serviços dinâmicos como estes no ambiente Web pode-se usar programas auxiliares(*scripts)* que são executados na plataforma do servidor quando solicitados pelo usuário. Os scripts são programas que podem ser executados pelo servidor Web em resposta a um pedido do cliente, tendo como principal objetivo a geração de informações dinâmicas.

Cabe ao servidor Web as seguintes tarefas quando inicia um programa *script:*

1. Avaliar se o programa *script* pode ser executado. Por exemplo, o servidor NCSA HTTPd [NCSA97c] para sistemas UNIX, requer que os programas *scripts* estejam localizados em um diretório específico *("/*cgi-bin" por exemplo), com permissão de execução apropriados. Outros servidores UNIX podem ter regras diferentes. Em servidores Web para Windows NT pode-se executar programas *scripts* em qualquer diretório desde que ele tenha extensão ".EXE", por exemplo.

2. Iniciar o programa *script* e garantir que os dados de entrada do cliente Web serão passados para o programa *script.*

3. Receber os dados retornados pelo programa *script* e os enviar de volta ao cliente

Web.

4. Enviar uma mensagem de volta ao cliente Web se algo de errado acontecer com o programa *script*.

5. Encerrar a conexão de rede apropriadamente, quando o programa *script* finalizar sua execução.

Para a implementação destas ações foi especificada a interface CGI *(Common Gateway Inteface)*. CGI é uma interface padrão, implementada pela maioria dos servidores Web existentes, para a execução de programas *scripts* externos ao servidor Web. A interface descreve como os programas *scripts* serão inicializados e como os dados serão passados entre estes e o servidor Web.

Os detalhes de implementação das regras CGI são dependentes do sistema operacional em que reside o servidor Web. Para sistemas *UNIX,* por exemplo, os programas *scripts* são executados como processos separados, os dados são enviados pelo servidor Web para a entrada padrão *(stdin)* e os resultados são retornados ao servidor Web através da saída padrão *(stdout)*. Para servidores *Windows,* o programa *script* é executado como uma nova aplicação e os dados são passados através de arquivos temporários.

A principal vantagem de se escrever aplicações Web através da interface CGI é a portabilidade da aplicação. Apesar disso, existem algumas limitações no uso da interface CGI. A principal delas está relacionada com problemas de desempenho, especialmente em aplicações multi-usuário. Por exemplo, aplicações CGI não podem ser compartilhadas por vários usuários clientes. Mesmo se uma aplicação CGI ainda estiver sendo executada, quando da chegada de novos pedidos ao servidor Web, este inicia um novo processo da aplicação CGI. Quanto mais pedidos forem chegando, mais processos concorrentes serão criados na plataforma do servidor. O fato de se criar um processo para cada pedido consome tempo e grande soma de recursos de memória, comprometendo o desempenho da aplicação.

2.2.2 SSIs e APIs de servidores Web

Além da interface CGI, existem outras formas de se gerar documentos Web[Den96], dentre elas citamos as *SSIs (Server Side Includes)* e as *APIs(Application Programiming Interface*) de servidores Web que serão descritas nas duas seções seguintes.

2.2.2.1 *SSIs*

SSIs são trechos de códigos, também denominados *tags,* inseridos em um documento Web e que serão executados pelo servidor Web no momento em que um cliente solicita o documento. É uma funcionalidade proprietária de servidores Web presentes, por exemplo, no servidor *NCSA HTTPd* [NCSA97] e no servidor *WebQuest* [WebQ97].

Apesar da simplicidade no desenvolvimento, existem duas grandes desvantagens de se usar SSIs para o desenvolvimento de aplicações na Web. Em primeiro lugar o servidor Web deve verificar em cada documento Web solicitado a existência de códigos SSIs. Isto pode afetar o desempenho do servidor. Segundo, é um mecanismo não muito seguro sob o ponto de vista de acesso aos dados. Algumas *tags* SSIs podem permitir que usuários clientes executem códigos na plataforma do servidor Web sem qualquer restrição, como por exemplo, por meio da *tag* 'EXEC'.

Para amenizar estes problemas os servidores Web podem ser configurados de forma a limitar o uso de algumas *tags* SSIs a alguns diretórios, ou até mesmo a completa desativação de *tags* selecionadas. Pode-se também, configurar o servidor Web de forma a limitar o número de documentos Web para os quais o servidor deva verificar a existência de código SSI. Por exemplo, pode-se configurar o servidor Web para analisar a presença ou não de *tags* SSIs nos documentos que tenham somente a extensão ".shtml".

2.2.2.2 *APIs*

Uma API de servidor Web é semelhante a uma API para desenvolvimento de aplicações em Banco de dados cliente/servidor[Gen97]. Trata-se de um conjunto de funções que são incorporadas ao servidor Web com o objetivo de estender suas funcionalidades. As APIs podem fazer tudo o que a interface CGI faz e muito mais, permitindo, entre outras coisas, melhorar significativamente o desempenho de aplicações externas à Web. São exemplos de servidores Web que incorporam APIs os servidores da *Netscape* [Net97], o servidor *IIS* da Microsoft [Mic97]

As APIs resolvem os problemas de limitação e de eficiência comuns às aplicações que usam a funcionalidade CGI. Dentre estes destacam-se:

1. Os programas escritos como APIs de servidores Web ficam residentes em memória uma vez iniciados e, portanto, executam muito mais rápido do que quando a interface CGI é usada. Uma solução integrada ao servidor Web como a proporcionada pelas APIs é bem mais eficiente[Den96]. De fato, mais de um cliente Web pode executar a mesma função API simultaneamente, e ao contrário de programas CGI, não é inicializado um novo processo para cada usuário distinto. Ao invés disso somente novas variáveis de memória são inicializadas.

2. As APIs permitem o compartilhamento de dados e recursos de comunicação com o servidor Web, ao contrário de programas CGI. Aplicações APIs executam no mesmo espaço de memória em que o servidor Web está executando e, em algumas implementações [Net97], pode-se ter acesso inclusive às estruturas de dados internas do servidor Web.

3. A interface CGI é limitada ao envio e retorno de dados de/para o servidor Web. As APIs estendem esta característica permitindo, por exemplo, o desenvolvimento de funções para controlar o acesso a um Banco de dados, lidar com erros específicos de execução de programas ou ainda o desenvolvimento de mecanismos auxiliares para a autenticação de clientes Web.

As diferenças funcionais entre programas APIs de servidores Web e programas

CGIs são:

1. Uma aplicação API recebe os dados do servidor Web por meio de canais de comunicação próprios da implementação do servidor Web e não da entrada padrão *(stdin)* como nos programas que usam a interface CGI. Por exemplo, na plataforma Windows o servidor IIS da *Microsoft* utiliza *estruturas de dados* denominadas ECB *(Extension_Control Block)* [Mic97]. Todas as variáveis ambientes enviadas pelo cliente Web são recuperadas, neste caso, em estruturas ECBs, ao contrário de programas CGI, que devem recuperá-las por meio de funções específicas da linguagem de programação utilizada.

2. O resultado da execução da aplicação API não é enviado para a saída padrão *(stdout)* como nos programas CGI, mas sim recuperados por meio de chamadas de funções APIs escritas especificamente para este fim.

Uma vez executada pelo servidor Web a aplicação API fica residente no mesmo espaço de memória utilizado pelo servidor Web, esperando por novos pedidos do cliente.

Apesar das várias vantagens de se usar APIs para o desenvolvimento de aplicações, existem algumas desvantagens:

1. São dependentes do servidor Web onde são implementadas.

2. Podem inabilitar o servidor Web *(crash)* uma vez que elas compartilham dos mesmos recursos de memória que o servidor Web e, geralmente, tem total acesso às estruturas de dados internos do servidor Web e a funções de I/O que são utilizadas, por exemplo, para escrever no espaço da memória principal.

3. Podem afetar o desempenho do servidor Web, principalmente se os recursos de CPU forem limitados.

4. A programação por meio de APIs não é uma tarefa muito trivial, exigindo do desenvolvedor conhecimentos sobre técnicas de programação como multiprocessamento, sincronização de concorrência, programação de protocolos

de rede e tratamento de exceção estruturada, entre outros.

Concluindo, SSI é uma boa técnica para o desenvolvimento de aplicações Web dinâmicas, mas é um mecanismo proprietário, limitado e que pode afetar o desempenho do servidor Web. Já as APIs são mais flexíveis e permitem estender as funcionalidades dos servidores Web porém, também são proprietárias, requerem um profundo conhecimento do funcionamento do servidor Web, apresentam alto risco operacional de inabilitar o servidor Web e o desenvolvimento de aplicações APIs pode ser demorado se comparado a aplicações que usam a interface CGI.

2.3 LINGUAGENS DE PROGRAMAÇÃO PARA A WEB.

São várias as linguagens de programação existentes que podem ser usadas no ambiente Web[2], como por exemplo, as linguagens tradicionais como C e C++, e linguagens emergentes[Moh97] como Java, JavaScript, VBScript e PERL.

Especialmente interessantes hoje na Web são as linguagens ditas de "código móvel" [W3C97b]. Estas linguagens, que são transportadas pela rede e interpretadas pelo cliente Web, estão alterando a maneira de se construir aplicações Web. A seguir são analisadas brevemente três dessas linguagens: Java, JavaScript e VBScript.

2.3.1 A linguagem Java

Java é uma linguagem que vem sendo desenvolvida pela *Sun Microsystems* desde 1991 e foi projetada para ser usada em ambientes distribuídos, oferecendo suporte para execução em diferentes ambientes de redes, múltiplos sistemas operacionais e arquiteturas de *hardware*. Com o surgimento da Web e sua conseqüente expansão, os criadores de Java perceberam que as características da linguagem a colocava como uma linguagem ideal para o ambiente Web. Assim, surgiu o primeiro *browser*

---

[2] *A linguagem HTML não se enquadra nessa categoria por servir apenas para a criação de páginas Web e por não ter nenhum poder computacional.*

a suportar a linguagem, conhecido como *Hotjava.* Em 1993, o *browser Netscape Navigator,* ofereceu suporte para a linguagem e desde então sua aceitação não para de crescer, sendo atualmente suportada por diversos outros *browsers* comerciais, como o *internet Explorer* da *Microsoft.*

Java é uma linguagem orientada a objeto, com excelentes características de segurança, semelhante à linguagem C++ mas sem algumas das complexidades desta linguagem, como ponteiros e alocação de memória. A principal característica da linguagem é, no entanto, a portabilidade entre plataformas distintas, o que pode ser explicado devido à sua arquitetura [Sun98].

Os principais componentes da linguagem Java são um compilador Java, que gera o código fonte Java para uma linguagem intermediária, independente da máquina, denominada *"bytecodes"* e um interpretador Java, que interpreta e executa os *bytecodes* Java (descarregados sob a rede) nos clientes Web. Daí sua principal característica de portabilidade em diferentes plataformas. Para que a linguagem Java execute em uma dada plataforma basta que o interpretador para aquela plataforma esteja presente.

Os programas Java, também denominados *"applets",* permitem estender a linguagem HTML de modo a permitir a construção de aplicações diversas com controle total da interatividade com o usuário. Estas características têm dado à linguagem quase um *status* de linguagem "padrão" na Web.

2.3.2 As linguagens JavaScript e VBScript

Alguns *browsers* Web comerciais disponibilizam linguagens interpretadas projetadas especificamente para o ambiente Web. É o caso, por exemplo, da linguagem *JavaScript,* incorporada no *browser Nestscape Navigator* e *Visual Basic Scripting Edition(VBScript)* incorporada ao *browser Internet Explorer.* Essas linguagens procuram resolver os problemas da falta de funcionalidade e limitações da linguagem HTML no lado cliente [Rah96].

O código destas linguagens é inserido dentro de páginas HTML e interpretado pelos *browsers*. Como Java, estas linguagens permitem que clientes Web individualmente realizem tarefas de processamento, tais como verificação de campos de entrada, aliviando a carga de processamento do servidor Web.

Existem vários problemas em se usar estas linguagens, como por exemplo, serem linguagens proprietárias e terem restrições de plataforma. *VBScript* ainda não é suportada na plataforma UNIX, ao passo que *JavaScript* já funciona em plataformas Solaris, Sun OS, Linux e Windows.

2.4 Arquiteturas de Integração Web Banco de Dados

A integração SGBD e Web já é hoje uma realidade dando suporte aos mais diversos tipos de aplicações, comerciais ou não. Existem diversos *softwares* ou *gateways* de integração Web e Banco de Dados[Int96, Moh97, Key97] que visam tirar proveito do ambiente aberto e multi-plataforma da Web e das facilidades de gerenciamento e mecanismos de otimização para acesso aos dados presente nos SGBDs.

Nessa seção iremos apresentar uma classificação dos diversos gateways em categorias com base na localidade em que o software integrador é executado: se no lado do cliente Web ou no lado do servidor Web [Kim96, Lim96].

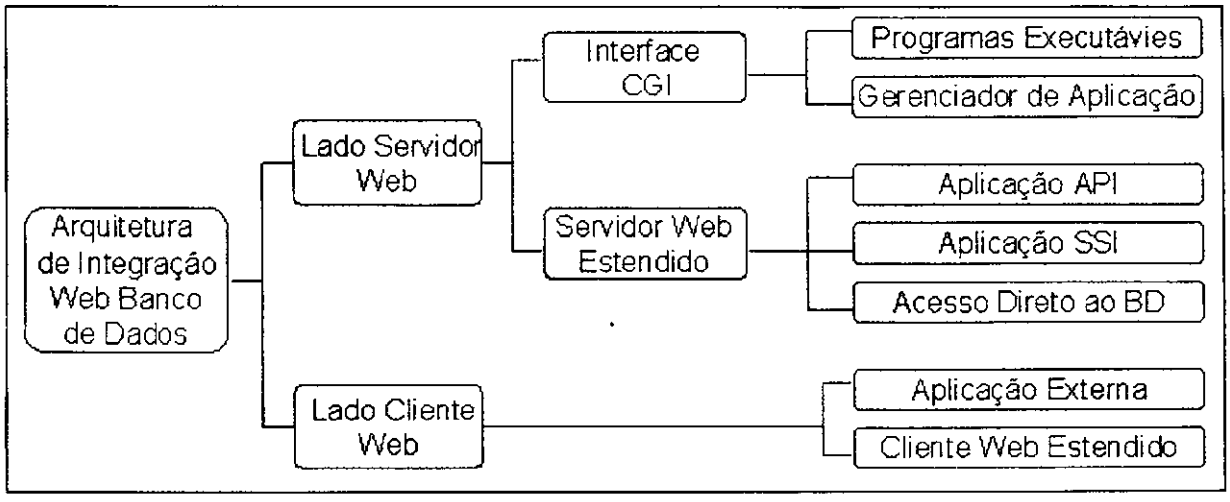

Figura 2.2 – Taxonomia das arquiteturas dos *gateways* Web Banco de Dados.

2.4.1 Arquiteturas no Lado Servidor Web

As arquiteturas de integração Web e banco de dados localizados no lado do servidor têm como principal característica o fato de serem semelhantes à arquitetura cliente/servidor de banco de dados em três camadas a saber:

- **Cliente Web**: Onde reside a lógica da apresentação, que controla a interação entre o usuário e o computador. Nessa camada pode-se ter alguma lógica da aplicação, tais como validação de dados implementados por linguagens como *Java, JavaScript* ou *VBScript.*

- **Servidor Web/Aplicação cliente do banco de dados:** Esta camada é composta pelo servidor Web e aplicação que acessa o banco de dados. Ela contém a lógica da aplicação, responsável pelas decisões, cálculos e operações que a aplicação deve realizar, mecanismos de segurança para acesso à aplicação, e controle de acesso simultâneo de muitos usuários à aplicação.

- **Servidor de Banco de Dados:** Esta camada consiste do servidor de banco de dados, tipicamente rodando em máquinas de alto desempenho.

A seguir é apresentado um esboço genérico das arquiteturas localizadas no lado do servidor Web, segundo as três camadas descritas anteriormente.

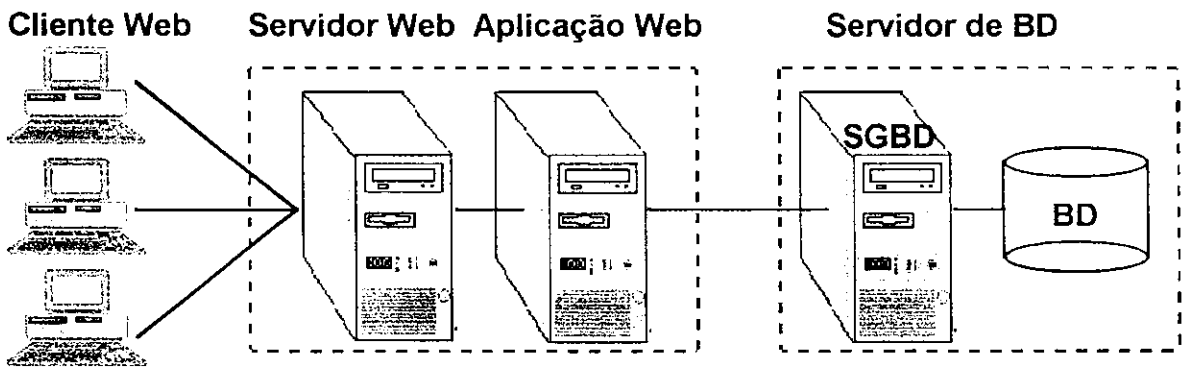

Figura 2.3 – Arquitetura de três camadas dos *gatways* no lado do servidor Web.

O código da aplicação reside em um único local, de forma que toda a manutenção no código da aplicação se reflete imediatamente a todos os clientes Web.

Na seção 2.5.1.1 apresentamos as arquiteturas que usam a interface CGI para a execução da aplicação e na seção 2.5.1.2 serão apresentadas as arquiteturas que usam servidores Web estendidos para a execução da aplicação.

2.4.1.1 *Arquiteturas que usam a Interface CGI*

Como visto anteriormente a interface CGI possibilita a execução de aplicações externas ao servidor Web. Dessa forma, ela é a maneira mais natural para o desenvolvimento de aplicações na Web com acesso a banco de dados, já que são aplicações externas ao ambiente cliente/servidor Web. Apesar da implementação da interface CGI diferir em alguns servidores Web as arquiteturas dos *gateways* que Web/banco de dados usam essa interface são considerados os mais portáveis que existem atualmente.

2.4.1.1.1 Programas Executáveis CGI

Esta arquitetura é composta por um ou mais programas CGI. Geralmente implementados usando-se alguma linguagem de programação hospedeira do SGBD. Esta arquitetura é a mais imediata para se desenvolver aplicações Web banco de dados, pois para usar a interface CGI implementada pelos servidores Web é necessário um linguagem que possa ler da entrada padrão, obter as variáveis de ambiente e escrever o resultado na saída padrão. Como a grande maioria dos SGBDs existentes incorporam alguma linguagem de programação com tais características, praticamente todos eles podem se comunicar com a Web por meio de **Programas Executáveis CGI** sem grandes custos adicionais.

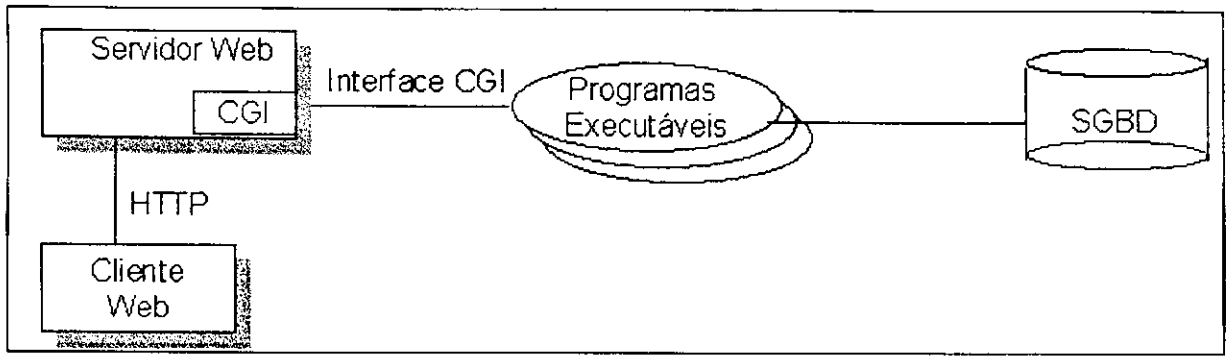

Figura 2.4 – Arquitetura Programas Executáveis CGI.

O funcionamento da arquitetura é mostrado a seguir:

1. O cliente Web solicita ao servidor Web a execução de uma aplicação externa, que neste caso é um programa executável na plataforma do servidor Web.

2. O servidor Web dispara um processo para a execução do programa através da interface padrão CGI, enviando-lhe os dados recebidos do cliente Web de acordo com a implementação do servidor Web para a interface CGI.

3. O programa recebe os dados passados pelo servidor Web via interface CGI, decodifica-os, formula o comando SQL e abre um conexão com o Banco de dados para sua execução.

4. O SGBD retorna os dados e a conexão é finalizada. O programa executável trata os dados recebidos e os repassa ao servidor Web via interface CGI, num formato que o cliente Web entenda, como por exemplo, no formato HTML. O processo iniciado pelo servidor Web é finalizado neste instante.

5. O servidor Web retorna os dados repassados pelo programa ao cliente Web.

A maioria dos *Gateways* existentes hoje em dia estão implementados segundo esta arquitetura. São exemplos os *gateways* **DB2WWW** Connection, ColdFusion, Web.sql [Moh97, Key97], e OpenBase[Man].

2.4.1.1.2 Gerenciador de Aplicação *CGI*

De forma a suplantar os diversos problemas da arquitetura CGI anterior pode-se implementar uma arquitetura de integração Web banco de dados dividida em dois módulos: vários despachantes (*dispatchers*), que são pequenos programas executáveis, e um(ou mais) gerenciador que coordena a aplicação. Esta arquitetura da origem à arquitetura de integração denominada ***Gerenciador de Aplicação CGI.*** A Figura 2.5 ilustra esta arquitetura.

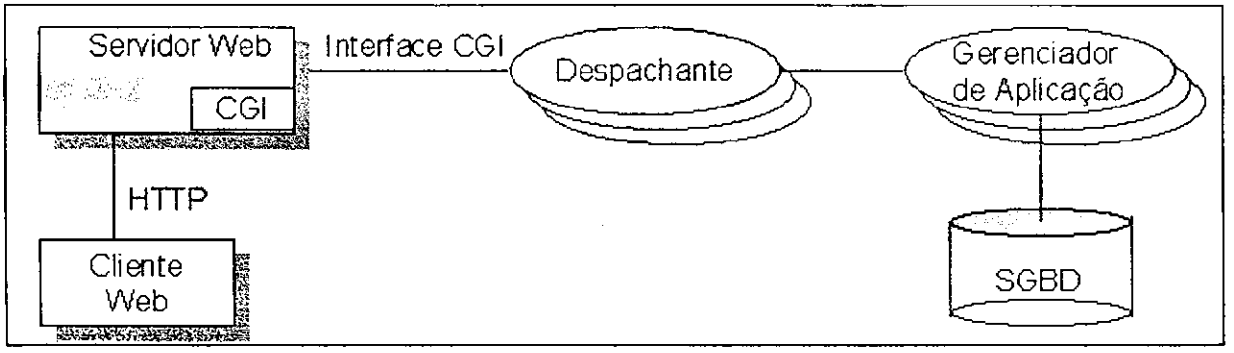

Figura 2.5– Arquitetura *Gerenciador de Aplicação CGI*

O funcionamento da arquitetura é mostrado a seguir:

1. O cliente Web solicita ao servidor Web a execução de um aplicativo externo que neste caso é um *Gerenciador de Aplicação* CGI.

2. O servidor Web inicia um processo para um dos despachantes que compõe o primeiro módulo da aplicação através da interface padrão CGI, enviando os dados passados pelo cliente Web de acordo com a implementação do servidor Web para interface CGI.

3. O despachante identifica qual *Gerenciador de Aplicação* pode processar o pedido do servidor Web, transfere em seguida os dados ao gerenciador escolhido e fica aguardando a resposta.

4. O *Gerenciador de Aplicação* escolhido começa a executar sob o SGBD o pedido solicitado pelo despachante e, ao contrário da arquitetura *Programas Executáveis* CGI, não finaliza a conexão com o Banco de dados. O *Gerenciador de Aplicação*

fica esperando por novos pedidos de despachantes relativos ao usuário cliente que está utilizando a aplicação.

5. O despachante recebe os dados do *Gerenciador de Aplicação* e repassa-os ao servidor Web via interface CGI. O processo iniciado pelo servidor Web para a execução do despachante é finalizado neste momento.

6. O servidor Web retorna os dados ao cliente Web .

Um despachante CGI pode ser implementado sem nenhum conhecimento do SGBD. Analogamente os *Gerenciadores de Aplicação* são implementados sem conhecimento da interface CGI. Como os *Gerenciadores de Aplicação* mantêm a conexão com o Banco de dados aberta após a execução de algum pedido do despachante, pode-se tomar vantagens integrais do esforço de otimização do SGBD, além de ser possível fazer o gerenciamento do estado da aplicação de forma eficiente.

São exemplos de *Gateways* que se enquadram nesta arquitetura os *softwares* Web Connect [key97] e O2Web [Moh97].

2.4.1.2 *Servidor Web estendido*

As arquiteturas que se encontram nessa categoria não usam a interface padrão CGI para a execução das aplicações Web banco de dados. Ao contrário, as funcionalidades do servidor Web são estendidas de forma a proporcionar acesso mais otimizado e mais rápido a aplicações externas[Rei96]. Ganha-se portanto em desempenho quando comparado às arquiteturas que usam a interface CGI, mas perde-se em portabilidade da aplicação, ficando o desenvolvedor limitado às características do servidor Web proprietário.

2.4.1.2.1 Aplicação API

Vimos que as APIs possibilitam a extensão das funcionalidades dos servidores Web. Um destas funcionalidades é a possibilidade de se escrever aplicações APIs

para acesso a um Banco de dados dando origem à arquitetura denominada ***Aplicação API***. A Figura 2.6 ilustra esta arquitetura.

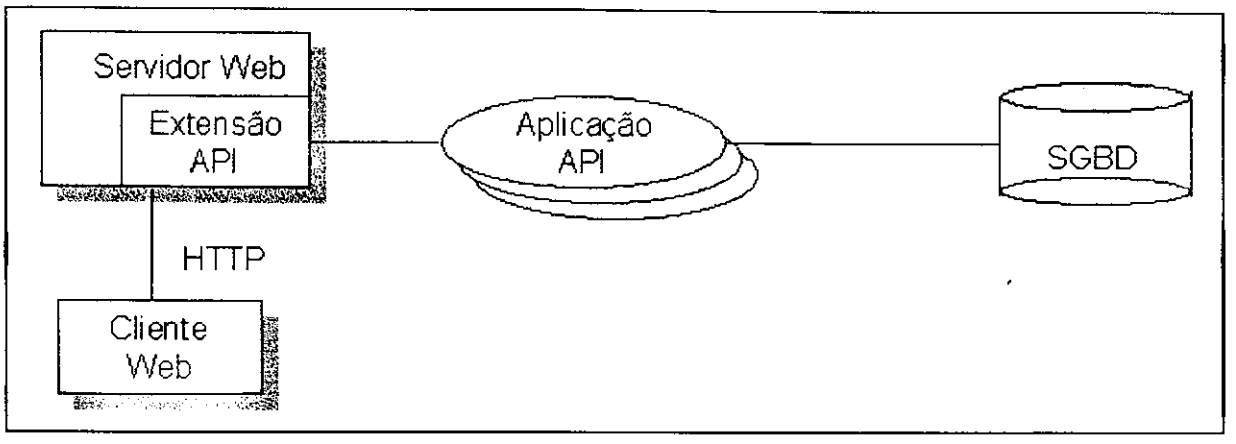

Figura 2.6 Arquitetura *Aplicação API.*

O funcionamento da arquitetura é mostrado a seguir.

1. O cliente Web solicita ao servidor Web a execução de uma aplicação API, que pode ou não já estar sendo executada no mesmo espaço de memória em que o servidor Web está executando.

2. O servidor Web inicia a execução da aplicação API se ela ainda não estiver sendo executada e repassa os dados enviados pelo cliente Web de acordo com a implementação API do servidor Web, ou seja, não é usado a interface CGI.

3. A aplicação API inicia a conexão com o banco de dados ou, dependendo da implementação da aplicação, pode-se substituir um despachante CGI(como na arquitetura anterior) por um despachante API que envia um pedido para um *Gerenciador de Aplicação.*

4. A aplicação API realiza a conexão com o Banco de dados e, em seguida, repassa os dados ao servidor Web por canais próprios da implementação API do servidor Web.

5. O servidor Web retorna os dados ao cliente Web. A aplicação API normalmente fica residente em memória esperando por novos pedidos até que o servidor Web

decida quando ela pode ser finalizada para liberar recursos do sistema.

Muitos produtos comerciais que usam uma das duas arquiteturas descritas anteriormente possuem uma de forma a suportar uma ou mais APIs de servidores Web proprietários, como por exemplo, ColdFusion, NetImpact Dynamo, Web.sql, NetDynamics, que já suportam as APIs NSAPI e ISAPI e OpenBase que suporta ISAPI. Nesse sentido, esses *gateways* podem ser classificados em uma ou outra arquitetura, dependendo de como são executados pelo servidor Web.

2.4.1.2.2 Aplicação SSI

Na seção 2.2.2.2 vimos que a funcionalidade SSI implementada por alguns servidores Web permite a inserção de códigos(ou *tags*) especiais em documentos HTML que são interpretados e executados automaticamente pelo servidor Web quando solicitado pelo cliente Web. Alguns desses códigos podem incluir funcionalidades para a execução de comandos em um SGBD, dando origem à arquitetura de integração Web com Banco de dados denominada ***Aplicação SSI.*** A Figura 2.7 ilustra esta arquitetura .

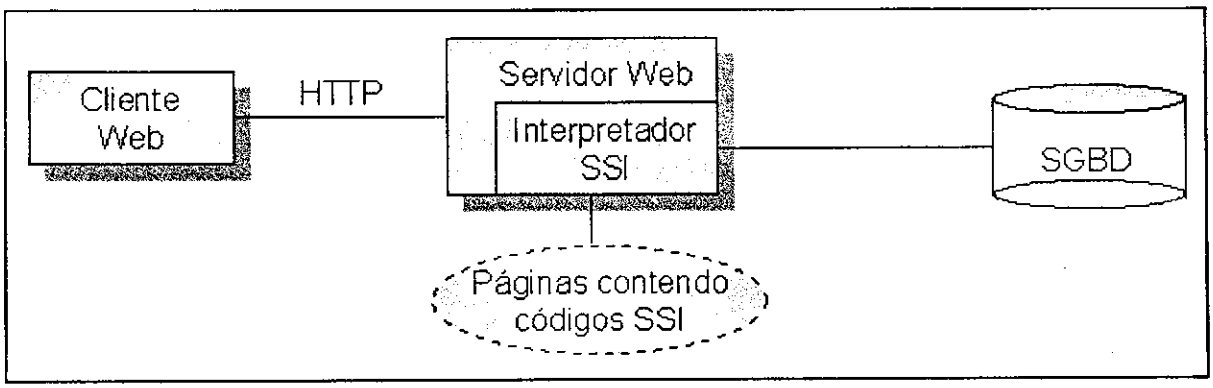

Figura 2.7 – Arquitetura *Aplicação SSI.*

O funcionamento dessa arquitetura é mostrado a seguir:

1. O cliente Web solicita uma página que possui, além dos elementos da linguagem HTML, códigos SSI para a conexão e execução de comandos em um SGBD.

2. O servidor Web recupera a página e reconhece que a página contém códigos SSI

para serem executados dinamicamente. Neste momento o servidor Web ativa o interpretador SSI, que é um de seus componentes.

3. O servidor Web se conecta ao SGBD apropriado e solicita a consulta. Os dados são retornados pelo SGBD e o servidor Web os formata de acordo com outros códigos SSI de formatação presentes no restante da página HTML.

4. O servidor Web encerra a conexão com o Banco de dados e retorna a página de dados ao cliente Web.

Um exemplo de servidor Web que se enquadra nesta arquitetura é o servidor WebQuest[Web97].

2.4.1.2.3 Acesso Direto ao Banco de Dados

O servidor Web pode ser implementado para, além de atender pedidos HTTP, atender pedidos que exijam acesso ao Banco de dados sem nenhuma interferência da interface CGI ou dos mecanismos API ou SSI já descritos. Ele incorpora o protocolo de comunicação do SGBD e adiciona mecanismos próprios para o desenvolvimento de aplicações Web Banco de dados totalmente integradas ao SGBD. Dessa forma o servidor Web funciona como um cliente do SGBD que suporta o protocolo HTTP. Esta forma de conexão origina a arquitetura denominada *Acesso Direto ao Banco de dados.* A Figura 2.8 ilustra essa arquitetura.

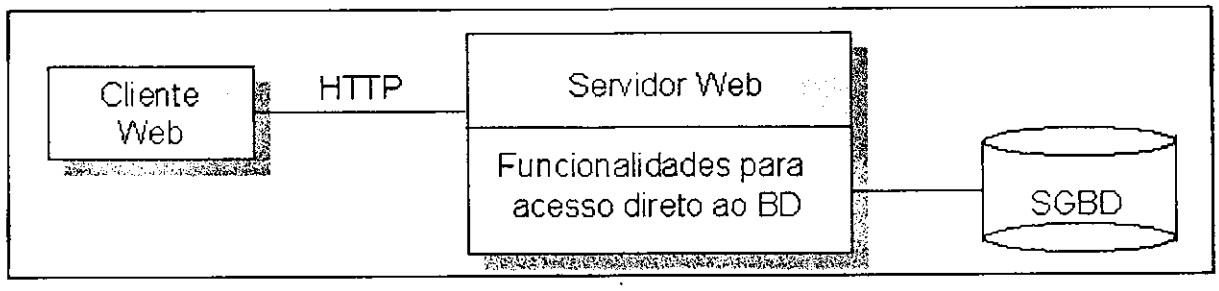

Figura 2.8 – Arquitetura *Acesso Direto ao Banco de Dados*

O funcionamento desta arquitetura é mostrado a seguir:

1. O cliente Web solicita ao servidor Web a execução de uma aplicação

incorporada ao próprio servidor Web.

2. O servidor Web faz uma série de verificações internas(*login e password, por exemplo*) e inicia uma conexão com o Banco de dados como se fosse mais um cliente do SGBD.

3. O SGBD atende o pedido e retorna os dados ao servidor Web.

4. O servidor Web formata os dados de acordo com a aplicação nele desenvolvida e retorna os dados ao cliente Web. Normalmente o servidor Web fica esperando por novos pedidos do cliente e não libera os recursos da sessão do usuário no SGBD enquanto o cliente Web estiver utilizando a aplicação.

Esta arquitetura pode ser usada também para transpor para o ambiente Web as aplicações de Banco de Dados cliente/servidor[MSA97]. Neste caso o servidor Web emula uma sessão de usuário para uma aplicação já existente como se fosse um cliente normal da aplicação.

São exemplos desta arquitetura os servidores CorpWeb Server[MSA97] e AOLServer [Ame97].

2.4.1.3 *Arquiteturas no lado do cliente Web*

As arquiteturas de integração Web Banco de Dados que se enquadram nesta categoria têm como principal característica o fato do aplicativo que acessa o Banco de dados usar recursos do lado cliente Web. Embora o código resida no servidor Web todo o processamento da lógica da aplicação é realizado na máquina do cliente Web. Assim o cliente Web alivia o processamento do lado do servidor Web, realizando o processamento da lógica da aplicação Web Banco de Dados.

2.4.1.3.1 Aplicação Externa

Os clientes Web podem ser configurados para utilizar outros aplicativos para a apresentação de dados que eles não sejam capazes de lidar diretamente. Por

Exemplo, a maioria dos clientes Web invoca um aplicativo externo específico quando ele recebe um arquivo de áudio. De forma semelhante pode-se usar esta configuração para associar aplicações de Banco de dados e a Web. Ou seja, a arquitetura de integração pode consistir de uma *Aplicação Externa* ao cliente Web para acesso ao Banco de dados. A Figura 2.9 ilustra esta arquitetura.

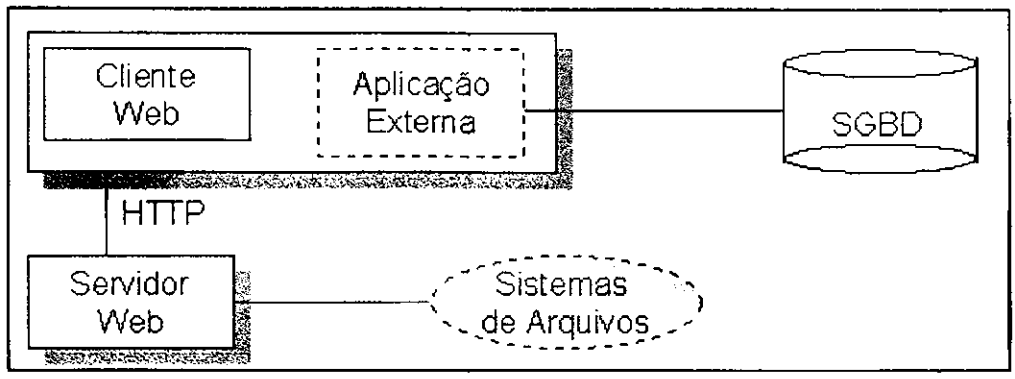

Figura 2.9 – Arquitetura *Aplicação Externa*

O funcionamento da arquitetura é mostrado a seguir:

1. O Cliente Web faz um pedido ao servidor Web.

2. O servidor Web atende o pedido enviando instruções para o cliente Web executar uma *Aplicação Externa*. Geralmente são enviados também arquivos de configuração da aplicação com uma extensão para que a *Aplicação Externa* consiga interpretar.

3. O cliente Web recebe o pedido e imediatamente inicia a execução da *Aplicação Externa* para acesso ao Banco de dados.

4. A *Aplicação Externa* assume o controle da aplicação, se conecta ao Banco de dados, solicita um pedido e recebe os dados enviados pelo SGBD.

5. Os dados são exibidos no cliente Web pela *Aplicação Externa*. A execução da *Aplicação externa* é finalizada quando o usuário seleciona uma opção apropriada no cliente Web.

Um exemplo de *gateway* nesta arquitetura é o PowerBuilder Plug-In[Pow97].

2.4.1.3.2 Cliente Web Estendido

A grande maioria dos clientes Web disponíveis no mercado possuem outras funcionalidades diferente daquela básica de apresentação dos dados recebidos do servidor Web. Essa característica surge diante da grande limitação do cliente Web e de seus componentes, como a linguagem HTML. A extensão do cliente Web para acessar um banco de dados surgiu naturalmente quando da extensão do cliente Web para aceitar linguagens de código móvel. Esta forma de conexão, onde o *browser* é estendido de forma a ser capar de acessar um banco de dados via uma linguagem de código móvel, dá origem à arquitetura *Cliente Web Estendido*. A Figura 2.10 ilustra esta arquitetura.

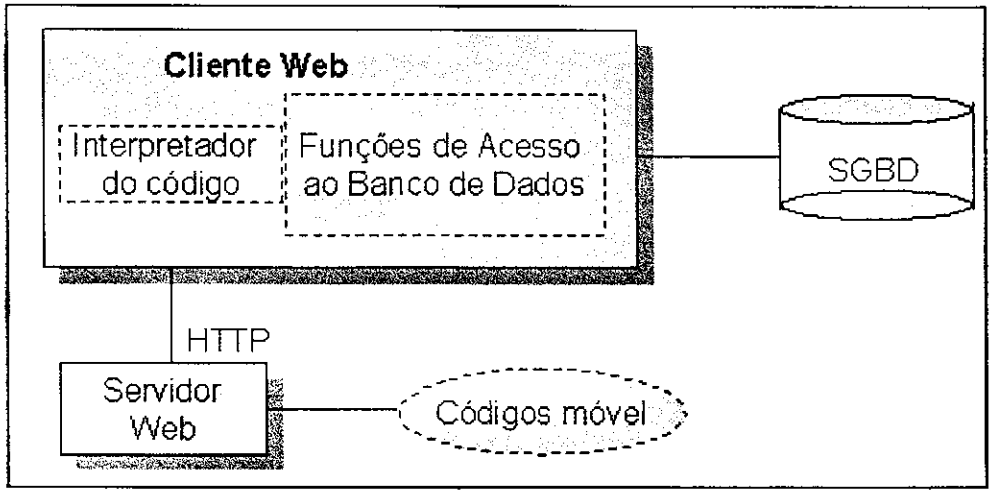

Figura 2.10 – Arquitetura Cliente Web Estendido

O funcionamento da arquitetura é mostrado a seguir:

1. O cliente Web solicita um pedido de um documento que contém ou faz referência ao código da aplicação para acesso ao Banco de dados.

2. O servidor Web envia o documento e o código da aplicação. Juntamente com o código da aplicação pode ser enviado o código para gerenciamento da conexão como o Banco de dado, como por exemplo, um *driver* gerente de conexão

semelhante a um *driver* ODBC[3][Nor96].

3. O cliente Web recebe o documento e o código da aplicação e imediatamente reconhece que deve iniciar a execução do código da aplicação com auxílio de um *interpretador* que é um componente do *browser* Web.

4. O cliente Web se conecta ao Banco de dados usando as funcionalidades de acesso ao Banco de dados enviados junto com o código da aplicação ou incorporados no cliente, executando o pedido do usuário.

5. O SGBD envia o resultado ao cliente Web e este, de acordo como o código da aplicação, exibe o resultado da operação ao usuário. A conexão com o Banco de dados é, geralmente, finalizada após cada pedido Ter sido atendido.

Como exemplo dessa arquitetura existe a interface API JDBC[Sun98] da *Sun Microsystems* que incorpora funções de acesso a Banco de dados implementadas na linguagem Java semelhante à interface ODBC da *Microsoft* . O *gateway* dbANYWHERE server é um exemplo da implementação de JDBC[4] para acesso aos diversos SGBDs disponíveis no mercado.

2.5 CONCLUSÃO

A tecnologia Web e, em particular, a tecnologia de integração Web e bancos de dados ainda está em pleno desenvolvimento e ainda existem diversos problemas técnicos para que a integração seja satisfatória.

O crescimento em larga escala da Internet nos últimos anos tem levado a rede a ser considerada um emergente meio de comunicação para a realização e expansão do comércio eletrônico e de serviços de informações de uma forma geral. A Web, como principal aplicação dentro da Internet, já vem sendo explorada como meio comercial

---

[3] ODBC (Open Database Conectivity) é uma API produzida pela Microsoft para se fazer acesso a banco de dados.

[4] JDBC uma API Java que implementa parte da linguagem SQL para que se possa fazer acesso a banco de dados a partir de programas Java.

tanto para a venda de produtos e serviços, como para propaganda e marketing. Apesar de todo este crescimento e desenvolvimento, a Internet ainda não inspira confiança em consumidores e empresas, haja vista que ela ainda não provê mecanismos que tornem seguras as transações comerciais.

2.6 COMÉRCIO ELETRÔNICO

O NTIA(*National Telecommunications and Information Administration*) [NTI97] define comércio eletrônico de uma maneira bem genérica como qualquer uso da tecnologia eletrônica em uma atividade comercial. Por esta definição, entende-se que esta atividade é muito mais do que simplesmente um "home-shopping" ou a venda de produtos online. Ela pode considerar, por exemplo, a realização de contratos, negociações e marketing eletronicamente.

Em [WB96] é analisado o impacto do mercado eletrônico na distribuição de bens e serviços sobre o aspecto dos custos das transações . Segundo eles, o comércio eletrônico facilitará a coordenação entre comerciantes e consumidores, reduzindo o custo das transações, eliminando a necessidade de intermediários e consequentemente baixando o preço final do produto. Outra análise sobre o impacto da utilização de redes de comunicação nos relacionamentos entre comerciantes e consumidores foi feito em[SKP96]. De um modo geral, o comércio eletrônico facilitará a interação e comunicação das empresas com seus fornecedores, clientes e parceiros comerciais.

O comércio eletrônico pode trazer melhorias para uma empresa aumentando a sua eficiência em encontrar e interagir com consumidores, na comunicação com seus parceiros comerciais e no desenvolvimento de novos produtos. Novos mercados serão abertos e o seu domínio de atuação deixará de ser limitado da sua região geográfica para um mercado global.

2.7 DESENVOLVIMENTO COMERCIAL NA WEB

A popularidade da Web como um meio comercial é devido a sua habilidade de facilitar o compartilhamento global de informações e recursos e o seu potencial em prover um canal eficiente de marketing, de propaganda e de distribuição de produtos e serviços[HP96]. Além disto a eficiência da Web como canal de marketing sugerindo que os resultados são 10 vezes melhor e que os custos são 1/4 mais baixos do que os meios convencionais.

Como meio comercial, a Web pode oferecer benefícios tanto para o lado do consumidor como para o lado das empresas, satisfazendo as necessidades de ambos. Para o consumidor, os benefícios do meio incluem a grande quantidade de informação disponível, ferramentas sofisticadas de buscas e serviços de compra online. Estes serviços podem ajudar na decisão de compra do consumidor. As empresas se beneficiam do potencial da Web como um canal de distribuição, um meio para propaganda e marketing e um mercado totalmente novo para a expansão de seus negócios.

O crescimento rápido da Internet, e particularmente da Web, fez surgir uma grande massa de consumidores e empresas participando de um mercado global online. Todo esse desenvolvimento dentro da Internet tem mostrado que além desta servir como um meio de comunicação ela servirá como um novo mercado de negócios. A adoção da Web como meio comercial fará as empresas experimentarem uma maneira inovadora de marketing para os consumidores em um ambiente totalmente computacional.

Apesar de todo o crescimento do comércio eletrônico na Web, ainda existem grandes limitações no que se refere às transações comercias online. Uma das principais é a que se refere à segurança no tráfego dos dados. Esforços têm sido desprendidos para desenvolver tecnologias para suplantar tal limitação. Na Seção seguinte discutiremos um pouco sobre a segurança na Internet.

2.7.1 Segurança no Comércio Eletrônico via Internet

Apesar do grande aumento no uso da Internet nos últimos anos, consumidores e empresas ainda estão apreensivos quanto a sua utilização como meio de se fazer negócios. A principal causa de tal receio é, sem dúvida, a falta de segurança, haja vista que a Internet foi originalmente desenvolvida para amparar pesquisas e não como um ambiente comercial[Bhi96]

A falta de segurança na Internet expõe os sistemas de informações a ataques de vândalos. Esses ataques podem se manifestar de várias formas, dentre elas citamos:

- Interceptação dos dados(*EavesDropping*) – Nesse tipo de ataque os hackers(piratas virtuais) ficam a espreita de dados que lhes interessem em algum sistema conhecido. Esse ataque em redes pode resultar no roubo de informações bancárias do cliente, tais como número de cartões de crédito, número ou saldo da conta. Similarmente tais ataques podem resultar no acesso, por parte do invasor, a sistemas cujo acesso é restrito a membros cadastrados.

- *Modificação de Dados* – Nesse tipo de ataque os conteúdo de certas transações é modificado. Tal ataque pode ser usado, por exemplo para modificar dados de um pedido ou a quantia em transferências de fundos para uma conta bancária.

- *Spoofing* – Pode ser usado para fazer com que uma parte estranha a uma transação se faça passar por uma das partes envolvidas na mesma. Nesse caso o invasor pode entrar em sites de empresas e obter centenas ou milhares de números de cartões de crédito, número de contas ou qualquer outras informações que, por ventura, esteja guardadas no site.

2.7.1.1 *Requisitos de Segurança em Transações Comerciais*

O uso de *firewalls*[5] em redes conectadas à Internet é comum para tentar provê-las com mecanismos de segurança, no entanto eles não podem ser considerados como solução adequada para suportar com segurança as transações comerciais, haja vista que eles não provêem segurança do início ao fim de uma transação[AJS+97].

Uma solução de segurança para o processamento de transações precisa satisfazer pelo menos os seguintes requisitos:

- Confidencialidade – Toda comunicação em uma transação é restrita às partes envolvidas na mesma. Esse requisito é fundamental na privacidade do usuário, assim como na proteção de dados proprietários. A confidencialidade pode ser obtida com o uso de algoritmos de encriptação[AJS+97].

- Autenticação – Cada uma das partes envolvidas em uma transação devem estar certas que estão se comunicando com que pensam que estão, e não com um intruso. A autenticação é geralmente provida por meio de assinaturas e ou de certificados digitais[Key97].

- Integridade dos dados – Os dados enviados como parte de uma transação não podem ser modificados em transito e nem no armazenamento. Assinaturas digitais e certificados de chave pública podem ser usados para prover a integridade dos dados.

2.7.1.2 *Iniciativas para Prover Segurança em Transações Comerciais*

Diversos protocolos de criptografia têm sido propostos nos últimos anos. Todas as propostas são similares no serviço provido e nos algoritmos de criptografia utilizados, porém eles diferem na forma em que os serviços são providos e em suas localizações com respeito ao resto da pilha do protocolo TCP/IP. Algumas propostas

---

[5] *FireWalls são programas residentes em um computador entre uma rede local e a Internet para filtrar todos os pacotes recebidos*

implementam a segurança na camada do IP, a camada de rede, outras na camada acima da camada TCP, a camada de sessão, ainda outras implementam a segurança dentro de protocolos especifico da camada de aplicação.

O relacionamento entre as diversas iniciativas de segurança e a pilha do protocolo são mostrados na Figura 3.1 e serão descritas a seguir.

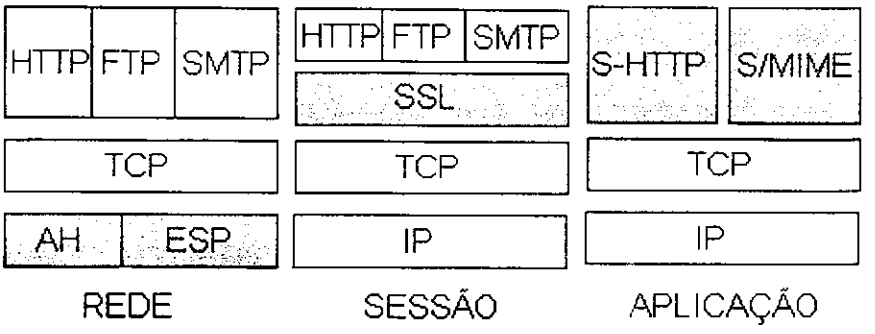

Figura 3.1 – Relacionamento entre as diversas propostas de segurança e a pilha de protocolos

2.7.1.2.1 Soluções para Segurança na Camada de Rede

Desde 1993, têm sido feitos muitos esforços no sentido de desenvolver uma arquitetura para prover segurança no IP visando proteger o tráfego na Internet por meio do uso de técnicas de criptografia avançadas. Esses esforços acabaram dando subsídios para a definição de um protocolo chamado *next-generation protocol* (IPng) que é uma extensão do protocolo IP. Essa Arquitetura inclui dois mecanismos para prover serviços de segurança:

- Autentication Header(AH) – Provê autenticidade e integridade usando o algoritmo MD5(*message-digest algorithm*).
- *Encapsulating Security payload*(EPS) – Provê confidencialidade usando o algoritmo padrão de encriptação de dados(*Data Encryptation Standard* – DES).

---

com o objetivo de reduzir a invasão de intrusos.

2.7.1.2.2 Solução para Segurança na Camada de Sessão

A principal e a mais prevalecente iniciativa para prover segurança implementada na camada de sessão é o *Secure Sockets Layer* (SSL)[Moh96,], introduzido pela Netscape em 1994. SSL provê serviço de segurança na camada acima da camada TCP, usando a combinação de chaves públicas e algoritmos simétricos de criptografia para prover confidencialidade, integridade de dados e autenticação do servidor e do cliente.

2.7.1.2.3 Soluções de Segurança na Camada de Aplicação

Paralelo aos esforços para prover segurança em protocolos da camada de sessão, várias iniciativas vem sendo tomadas para prover segurança em aplicações individuais:

- HTTPS – É uma implementação do protocolo HTTP sobre o SSL, logo a discussão se reduz ao SSL.

- S-HTTP – Foi primeiramente proposta em 1994 pela CommerceNet[ ], um consórcio de Organizações interessadas no comércio eletrônico. Como o SSL, S-HTTP provê confidencialidade, Integridade dos dados e Autenticação do Servidor e, opcionalmente, do cliente. Embora S-HTTP defina uma extensão ao protocolo HTTP provendo a segurança embutida na estrutura do mesmo. S-HTTP é mais flexível do que o SSL em termos dos algoritmos e mecanismos de segurança que ele suporta.

Recentes trabalhos têm sido desenvolvidos para suportar SSL e S-HTTP em uma mesma aplicação.

2.7.1.3 *Protocolos de Pagamento na Internet*

Todas as soluções discutidas nas seções acima tentam incorporar seguridade na pilha TCP/IP. Contudo, um conjunto de protocolos está sendo investigado para prover sistemas de pagamentos seguros na Internet. A maioria desses protocolos

têm como principal foco o desenvolvimento de métodos que tornem segura a transmissão de informações privadas(Exemplo: número de cartão de crédito).

Esses protocolos são específicos para a transmissão segura de informações privadas e independem do meio pelo qual elas são transportadas, isto é, eles podem ser implementados dentro de *browsers* Web usando o protocolo HTTP, em programas de email usando SMTP ou em outra aplicação com algum outro protocolo. O que se deve levar em conta é que o dado envolvido em uma transação deve ser conduzido com segurança meio se o meio não seja seguro.

Nas seções seguintes descreveremos alguns dos protocolos de pagamento mais utilizados atualmente.

2.7.1.3.1 First Virtual

O esquema First Virtual é um esquema de pagamento eletrônico desenvolvido, principalmente para pagamento de produtos que podem ser entregues sobre a Internet, como é o caso dos softwares. Ele não foi desenvolvido para compra de coisas tangíveis, como por exemplo computadores.

Antes de fazer uma compra usando o sistema First Virtual o consumidor deve abrir uma conta no site do mesmo preenchendo um formulário online. Após enviar os seus dados, para abrir a nova conta , o consumidor completa o processo por telefone. Durante esse contato telefônico o mesmo informa o seu número de cartão de crédito, ou outras informações confidenciais, e recebe um número de identificação pessoal(PIN – *Pesonal identification* number). Depois disso, para efetuar uma compra em uma empresa que participa do mesmo sistema, o usuário precisa informar somente o seu PIN. Depois o provedor First Virtual entra em contato com ele por e-mail, dando-lhe a possibilidade de aprovar ou não a compra antes que o seu cartão seja faturado. O consumidor paga $2.00 para abrir uma conta First Virtual. Não é cobrada nenhuma taxa adicional e nenhum software especial é requerido no lado cliente da conexão.

Empresas que desejem aceitar o pagamento por meio do sistema First Virtual

precisam abrir uma conta. First Virtual fornecerá à empresa um software para validar os números de identificação pessoal do usuário e para notificar a First Virtual quando uma compra tiver sido efetuada. Para cada transação o First Virtual cobra a taxa de $0.29, mais 3% do valor da transação.

2.7.1.3.2 DigiCash

DigiCash, um produto desenvolvido pela empresa holandesa DigiCash, é uma espécie de sistema de dinheiro digital que trabalha de forma semelhante a um cartão telefônico. Nesse sistema o usuário compra "CyberBucks"(Dolar Digital) de um banco que suporte o sistema DigiCash. Os *CyberBucks* consistem de uma série de números especialmente codificados. O usuário pode depositar CyberBucks em um banco e sacar o dinheiro correspondente.

Os softwares que suportam o DigiCash previnem as falsificações e também que seja gasto mais que uma vez. Como dinheiro de verdade, DigiCash pode ser gasto anonimamente. Você não precisa ter que se autenticar para gastar ou receber DigiCash, e o seu uso não deixa nenhum rastro. DigiCash pode ser usado para a transmissão de dinheiro com segurança entre consumidores e empresas permitindo que produtos e serviços sejam vendidos por meio da Internet sem que haja a interferência direta de bancos.

Esse sistema requer que a instalação de um software tanto no lado do consumidor quanto no lado da empresa. Ele está disponível nas plataformas Windows NT, Windows 95 e alguns sistemas Unix. O sistema DigiCash ainda é relativamente recente e ainda está sendo amplamente adotado, mas há expectativas de uma grande aceitação no mercado

2.7.1.3.3 CyberCash

CyberCash, um produto da CyberCash Corporation, usa softwares especializados nos lados da empresa e do cliente para prover pagamento seguro pela Internet. Para um consumidor efetuar um pagamento com CyberCash, ele precisa primeiramente fazer o download de um software, distribuido gratuitamente no site da

CyberCash, chamado "Wallet" e inicializá-lo com informações do pagamento e identificação pessoal. As opções de pagamento incluem números do cartão de crédito e débito em conta bancária. O wallet guarda essas informações, de uma forma encriptada, nos computadores do cliente. Quando um usuário vai comprar algo em uma empresa que aceita CyberCash , o software wallet é ativado e solicita ao usuário a escolha da forma de pagamento. Ele pode optar para que lhe seja cobrado pelo cartão de crédito ou por débito automático em conta corrente. O software instalado no lado da empresa valida e registra a transação conectando-se a servidor mantido CyberCash, um processo que leva de 10 a 15 segundos. O Wallet mantém o registro de cada transação, permitindo que o usuário tenha um certo controle de todas elas. CyberCash está disponível para muitas plataformas incluindo Macintosh, Windows 95 e Windows NT.

O sistema usa um forte esquema de criptografia para prevenir que as informações sejam interceptadas por uma terceira parte não autorizada. Além disso, como os números do cartão de crédito e da conta corrente do usuário nunca são armazenados no servidor da empresa, não há chance desses dados serem roubados por indivíduos que conseguem quebrar o sistema de segurança da mesma.

Para que uma empresa passe a usufruir do pagamento por meio do CyberCash ela deve abrir uma conta em um banco que suporte o sistema. São cobradas diversas taxas para ter e manter uma conta CyberCash, na abertura da conta é cobrada uma taxa de aproximadamente $100, para manter aberta a conta é cobrada uma taxa mensal de aproximadamente $15 e de cada transação é cobrada uma taxa de 2 a 3% do valor da mesma. Essas taxas podem variar de uma região para outra. Existem centenas de banco americanos que suportam contas CyberCash e esse número está crescendo continuamente.

Depois de abrir uma conta de CyberCash, a empresa tem que instalar, em seu servidor Web, o software chamado "caixa registrador eletrônico". O caixa registrador eletrônico é distribuído livremente e está disponível para muitas plataformas, incluindo Windows NT e Unix.

2.7.1.3.4 SET

O protocolo SET(*Secure Electronic Transaction*) é um padrão aberto usado para o processamento de transações na Internet criado por um consórcio de empresas formado pela Netscape, Microsoft, Visa e Mastercard.

O padrão SET usa um complexo sistema de certificados digitais para assegurar a identidade de cada parte de uma transação: consumidor, comerciante, administradoras de cartões de crédito e o banco. Para garantir privacidade nas transações, elas são separadas de tal forma que o comerciante tem acesso a informações sobre o que está sendo comprado, qual o preço e se o pagamento está aprovado, mas nenhuma informação de qual método de pagamento o cliente está usando. Da mesma forma, a administradora de cartões tem acesso ao preço da compra, mas não sabe nada sobre o tipo de mercadoria envolvida. Apesar de todas essas precauções o SET não provê o completo anonimato que o sistema DigiCash

2.8 SERVIÇOS DE INFORMAÇÕES COMERCIAIS NA WEB

Nessa seção apresentaremos alguns dos principais serviços de informações comerciais disponíveis na Web, descrevendo suas principais características e funcionamento e por fim fazendo uma análise comparativa dos mesmos para motivar a concepção de um novo serviço de informações comerciais que incorpora algumas das principais características dos mesmos e preencha algumas lacunas deixadas por eles.

Dentre os serviços disponíveis, descreveremos aqui serviços nacionais como UOL-compras online, Cadê?-compras online e MetaMiner assim como serviços internacionais como Acses, Visa Shoping guide, CompareNet e Jango.

2.8.1 MetaMiner

Um sofisticado serviço de informações desenvolvida na UFMG que busca na Web os preços de livros, CD'S e softwares e os compara entre os diversos

estabelecimentos que fornecem o produto requisitado retornando ao usuário uma lista de preços ordenada de forma crescente. Além disso esse serviço pode efetuar buscas genéricas, baseadas em palavras chaves, nos diversos serviços de busca nacionais e internacionais.

Principais Características:

- Lida com limitado número de ramos comerciais, o serviço é capaz de disponibilizar preços somente de Livros, CDs e Softwares.
- Não mantém dados próprios, os dados utilizados pelo serviço são extraídos diretamente do banco de dados das empresas na Web.
- Faz comparação de preços de um produto específico. Após o usuário selecionar o produto desejado, o serviço busca o preço do mesmo em empresas com presença na Web e monta uma lista de preços ordenadas de forma crescente.
- O usuário pode especificar o tempo de espera para obter um retorno da consulta.

2.8.2 Yahoo Shopping Guide

Serviço de informações mantido pela administradora de cartões VISA e mantido Yahoo. O serviço lida com uma diversificada gama de produtos que vai de livros e CDs a flores e cosméticos. Em algumas categorias há apenas indicações de sites que comercializam os produtos. Em outras há sofisticados mecanismos de comparações de preços incluindo o valor do frete.

Principais Características:

- Lida uma grande variedade de produtos.
- Faz comparações do preço de um produto de algumas categorias já incluindo o valor do frete e disponibilidade do mesmo.

- Disponibiliza compras online após o cliente Ter obtido a comparação do preço do produto.

- Pesquisa os preços diretamente dos sites das empresas.

2.8.3 Cadê – Compras online

O Cadê é um agente de buscas nacional que integra em seu site um serviço de informações comerciais que visa disponibilizar informações dos mais diversos ramos do comércio. Em quase sua totalidade, esse serviço disponibiliza apenas endereços de empresas que provêem serviços e/ou produtos de um ramo específico. Em alguns sites são fornecidos serviços de compras online totalmente interativos como é o caso de supermercados da rede "Pão de Açúcar", Sé Supermercados dentre outros.

Principal característica

- Agrupa links de sites de diversos ramos comerciais.

O site não provê uma integração entre as diversas empresas. Com isso cada consulta a um determinado produto ou serviço em uma empresa específica independe totalmente do serviço/produto que outras empresas fornecem. Caso o cliente deseje obter a melhor opção de compra de seus produtos desejado ele precisa navegar por diversos sites de diversas empresas e manualmente verificar qual a melhor opção.

2.8.4 Acses

Um serviço de informações comerciais que visa pesquisar preços de livros em diversas livrarias americanas. Os critérios de buscas são variados podendo pesquisar por título, nome do autor, palavras chaves ou ISBN.

Principais Características:

- Lida somente com livros.

- Compara os preços de um livro em cerca de 25 livrarias americanas, disponibilizando inclusive o prazo estimado de entrega do mesmo.
- Utiliza dados extraídos diretamente do site das empresas.
- Provê um sistema de ajuda refinado que auxilia o usuário menos experiente.

2.8.5 Pão de Açúcar Delivery

Esse é um dos mais sofisticados serviços de informações brasileiros. Possibilita ao cliente a compra de produtos em um dos Supermercados Pão de Açúcar com entrega em diversas cidades de São Paulo e Brasília.

Principais Características:

- Disponibiliza compras online.
- Tem Serviço de Entrega a domicílio
- Lida somente com um estabelecimento comercial.

2.8.6 Comparação dos Serviços

Na tabela 3.1 apresentamos os serviços descritos nessa seção destacando as suas principais características e acrescentando algumas características desejáveis ao desenvolvimento de um novo serviço de informações comerciais, o SEICOM que será descrito no próximo capítulo.

Tabela 3.1 – Comparação dos serviços de informações Comerciais na Web.

Característica /Serviços	MetaMiner	Acses	Cadê	PDA	Yahoo
Compara preços de um produto em várias empresas	Sim	Sim	Não	Não	Sim
Compara preços de uma lista de produtos	Não	Não	Não	Não	Não
Extrai dados do site das empresas.	Sim	Sim	Não	Não	Sim
Armazena informações do produtos fornecidos pelas empresas em seu site.	Não	Não	Não	Não	Não
Disponibiliza links para o site das empresas	Não	Não	Sim	Não	Não
Tem interface adaptada ao cliente com base em seu perfil no sistema	Não	Não	Não	Não	Não
Provê compras online	Não	Sim	Não/Sim	Sim	Sim
Lida com produtos de diversos ramos comerciais	Sim	Não	Sim	Não	Sim
Fornece perfil de clientes, empresas e produtos	Não	Não	Não	Não	Não
Sugere produtos com base no perfil do cliente	Não	Não	Não	Não	Não

3. SERVIÇO DE INFORMAÇÕES COMERCIAIS SEI-COM

3.1 INTRODUÇÃO

A análise dos diversos serviços apresentados no capítulo anterior serviu para motivar .a concepção do SEI-Com, um novo serviço de informações comerciais que incorpore as principais características dos mesmos para prover consumidores e empresas com um serviço mais genérico e abrangente que lhes possibilite a comodidade de obter informações e serviços, dos mais diversos ramos comerciais, agrupados em um único site.

Dentre as principais características verificadas que podem ser aproveitadas para a definição desse serviço, destacam-se:

- Obter dados a serem utilizados por meio de extrações automáticas via Internet diretamente nos banco de dados das empresas;
- Lidar com produtos de diversos setores do comércio;
- Comparar preços de um produto em diversas empresas que mantém dados no SEI-Com ;
- Disponibilizar compras online.

Além das características citadas acima, ao SEI-Com foram incluídas novas características que visam estender suas funcionalidades:

- Comparar os preços de uma lista de produtos, ao invés de somente um produto por vez, selecionada interativamente pelo cliente;
- Oferecer o orçamento de uma lista de produtos comparando os preços nas diversas empresas;
- Fornecer o perfil de clientes, produtos e de empresas com base no histórico das preferências dos clientes e do comportamento dos mesmos no sistema,

dando subsídios básicos à tomada de decisão às empresas;

- Oferecer interface personalizada a cada cliente, conforme o seu perfil no sistema.
- Oferecer consultas com base em componentes dos produtos, possibilitando uma seleção mais criteriosa dos produto desejados;
- Manutenção dos dados das empresas do banco de dados do SEI-Com.

Estabelecidas algumas características desejáveis à concepção do SEI-Com, vamos agora explicitar os seus principais objetivos.

Apoio à tomada de decisões por parte das empresas

A manutenção do histórico do comportamento de clientes, produtos e empresas para o fornecimento de perfis dos mesmos. Empresas poderão utilizar esses perfis como subsídios básicos à *tomada de decisões* quer seja quanto à presença de seus produtos no mercado ou até mesmo sobre a sua posição frente aos concorrentes.

Estímulo à livre concorrência

Por comparar os preços dos produtos fornecidos por diversas empresas o SEI-Com torna-se uma ferramenta de auxílio à tomada de decisões por parte do consumidor no que diz respeito às empresas nas quais ele deve efetuar suas compras e isso faz com a concorrência entre essas empresas torne-se cada vez mais acirrada.

Apoio ao desenvolvimento do Comércio Eletrônico

Apesar da crescente difusão do comércio eletrônico na Web, ainda existem alguns fatores que contribuem para que o mesmo se não se estabeleça plenamente. Um desses fatores é a falta de conhecimento dos prováveis consumidores, ou seja não se tem ao certo o perfil dos mesmos. O SEI-Com surge como um grande aliado ao desenvolvimento do comércio eletrônico fornecendo perfis de usuários de uma dada região.

3.2 ARQUITETURA DO SEI-COM

A Figura 4.1 mostra a arquitetura do SEI-Com, que será introduzida nas próximas seções.

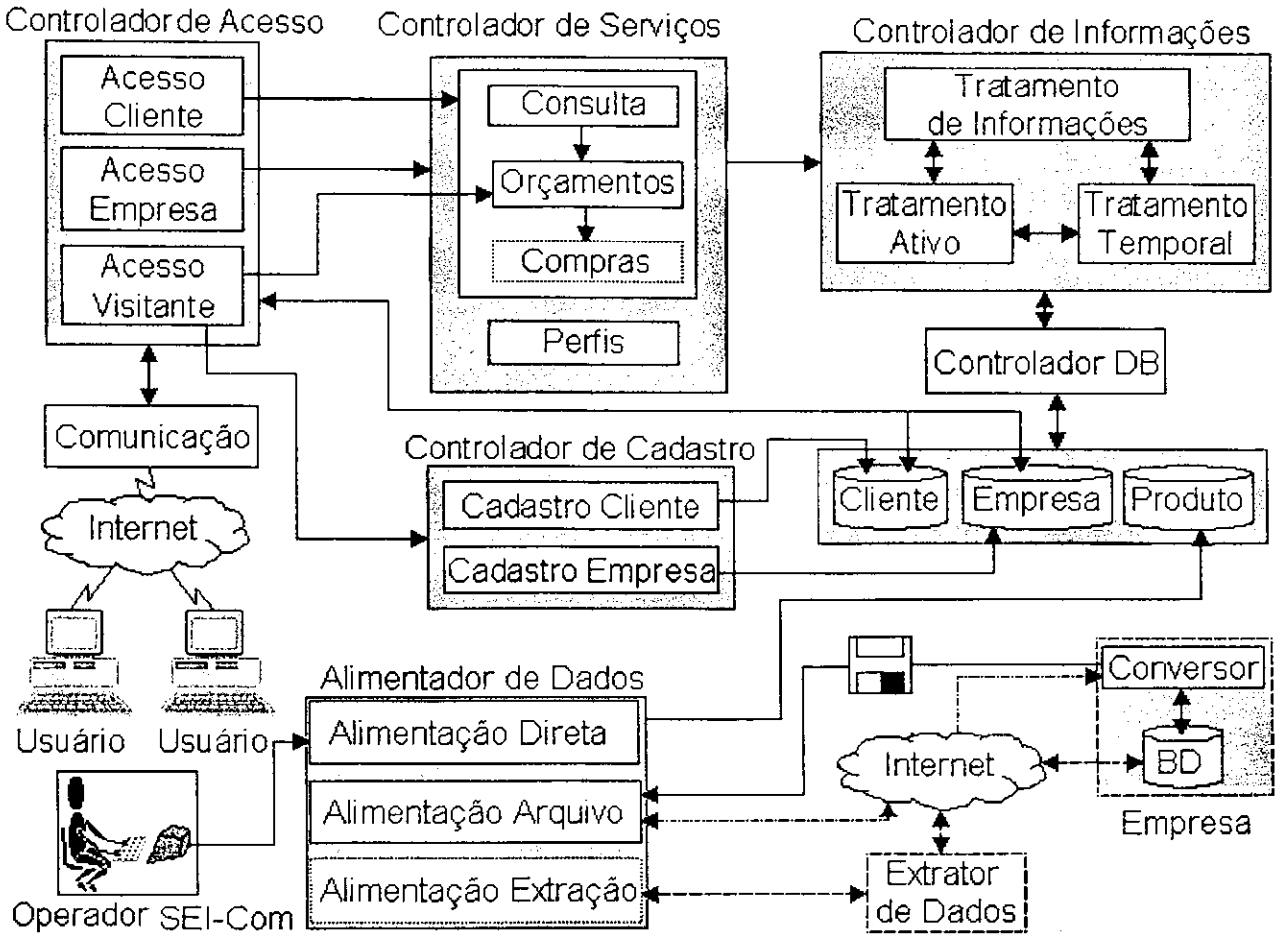

Figura 4.1 – Arquitetura do SEI-Com

Controlador de Acesso

O Controlador de Acesso processa o atendimento dos usuário do sistema, além de fazer o mapeamento de cada tipo de usuário ao(s) serviço(s) a ele disponível. A Tabela 4.1 mostra os tipos de usuários e os serviços disponíveis a cada um deles.

Usuário\Serviço	Consultas	Orçamentos	Compras online	Perfis
Cliente	Sim	Sim	Sim	Não
Empresa	Sim	Sim	Sim	Sim
Visitante	Não	Sim	Não	Não

Tabela 4.1 – Serviços disponíveis aos usuários

3.2.1 Controlador de Cadastro

O Controlador de cadastro tem como função efetuar o cadastro de clientes e empresas no SEI-Com. O cadastro do usuário é feito online e o registro correspondente ao mesmo é sinalizado para ser validado, posteriormente, pelos administradores do SEI-Com.

3.2.2 Controlador de Serviços

Ao controlador de serviços cabe a tarefa de fazer o gerenciamento dos diversos serviços providos pelo SEI-Com. Nas seções a seguir descreveremos cada um dos serviços disponíveis pelo SEI-Com.

3.2.2.1 *Orçamento*

Esse serviço tem como finalidade fornecer ao usuário a melhor opção de compra de sua lista de produtos. Ao ativar o serviço o usuário poderá montar sua lista de produtos interativamente, podendo obter sugestões de produtos com base em seu comportamento no sistema(perfil). Um orçamento poderá ser obtido considerando-se um dos seguintes fatores:

- Preço da lista completa, ou com a maior quantidade de itens satisfazendo a requisição do cliente. Nesse tipo de orçamento é fornecido ao usuário uma lista de preço considerando a combinação de até três estabelecimentos.

- Preço em estabelecimentos específicos – Para obter o orçamento de uma lista de produtos em estabelecimentos específicos, basta selecioná-los no início da montagem da lista, que a partir daí todos os dados disponibilizados serão referentes aos mesmos.

- Preço individual de cada produto – A lista de preços é montada considerando o preço de cada produto, ou seja o sistema informa onde cada produto satisfaz a solicitação do cliente.

3.2.2.2 *Perfil*

Esse serviço visa prover empresas com o perfil(estatísticas) de:

- Clientes – Quando um cliente acessa os serviços do SEI-Com o seu comportamento é registrado em histórico do mesmo. Esse histórico contém informações de suas preferências nos diversos serviços. Além dessas informações são também considerados informações pessoais do cliente para a obtenção do perfil do mesmo. O comportamento do cliente no sistema também pode ser utilizado para provê-lo com uma interface personalizada ou na sugestão de uma lista de produtos, obtida por meio de regras ativas e de uma análise temporal do histórico de seus orçamentos, quando o mesmo estiver utilizando o serviço de orçamento ou compras online.

- Produtos – As informações referentes à procura por um produto em orçamentos e em compras online são registradas em um histórico de produto para dar subsídios às empresas na avaliação da aceitação dos produtos por ela fornecidos.

- Empresa – Sempre que uma empresa atende a um orçamento, satisfazendo um critério escolhido pelo usuário, ou a uma compra online, os dados referente a esse atendimento é registrado no histórico de empresas para que cada uma delas possa avaliar a sua presença no mercado.

3.2.2.3 *Consultas*

O serviço de consultas visa prover clientes com informações de produtos e empresas. Uma das principais consultas implementadas é a que permite que o cliente possa selecionar produtos com base em seus componentes, por exemplo o cliente pode consultar todos os produtos que contenham gordura vegetal, farinha de trigo e ovos, mas não contenham açúcar.

3.2.2.4 *Compras online*

Logo após a obtenção do orçamento da lista de produtos do usuário, lhe é dada a opção de fazer as compras *on-line*, dando ao mesmo a comodidade de fazer suas compras pela Internet, economizando tempo e dinheiro. O pedido do cliente é enviado ao SEI-Com que o repassa às empresas para a efetivação da compra.

3.2.3 Controlador de Informações

Módulo responsável pelo processamento das informações que são utilizadas nos diversos serviços disponíveis.

3.2.3.1 *Tratamento ativo*

O módulo de tratamento ativo do SEI-Com tem por objetivo tratar das informações obtidas por meio de regras ativas no SGBD. Dentre as informações geradas por meio de regras ECA[6] podemos citar:

- Lista Sugerida – Sempre que um usuário solicita um orçamento ou efetua uma compra usando o SEI-Com, a lista de produtos escolhidos é registrada em um histórico e gera automaticamente uma lista de produtos a ser sugerida a ele em compras ou orçamentos futuros.

- Perfil – Cada vez que um usuário acessa o sistema e confirma um orçamento, uma consulta ou uma compra é registrado no histórico do perfil do mesmo todas as suas escolhas e preferências. A inserção de um item no histórico do perfil do usuário ativa uma regra no SGBD que atualiza automaticamente o perfil do mesmo para ser usado futuramente.

---

[6] *As regras ECA(Evento/Condição/Ação) são a base para os banco de dados ativos[ZCF+97,WC95]. Elas fazem com que o banco de dados reaja automaticamente quando da ocorrência de um evento que a dispare. A semântica de execução de uma regra ECA é a seguinte: sempre que seu evento disparador ocorrer e sua condição seja verdadeira a ação correspondente é executada. Um dos principais objetivos de uma regra ativa é reduzir as interferências externas[Sch96]*

3.2.3.2 *Tratamento Temporal*

Orçamentos feitos, compras realizadas e as diversas escolhas do usuário são armazenados em históricos. O tratamento temporal tem por objetivo a obtenção de dados dos históricos de clientes, produtos e empresas para fins de geração de perfis e estatísticas. Um dos principais usos dos dados temporais é a obtenção do comportamento de usuários, produtos e empresas. Esse comportamento é obtido por meio de uma análise temporal e ativa dos dados dos mesmos.

3.2.4 Alimentador de Dados

Esse módulo tem a função de inserir, atualizar e remover dados sobre produtos e fornecimento de produtos no SEI-Com. Todas essas operações são efetuadas localmente por um operador de Interface. São previstos três formas de Alimentação de dados: Alimentação Direta, Alimentação via Arquivo e Alimentação via Extração Automática dos Dados via Internet.

Todos os dados utilizados pelo SEI-Com são armazenados localmente, contrariando à característica distribuída da Internet, isso se dá pelo fato de que as diversas empresas que terão seus dados no SEI-Com ainda não têm sites na Internet disponibilizando os mesmos.

3.2.4.1 *Alimentação Direta*

Nesse tipo de Alimentação o operador de interface faz a entrada dos dados diretamente no teclado inserindo os dados um a um. Esses dados podem ser obtidos por coleta direta nas empresas, ou fornecidos pelas empresas.

3.2.4.2 *Alimentação via Arquivo*

Nesse tipo de alimentação uma empresa faz a conversão de seus dados para um formato previamente estabelecido e os guarda em um arquivo ASCII. Após o arquivo de dados Ter sido carregado o mesmo é enviado ao SEI-Com via Internet ou por

meio de uma mídia eletrônica tal como disquete ou CD.

3.2.4.3 *Alimentação via Extração Automática*

Nessa forma de alimentação o módulo de extração automática de dados é acionado para "minerar " os dados via Internet diretamente do banco de dados da empresa.

4. IMPLEMENTAÇÃO E INTERFACE DO SEI-COM

Neste capítulo descrevemos a implementação do SEI-Com. Na seção 5.1 apresentamos a arquitetura do ambiente de implementação do SEI-Com e na seção 5.2 apresentamos as interfaces do sistema.

4.1 ARQUITETURA DO AMBIENTE DE IMPLEMENTAÇÃO

O SEI-Com foi implementado segundo uma arquitetura de três camadas como vemos na Figura 5.1. Essa arquitetura se enquadra na arquitetura de Cliente Web Estendido conforme visto na seção 2.5.1.3.2.

A primeira camada é representada pelo cliente que é um browser Web capaz de reconhecer a linguagem Java. O cliente pode residir em qualquer plataforma incluindo UNIX, Windows e MacOS.

As classes Java do SEI-Com residem no servidor Web, mas quando usuário acessa a página HTML contendo o applet SEI-Com todas as classes necessárias ao seu funcionamento são transferidas para o cliente, a partir daí toda a lógica da aplicação é executada na máquina do usuário, diminuindo a carga no servidor.

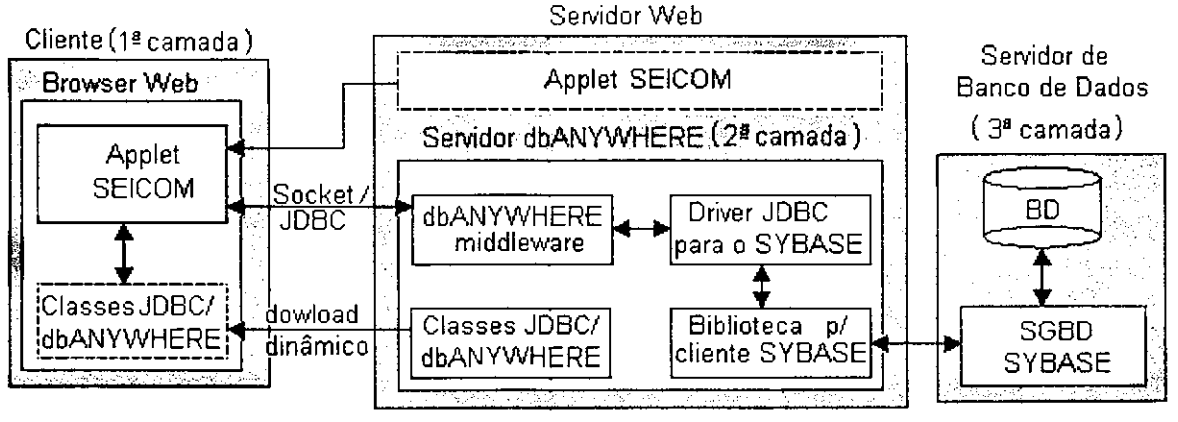

Figura 5.1 – Arquitetura do ambiente de Implementação

A Segunda camada é constituída pelo servidor dbANYWHERE, um servidor de acesso a dados. Os componentes nessa camada fazem o gerenciamento de toda comunicação com o banco de dados. O servidor pode residir em plataforma Windows 95 ou Windows NT

A configuração do servidor dbANYWHERE tem os seguintes componentes:

- Software do servidor(dbANYWHERE middleware) – faz o gerenciamento das conexões entre o cliente e o servidor de banco de dados.

- Driver do banco de dados – Usado para estabelecer a comunicação entre o dbANYWHERE e o servidor de banco de dados.

- Software cliente para o banco de dados – Além das funcionalidades providas nos componentes anteriores, o servidor dbANYWHERE é um cliente do servidor de banco de dados. A maioria dos banco de dados são suportados pelo dbANYWHERE, dentre eles citamos: ORACLE, SYBASE, Microsoft SQL Server e Informix.

A implementação do SEI-Com foi feita utilizando a linguagem Java, um servidor de acesso a dados chamado dbANYWHERE e o banco de dados SYBASE.

Quando o cliente acessa a página HTML, na qual o applet do SEI-Com se encontra, todas as classes do sistema são descarregadas em sua máquina em um único arquivo chamado seicom.jar. Os arquivos com extensão .jar podem ser compactados diminuindo assim a carga na transmissão dos dados e são descarregados na máquina do cliente de uma única vez eliminando dessa forma o custo adicional de se fazer uma conexão para cada classe utilizada.

4.2 DESCRIÇÃO DAS INTERFACES DO SISTEMA

Essa seção apresenta as principais telas do SEI-Com. Toda a Interface do sistema foi feita tendo em vista torná-la atrativa e fácil de usar[Hab91]. A manutenção de todas as escolhas dos usuários em seu histórico, torna possível que lhes

ofereçamos uma interface personalizada em todos os serviços disponíveis. Isso faz com que a navegação no sistema torne-se muito mais fácil e rápida.

As seções a seguir apresentarão as diversas interfaces do SEI-Com.

4.2.1 Interfaces de Acesso ao SEI-Com

Existem duas formas de acessar o SEI-Com. A primeira é acessar por meio da página do projeto SEI, cujo endereço é http://lsi.dsc.ufpb.br/SEI/sei.html, e a partir daí acessar o SEI-Com clicando no ícone do mesmo. A Figura 5.2 mostra a página principal do projeto SEI.

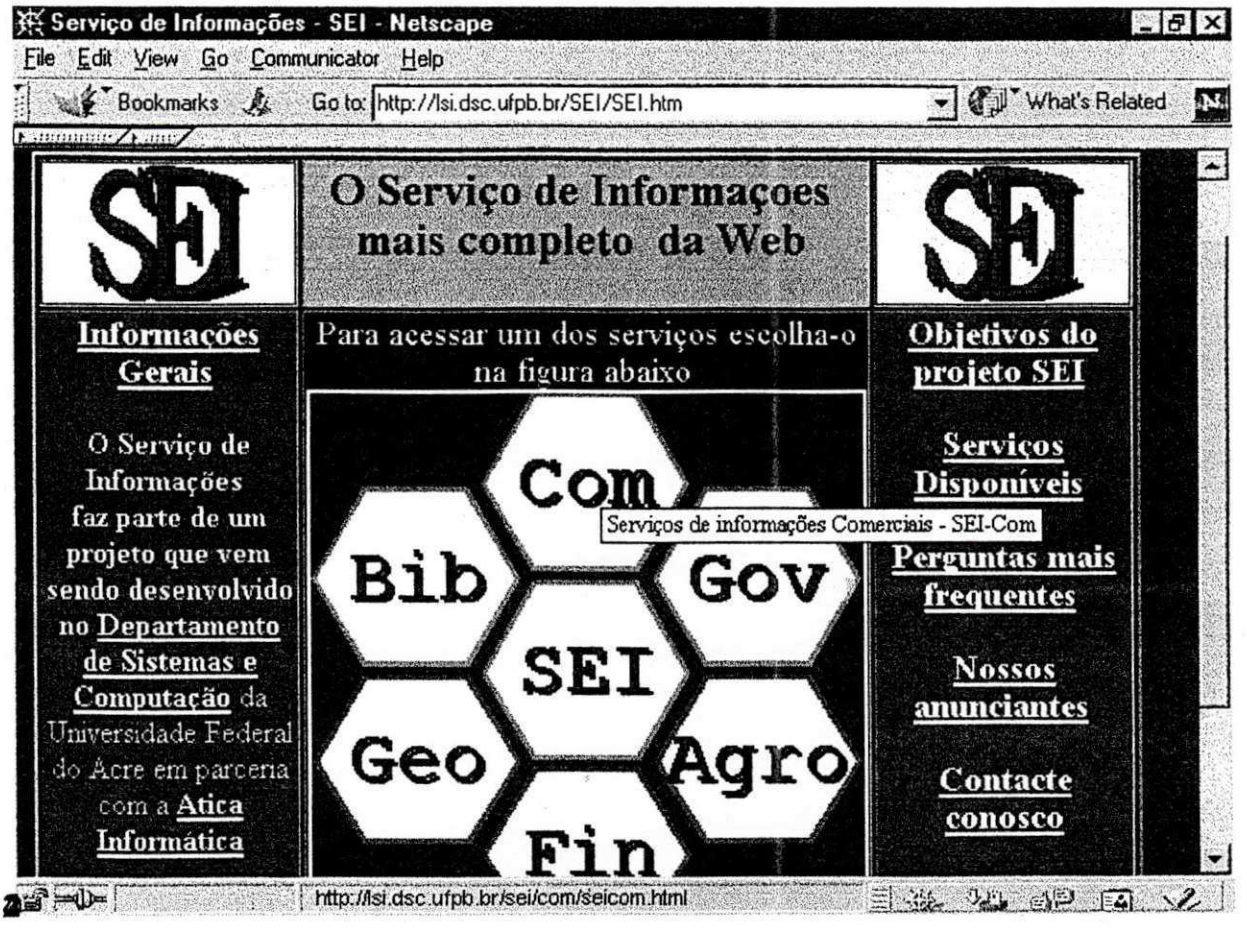

Figura 5.2 – Tela principal do Projeto SEI.

A Segunda forma de acesso ao SEI-Com é acioná-lo diretamente no endereço http:\\lsi.dsc.ufpb.br\sei\com\seicom.html. A Figura 5.3 mostra a tela principal do SEI-Com.

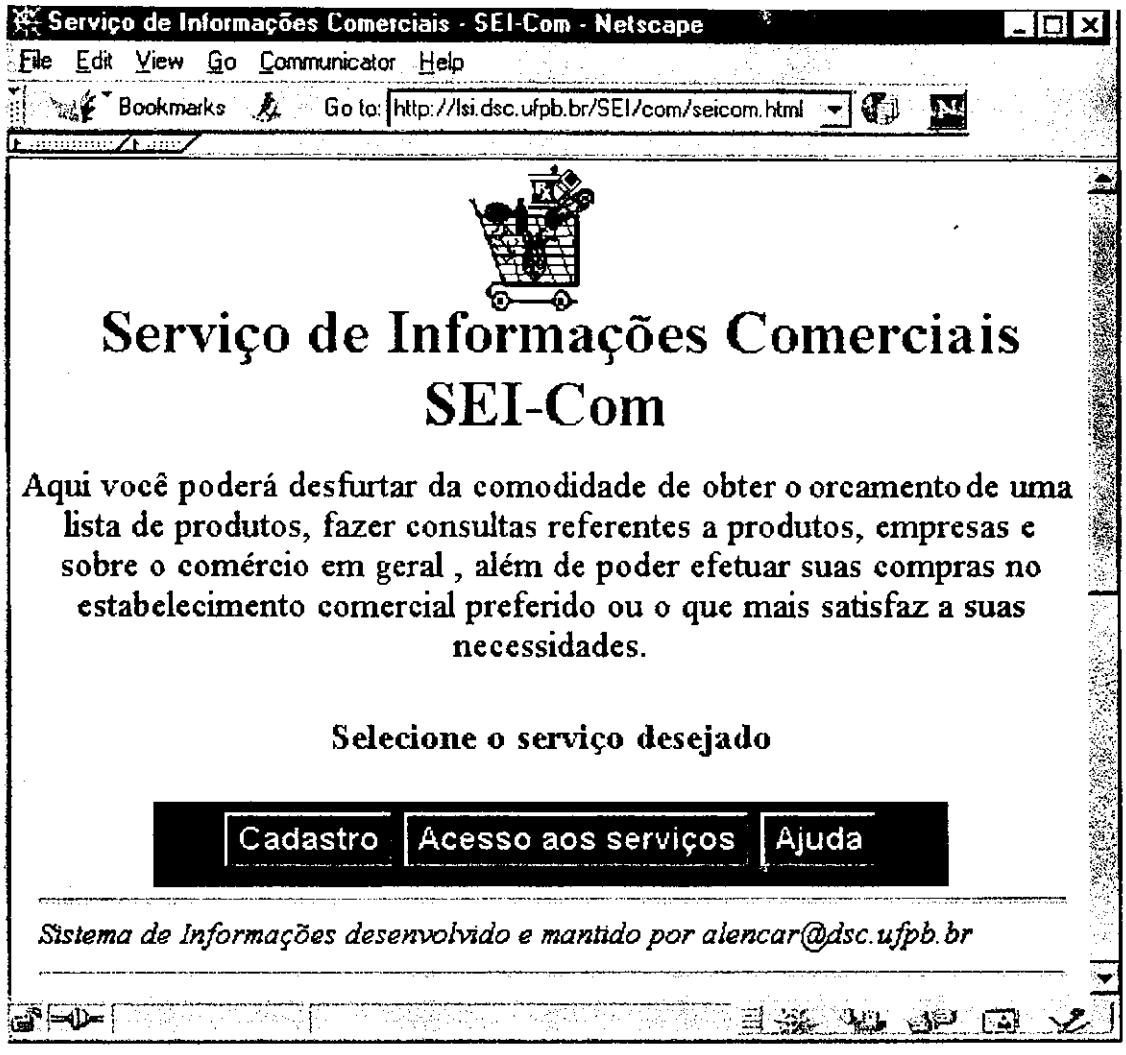

Figura 5.3 – Tela de principal do SEI-Com

4.2.3 Interfaces do Controlador de Cadastro

Caso o usuário seja um visitante pode cadastrar-se como cliente, conforme Figura 5.4, ou como empresa, veja Figura 5.5. O cadastro fica pendente até que o administrador do sistema o valide.

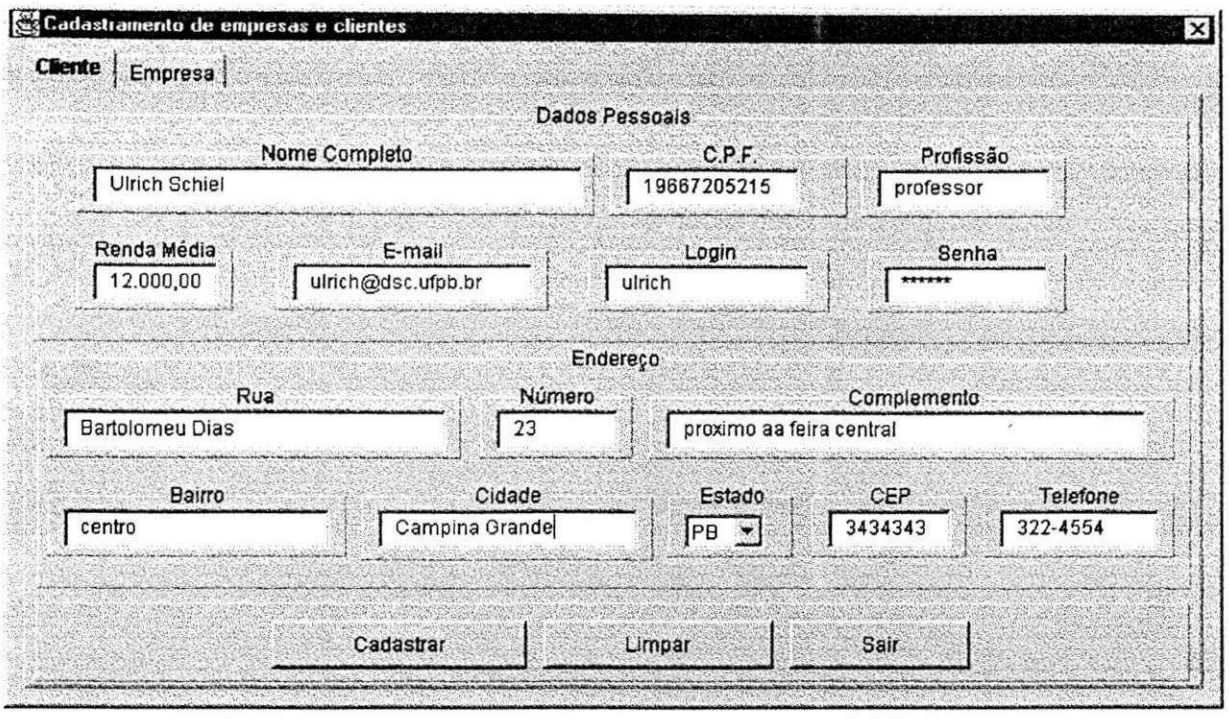

Figura 5.4 – Interface do Cadastro de Clientes

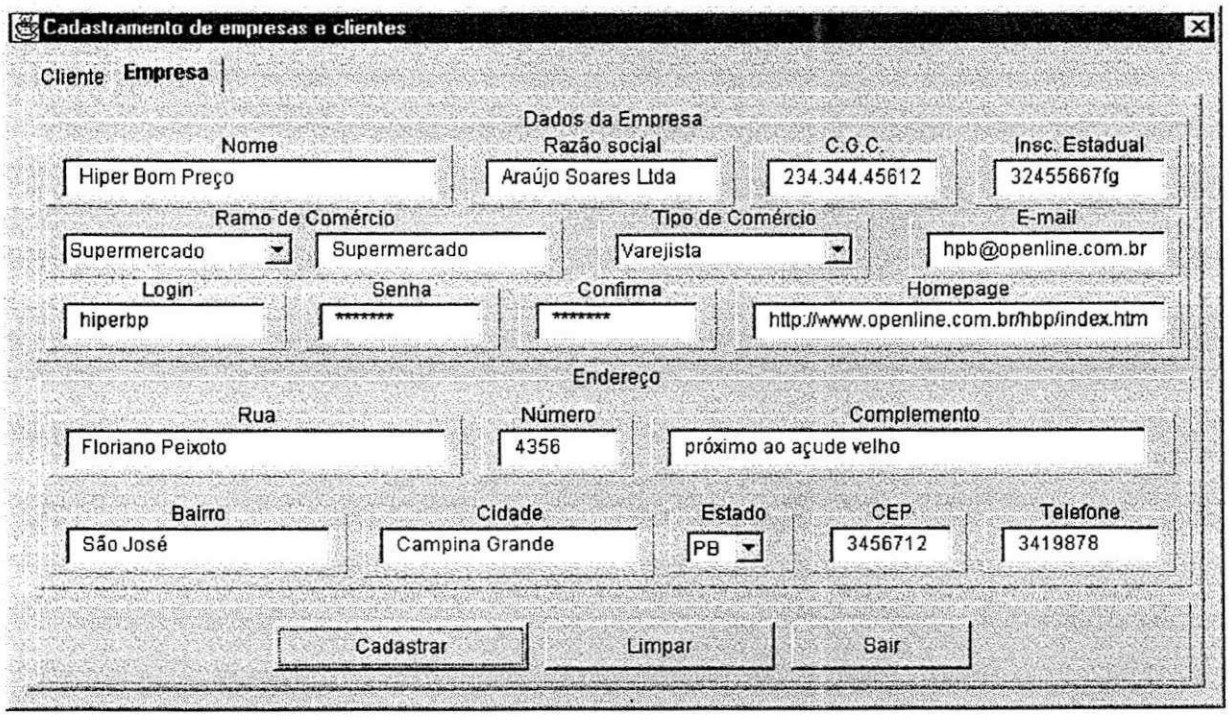

Figura 5.5 – Interface do Cadastro de Empresas

4.2.4 Interface do Controlador de Acesso

Quando a opção Serviço é escolhida pelo usuário, a tela do Controlador de Acesso

lhe é apresentada para fins de autenticação, conforme Figura 5.4. O Controlador de Acesso considera três tipos de usuários:

- Cliente – Usuário que tem acesso aos serviços de Orçamentos, Consultas e compras online.

- Empresa – Uma empresa que mantêm os dados no SEI-Com e acessa o SEI-Com para acompanhar o *comportamento* de seus produtos no mercado e para avaliar a sua presença frente à concorrência. Uma empresa tem acesso irrestrito aos serviços.

- Visitante – Um tipo de usuário especial para o SEI-Com pois o mesmo poderá vir a ser um cliente cadastrado. O usuário visitante pode somente usufruir do serviço de orçamento.

Além do seu tipo, o usuário deve informar seu login e sua senha, dispensados para o usuário visitante, para Ter acesso aos serviços.

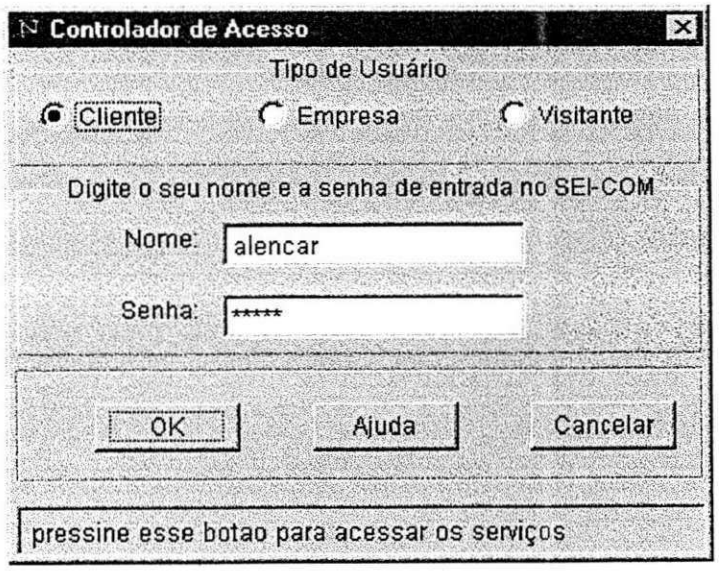

Figura 5.6 – Interface do controlador de Acesso aos serviços

4.2.5 Interface do Controlador de Serviços

Após a confirmação dos dados no Controlador de Acesso é apresentado ao usuário,

caso o mesmo esteja apto a acessar o SEI-Com, o Controlador de Serviços que exibe os serviços disponíveis. Para ver os serviços disponíveis para cada tipo de usuário vide tabela 4.1.

A Figura 5.5 mostra a interface do Controlador de Serviços. A escolha dos serviços de qualquer um dos serviços faz com que uma tela de escolha do ramo comercial desejado seja mostrada conforme a Figura 5.8.

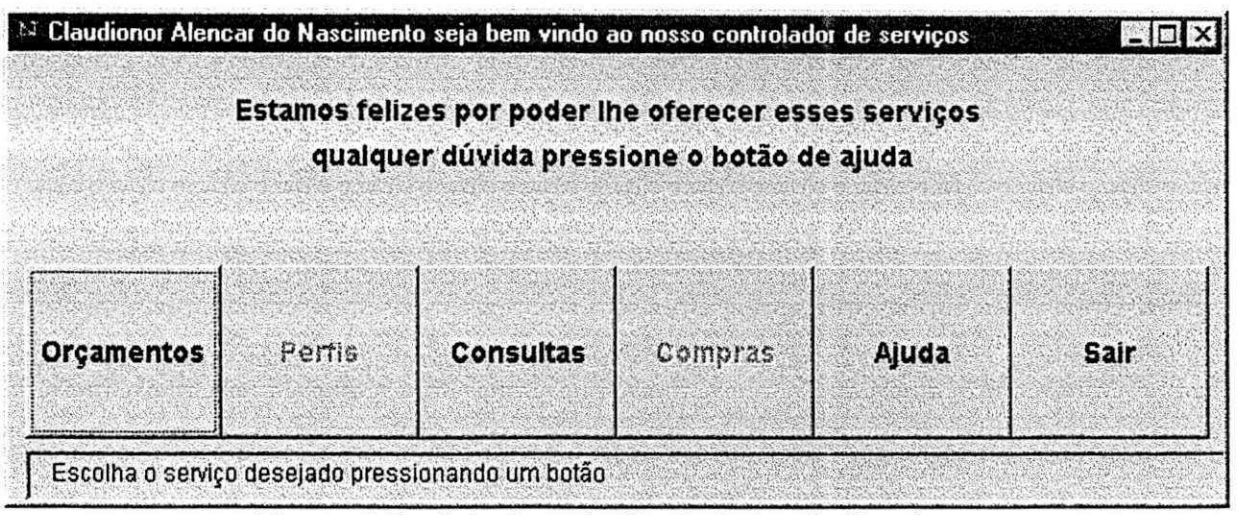

Figura 5.7 – Interface do Controlador de Serviços

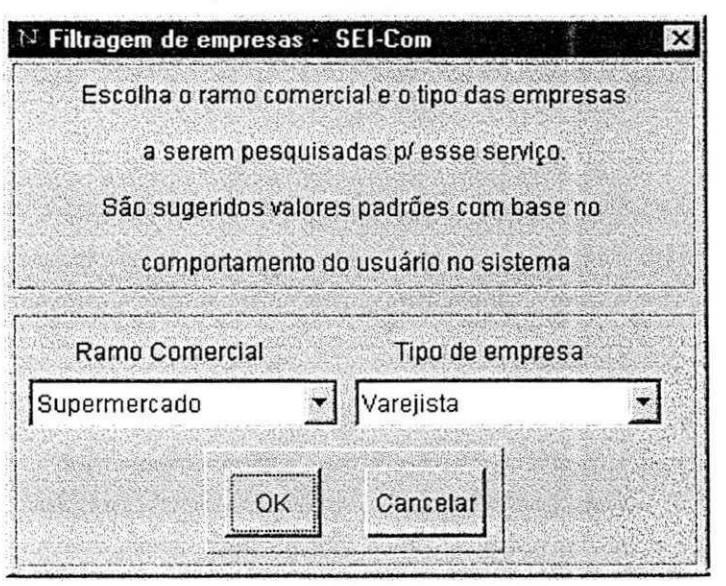

Figura 5.8 – Tela de escolha do ramo comercial e tipo de empresa

Interfaces de Orçamentos

Para obter o orçamento de uma lista de produtos o usuário deve montar

interativamente a lista de produto desejada, ver Figura 5.9, podendo o mesmo selecionar os estabelecimentos nos quais ele deseja obter o orçamento, conforme Figura 5.10. Na montagem de sua lista, o usuário pode solicitar sugestões de uma lista de produtos com base em seu *comportamento* no sistema, além disso ele pode solicitar produtos que estejam em promoção.

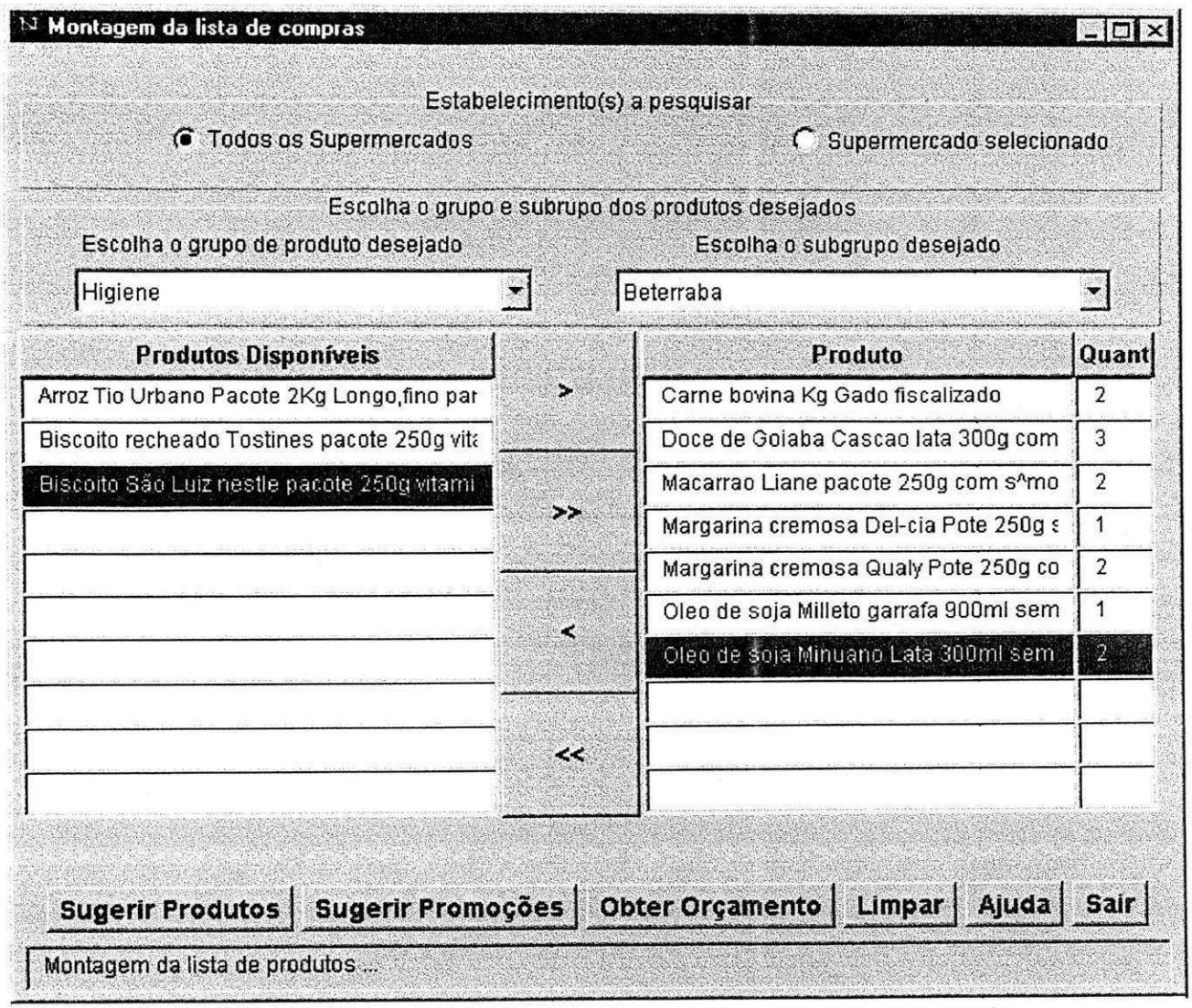

Figura 5.9 – Montagem da lista de produtos para obter o orçamento

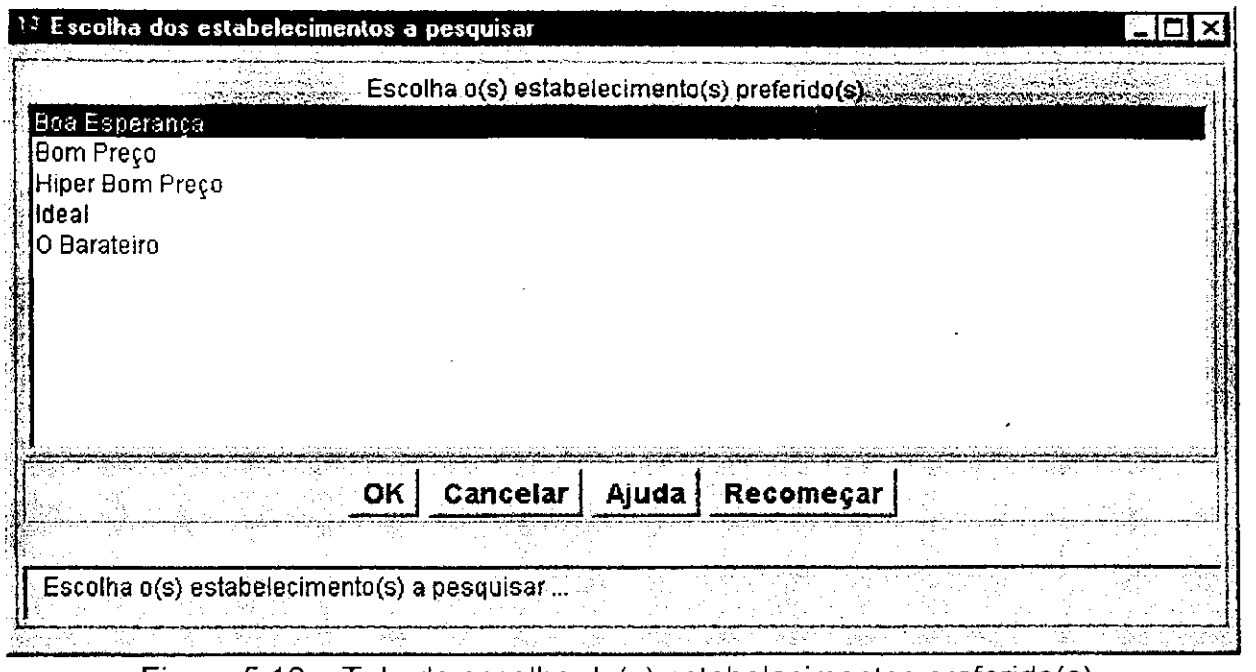

Figura 5.10 – Tela de escolha do(s) estabelecimentos preferido(s)

Interface do Alimentador de Dados

O Alimentador de dados(AD) é um dos principais módulos do SEI-Com que serve para fazer a manutenção de produtos e fornecimento de produtos. O AD é uma aplicação Java deve ser instalado em uma rede local na qual o servidor de banco de dados se encontra. Sua interface é composta de três partes mostradas nas figuras 5.11, 5.12, 5.13.

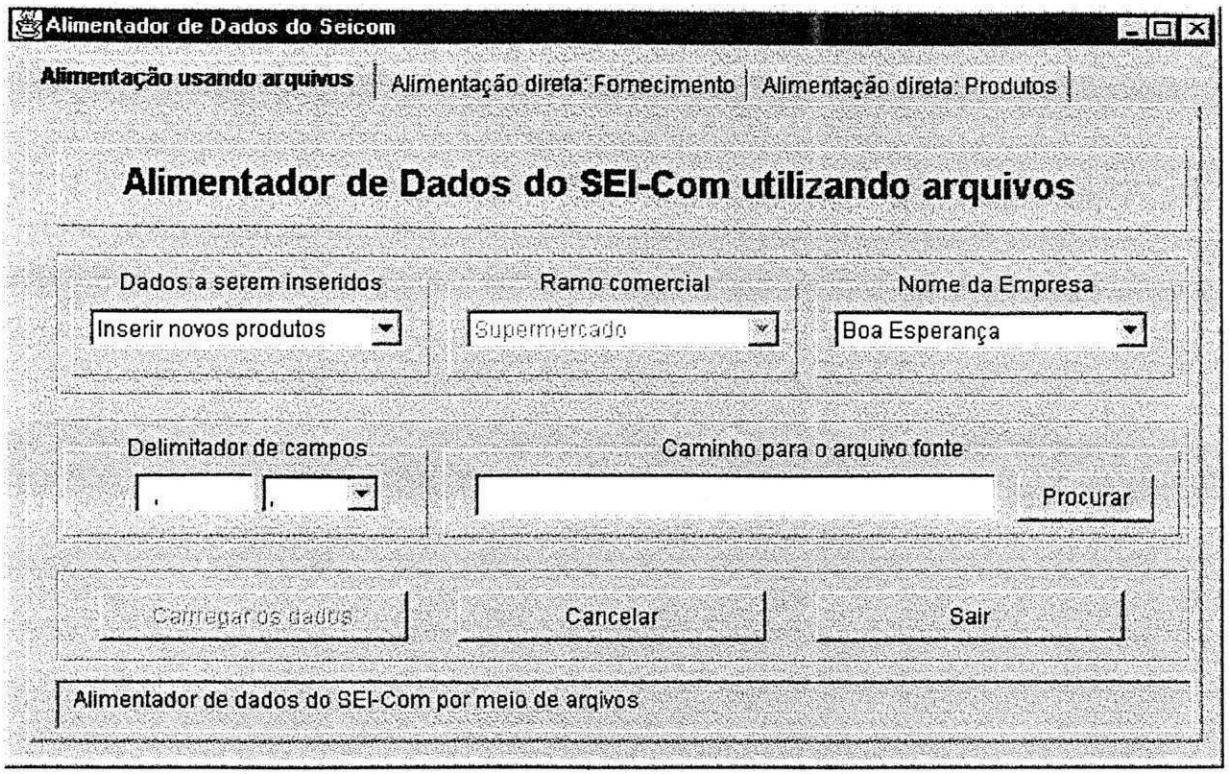

Figura 5.11 – Interface do Alimentador de Dados via Arquivo

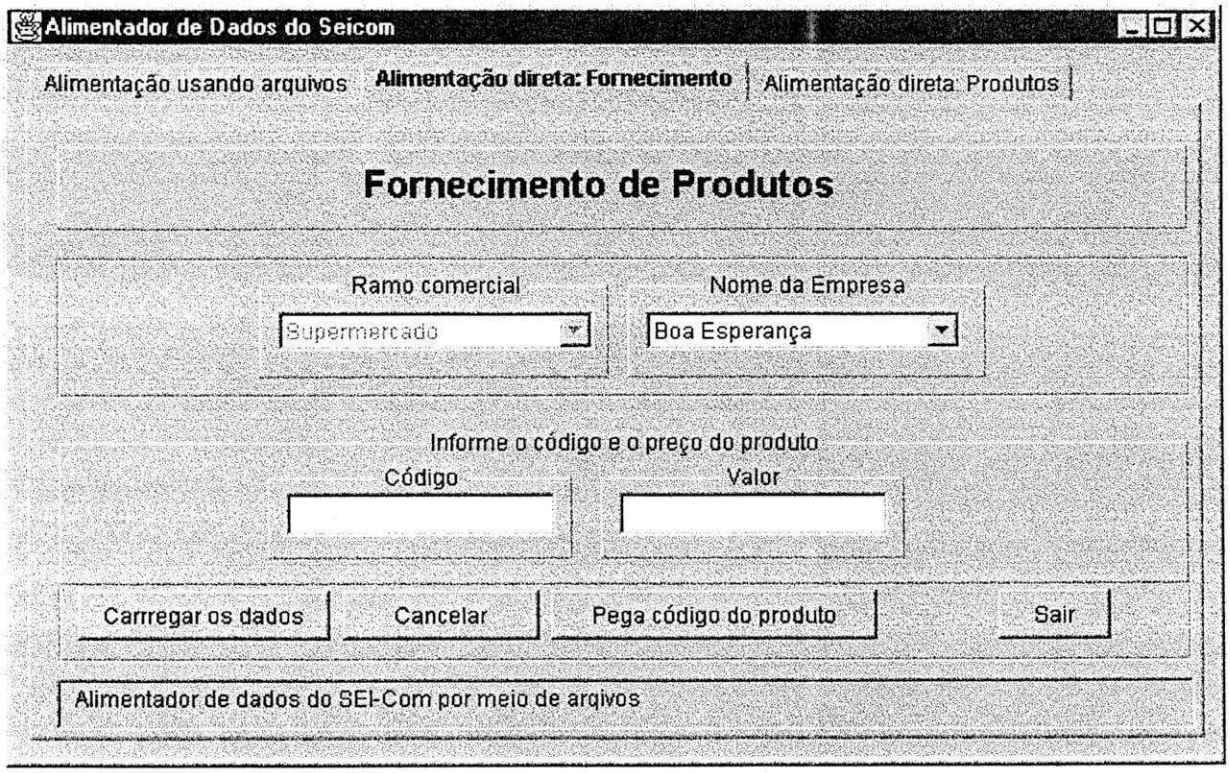

Figura 5.12 – Alimentação Direta de Fornecimento

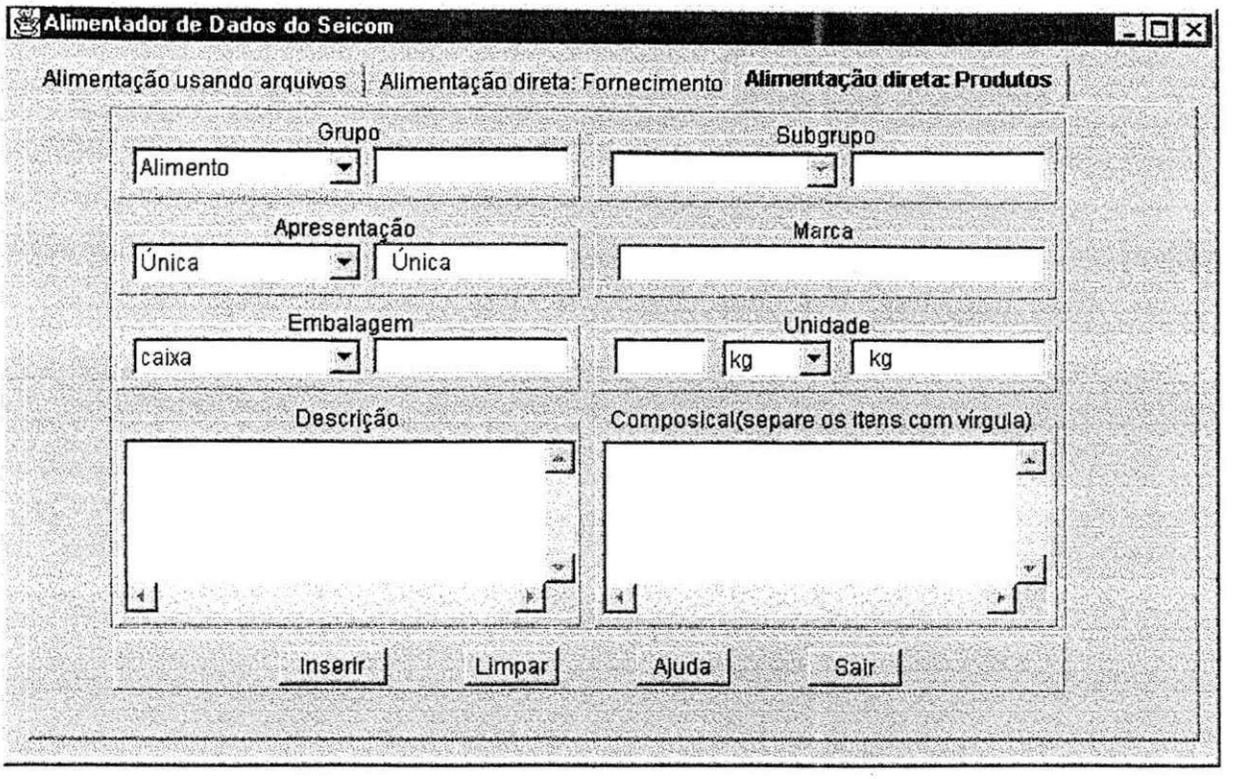

Figura 5.13 – Alimentação Direta de Produtos

4.3 ESTRUTURA GENÉRICA DOS DADOS

Uma das características do SEI-Com é a capacidade de agrupar dados dos mais diversos ramos comercias e disponibilizá-los aos usuários nos diversos serviços a eles disponíveis.

Para termos uma interface que suporte os diversos ramos comerciais sem a necessidade de mudanças no código fonte e nem na estrutura das tabelas é que estabelecemos um formato a ser usado nas tabelas que contêm os dados dos produtos para cada ramo comercial. Utilizaremos a tabela produtos(Figura 6.1), utilizada para armazenar os produtos de supermercados, para mostrar como os diversos ramos são suportados.

Os atributos que têm o símbolo (*) ao seu lado são obrigatórios para as tabelas de produtos de qualquer ramo comercial e devem aparecer nas tabelas seguindo a ordem em que aparece na Figura 6.1. NA Tabela 6.1 são descritos os atributos

obrigatórios às tabelas de produtos dos todos os ramos comerciais.

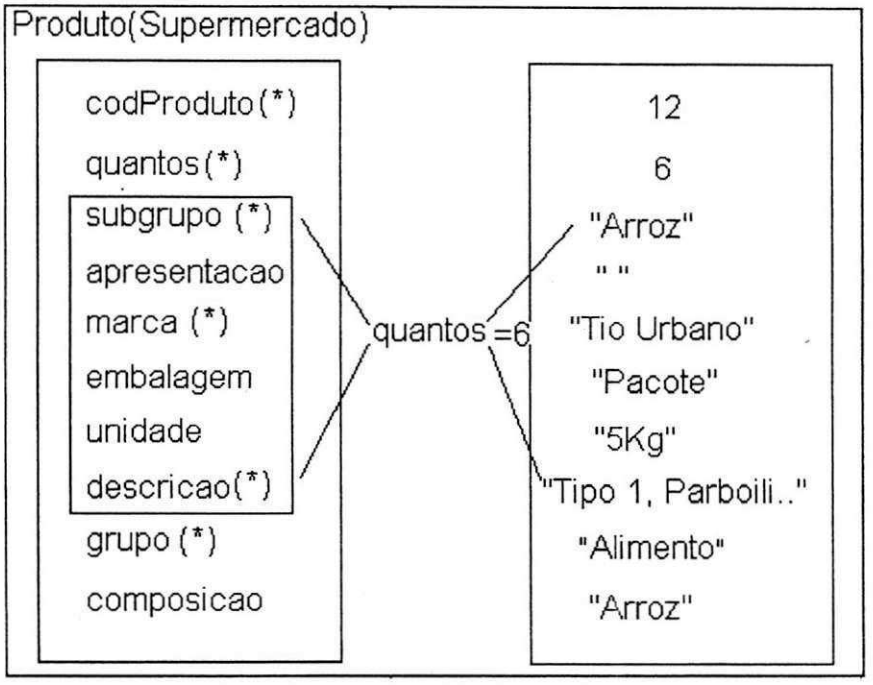

Figura 6.1 – Tabela de produtos de supermercados e uma instancia

Atributo	Descrição
codProduto	Código do produto
quantos	Especifica quantos atributos após o segundo serão concatenados para serem exibidos nos serviços. Exemplo: Para concatenar uma tupla da tabela de produtos conforme Figura 6.1 basta considerar os seis(6) atributos após o segundo atributo quantos. Assim o resultado seria: "Arroz Tio Urbano pacote 5Kg, tipo 1, Parboilizado"
subgrupo	Especifica o subgrupo ao qual o produto pertence. Exemplo: No Exemplo acima temos que o produto pertence ao grupo "Alimento" e ao subgrupo "Arroz".
marca	Marca do produto.

descrição	Descrição sucinta do produto.
grupo	Grupo(Categoria) ao qual o produto pertence. Exemplo: No Exemplo acima temos que o produto pertence ao grupo "Alimento"

Tabela 6.1 – Atributos obrigatórios às tabelas de produtos de ramos comerciais.

O relacionamento entre um ramo comercial e a tabela de produtos correspondentes é feita quando o usuário seleciona um ramo comercial conforme Figura 5.8 e então o programa busca na tabela "InfoRamo", descrita no apêndice, a tabela de produtos correspondente.

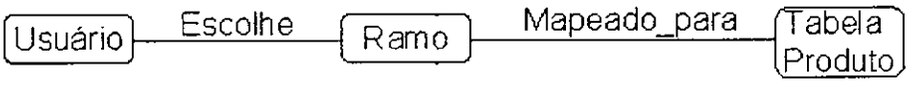

Figura 6.2 – Mapeamento de um Ramo comercial à tabela de produtos correspondente.

A Figura 6.3 mostra trechos de código que mostra como o mapeamento de um ramo comercial à sua tabela de produtos é feito.

```
public void PegaRamo(){
        // Faz uma consulta para pegar os atributos de infoRamo que
        // são: RamoComercial, tabelaReferencia, quantos
        C_Acesso.CDB.Consulta("Select * from InfoRamo order by Ramo");
        // O resultado da consulta é guardado na Matriz Ramo
        Ramo = C_Acesso.CDB.dataBase.Dados;
        quantos = C_Acesso.CDB.quantos;
        // guarda o nome da tabela que contém os produtos do ramo comercial
        // escolhido.
        tabela = Ramo[0][1];
}
```

Figura 6.3 – Código fonte do mapeamento de um Ramo Comercial à tabela correspondente

5. CONCLUSÕES

Neste capítulo apresentamos as conclusões deste trabalho. Analisamos o Serviço de Informações Comerciais(SEI-Com) , principal objetivo desta dissertação, mostrando suas contribuições e trabalhos futuros gerados por ele.

5.1 CONSIDERAÇÕES FINAIS

Neste trabalho apresentamos o Serviço de Informações Comerciais na Web com características Ativas(SEI-Com). O SEI-Com foi concebido para prover o consumidor com serviços comerciais que lhe possibilite obter, por meio da Internet, orçamentos de lista de produtos, efetuar compras online de modo a minimizar tempo e dinheiro. Às empresas o SEI-Com possibilitará o acompanhamento da sua presença no mercado por meio do serviço de perfis tanto dos produtos fornecidos, quanto de suas posição frente à concorrência.

Os perfis de clientes, produtos e empresas são obtidos por meio do uso de regras ECA. Essas regras analisam o histórico de cada um desses itens para traçar o comportamento no sistema.

As principais contribuições desta dissertação são:

- A criação de uma ferramenta de apoio à tomada de decisões por parte das empresas, haja vista que por meio do SEI-Com elas poderão acompanhar a presença no mercado.
- O estímulo à livre concorrência entre as empresas pois cada vez que usuário efetua uma compra online ou um orçamento os preços de cada produto são confrontados nas diversas empresas.
- A criação de uma ferramenta de apoio ao desenvolvimento do comércio eletrônico, pois o SEI-Com disponibiliza o perfil de seus usuários.

Dentre as dificuldades encontradas nesse trabalho podemos citar as seguintes:

- A falta de componentes na linguagem Java adequados à implementação do SEI-Com, fez com que eu tivesse que desenvolver meus próprios componentes . O principal que tive que implementar foi:

  ☞ Grade – Classe utilizada para dispor elementos em forma tabular de modo que se possa fazer inserção, remoção e edição de dados. Quando os número de elementos for maior que o número de linhas visíveis na tela, a mesma pode ser rolada.

  ☞ BrowserPanel – Classe para criar panels personalizados. Com essa classe pode-se criar panels com etiquetas, panels com relevo como botões, panel com depressões, dentre outros.

- A falta de uma metodologia de desenvolvimento de aplicações voltadas à Internet, haja vista que as metodologias existentes não foram projetadas para tratar, especificamente com esse ambiente considerando suas peculiaridades.

5.2 TRABALHOS FUTUROS

A implementação do SEI-Com utilizando a linguagem Java fez com que tivéssemos um controle muito grande sobre as interações com o usuário, no entanto , dadas às limitações da Internet, o desempenho do sistema ficou bastante comprometido já que cada vez que o usuário acessa o sistema todas as classes do SEI-Com têm que ser descarregadas em sua máquina. O que pode ser feito é implementar uma nova versão do SEI-Com utilizando, por exemplo a Interface CGI.

Dado ao fato do protótipo desenvolvido não ter sido totalmente implementado, uma sugestão é concluir o módulo de perfis e o extrator de dados.

REFERÊNCIAS BIBLIOGRÁFICAS

[AJS+97] Asokan, N.; Jason, P. ; Steiner, M.; Waidner, M. *The State of the Art in Electronic Payments Systems.* Computer , IEE Computer Society, Vol 30, N°9 , setembro 1997.

[Ame97] America Online. *AOL Server – Administrator's Guide.* 1997. http://www.aolserver.com/server/docs/2.1/html/admin.htm.

[BCL+94] Berners, L; Connoly, D.; Loutenen, A.; Nielsen, H.; Secret, S. *The World Wide Web, Communications of the ACM, vol 37(8), 1994.*

[Bhi96] Bhimani, A. *Securing the Commercial Internet.* Communicator of the ACM , Vol. 39, N 6 , 1996.

[BN97] Bernstein, P.; Neewcomer, E. *Principles of Transaction Processing For the Systems Professional*, Morgan Kaufman, 1997.

[BM97] Beveridge, T.; McGlashan, P. *High Performance ISAPI/NSAPI Programming.* Coriolis Group Books, 1997.

[BMM94] Lee, B. ; Masinter, L.; McCaihiel, M. *Uniform Resourse Locator(URL).* RFC 1738., University of Minnesota, 1994.

[Cre96] Crede, A. *Eletronic commerce and the banking industry: The requirement and opportunities for new payment systems using internet.* JCMC, number 1, vol 3, December 1996. http://www.usc.edu/dept/annenberg/vol1/issue3/crede.html .

[Den96] Denny, B. *SSI, CGI, API or SSS, Chosing the Right Tool for the job.* 1996. http://solo.dc3.com/white/extending.html

[Enr96] Enrlich, S. *Building Database Aplications for the Web.* ORACLE Coorporation, July 1996.

[Fro95] Frog, J. *Mapping Extended Entity Relationship Model to Object Modeling Techinique.* SIGMOD Record, Vol. 24, N ° 3, September 1995, 18-22.

[Gen97] Genusa, S. *Using ISAPI.* QUE Corporation, 1997.

[Hab91] Habermann, F. *Giving a Real Meaning to 'easy-to-use' Interfaces. IEEE Software, July 1991, pp 90-91.*

[HN96] Hoffman, D., Novak, T. P. *Commercial scenarios for the web: Opportinites and challenges.* JCMC, december, 96. Http://www.usc/dept/annenberg/vlo1/issue3/hoffman.html.

[HN94] Hoffman D. Novak, T. *Wanted: Net census. Wired*, (2.11):93-94, November ,1994. http://www2000.ogsm.vanderbilt.edu/wanted.net.census.html .

[HN96a] Hoffman, D.; Novak, T. . *A new marketing paradigm for eletronic commerce.* The Information Society, Special Inssue on Eletronic Commerce, February 1996. http://www.2000.ogsm.vanderbilt.edu/wanted.net.census.html .

[Int96] Internet Systems. *Internet tools product chat.* DBMS online Magazine, 1996. http://www.dbmsmag.com.9605i04.html.

[Key97] Keyes, J. *Datacasting, how to stream databases over the Internet.* McGraw-Hill USA, 1997.

[Kim96] Kim, C. *A Taxonomy on the Arquitecture of Database Gateways for the Web.* http://grigg.chungnam.ac.kr/projects/UniWeb/Documents/text.html.

[LCL+94] Lee, B.; Calilau, R. ; Loutonen, A.; FrystykNielsen, H. *The world wide web*. Comunications of the ACM, 37 (8), August 1994.

[LCF+96] Lee, B.; Cailliau R.; Luotonen A.; FrystykNielsen H. ; Rolt, T. Benjamin. Eletronic Commerce: Effects on Eletronic Markets. JCMC, 1(3), December 1996. http://www.usc.edu/dept/annenberg/vol1/Issue3/wingand.html .

[Lim96] Lima, I. *Ambiente Web Banco de Dados: Funcionalidades e Arquiteturas de Integração.* Dissertação de Mestrado , DI , PUCRJ, 1997.

[MD89] McCarthy, D.; Dayal, U. *The Architecture Of na Active Data Base management System* .SIGMOD RECORD 1989 pp. 215 –

224.

[Mic97] Microsoft Corporation. *Internet Server API Reference.* 1997. http://microsoft.com/win32dev/apiext/isapiref.html

[NCSA97] National Computer Center for Supercomputing Aplications . *Server Side Includes(SSI).* University of Illinois 1997. http://hoohoo.ncsa.uiuc.edu/docs/tutorial/includes.html.

[Net97] Netscape Communications Corporation. *The NSAPI – Netscape Server API.* http://www.netscape.com/newsref/std/server_api.html.

[Nor96] North, K. *Understanding ODBC 3.0 and OLE DB.* DBMS Online Magazine, April,1996. http://www.dbmsmag.com/9604d53.html.

[NTI97] Eletronic commerce. NTIA – Office of Assistant Secretary, 1997. http://www.ntia.doc.gov:80/opadhome/ecom3.html.

[Man96] Manual de Referência OpenBase , OpusWeb. Ed. Rio de janeiro, 1996

[MSA97] MAS-INFOR Sistemas de Automação Ltda. *CorpWeb Server – Um servidor Web para o ambiente Unisys.* 1997. http://www.msainfor.com.br/icorpweb.htm.

[Pow97] PowerSoft Corporation. *Building Internet Applications with Power Buider 5.0.* 1997. http://www.powersoft.com/products/devtools/pb50/bld_ipb.html.

[Rah96] Rahmel, D. *Comparing Javascript and VBScript.* Internet System, 1996. Http://www.dbmsmag.com/9610i07.html.

[Rei96] Reichard, K. *Web Server for database applications.* Internet Systems, April , 1996. http://www.dbmsmag.com/9610i08.html.

[Shn83] Shneiderman, B. *Direct Manipulation: A Step Beyond Programming Languages.* Computer, August 1983, Vol. 16, No. 8, pp. 57-69.

[Shn87] Shneiderman, B. *Designing the User Interface: Strategies for Effective Human-Computer Interaction.* Addison-Wesley Publishing Company, Inc., 1987.

[Sch96] Schiel, U. *Aspectos Temporais em Sistemas de Informação.*

Relatório técnico DSC/01/96, 1996.

[SN97] Sampaio, M.; Nascimento, C. *Rule Support in na Object-Oriented Data Manager for Cooperative Design.* XII SBBD, 1997.

[Ste96] Stevens, W. *TCP/IP Illustrated, the Protocols*, vol.1. Addison Wesley, 1996.

[SKP96] Steinfield, C.; Kraut, R.; Plummer, A.. The impact of interorganizacional networks on buyer-seller relationships. JCMC, 1(3), December 1996. http://www.usc.edu/dept/annenberg/vol1/issue3/steinfld.html.

[Sun98] Sun Microsystems. *The Virtual Machine Specification,* 1998. http://java.sun.com/doc/vmspec/vmspec_1.html

[Ter98] Terisa Systems *Secure web payments. A complete SET development enviroment from Terisa Systems*, 1997. http://www.terisa.com/products/swp/index.html .

[ZCF+97] Zaniolo, C.; Ceri, S.; Faloutsos, C.; Snodgrass, R.; Subrahmanian, V.; Zicari, R., *Advanced Database Systems* . Morgan Kaufman Publishers, Inc. – San Francisco – CA, 1997.

[WB96] Wingand, R.; Benjamin R.. Eletronic Commerce: Effects on Eletronic Markets. JCMC, 1(3), December 1996. http://www.usc.edu/dept/annenberg/vol1/issue3/wingand.html.

[WC95] Widom, J.; Ceri, S. *Active database systems.* Morgan Kaufmann, 1995.

[Win97] Winslertt, M. *Database and the Word Wide Web Tutorial.* SIGMOD Record, number 26 vol. 3 march 1997.

[W3C97b] W3C Consortium. *Mobile Code*, 1997. http://www.w3.org/pub/WWW/MobileCode.

[W3C97d] W3C Consortium. *The HiperText Transfer Protocol*, 1997. http://www.w3.org/hypertext/WWW/Protocols/Overview.html.

[W3C97e] W3C Consortium. *The HiperText Markup Language*, 1997.

http://www.w3.org/hypertext/WWW/Markup/Markup.html.

[WebQ97] WebQuest Support and Information Center. *SSI+2.0 Reference*, 1997. http://webquest.questar.com/reference/ssi/ssi.html.

APÊNDICE A – RELAÇÕES, TRIGGERS E ÍNDICES DO BD DO SEI-COM

Nesse apêndice apresentamos a definição de tabelas, triggers e índices que compõe o banco de dados do SEICOM. O SGBD utilizado como servidor de banco de dados foi o SYBASE SQL-Anywhere . O dialeto suportado pelo SYBASE SQL-Anywhere e utilizado para a definição de tabelas, triggers e índices foi o Watcom-SQL.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  Criação de Tabelas
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
CREATE TABLE "alencar"."empresa"
(       "codEmpresa"        integer NOT NULL UNIQUE,
        "nome"              char(30) NOT NULL,
        "razaosocial"       char(40) NULL,
        "cgc"               char(15) NOT NULL,
        "insc_est"          char(20) NULL,
        "cep"               char(10) NULL,
        "complemento"       char(30) NULL,
        "ramocomercial"     char(20) NULL DEFAULT 'Supermercado',
        "tipo"              char(15) NULL,
        "login"             char(8) NOT NULL UNIQUE,
        "senha"             char(8) NOT NULL,
        "NumItens"          integer NOT NULL DEFAULT 0,
        "estado"            char(2) NOT NULL,
        "cidade"            char(20) NULL,
        "bairro"            char(20) NULL,
        "rua"               char(20) NULL,
        "numero"            char(12) NULL,
        "fone"              char(15) NULL,
        "email"             char(15) NULL,
        "homepage"          char(30) NULL,
        PRIMARY KEY ("codEmpresa"),
);
CREATE TABLE "alencar"."cliente"
(       "codCliente"        integer  NOT NULL UNIQUE,
        "nome"              char(40) NOT NULL,
        "cpf"               char(15) NOT NULL UNIQUE,
        "profissao"         char(20) NULL,
        "salario"           char(9)  NULL,
        "email"             char(15) NULL,
        "login"             char(15) NOT NULL UNIQUE,
        "senha"             char(15) NOT NULL,
        "cep"               char(15) NOT NULL,
        "rua"               char(20) NULL,
```

```
        "numero"            char(7) NULL,
        "complemento"       char(30) NULL,
        "bairro"            char(20) NULL,
        "cidade"            char(20) NULL,
        "estado"            char(2) NULL,
        "fone"              char(15) NULL,
        PRIMARY KEY ("codCliente"), );

CREATE TABLE "alencar"."Grupo"
(       "grupo"             char(15) NOT NULL,
        "tabela"            char(20) NOT NULL,
        "quantos"           integer NOT NULL,
        PRIMARY KEY ("grupo", "tabela"),
);

% - Mantém todas as listas de produtos que o cliente montou
% - no serviço de orçamento e/ou ou compras.
CREATE TABLE "alencar"."ListaProdutos"
(       "CodLista"          integer NOT NULL,
        "codCliente"        integer NOT NULL,
        "NumeroItens"       integer NOT NULL,
        "Data"              date NOT NULL DEFAULT current date,
        "TabelaReferencia"  char(15) NOT NULL,
        "valor"             numeric(6,2) NULL,
        "lista"             varchar(300) NULL,
        PRIMARY KEY ("CodLista", "codCliente"),
        FOREIGN KEY "cliente" ("codCliente")  REFERENCES "alencar"."cliente" ("codCliente")
                                                        on delete cascade,
);

CREATE TABLE "alencar"."ItemLista"
(       "codCliente"        integer NOT NULL,
        "codProduto"        integer NOT NULL,
        "CodLista"          integer NOT NULL,
        "quantidade"        integer NULL,
        PRIMARY KEY ("codCliente", "codProduto", "CodLista"),
        FOREIGN KEY "ListaProdutos" ("CodLista","codCliente")
                REFERENCES  "alencar"."ListaProdutos"  ( "CodLista",  "codCliente")
                        on delete cascade,
        FOREIGN KEY "produto" ("codProduto")
                REFERENCES "alencar"."produto" ("codProduto")
                                        on delete cascade;

);

CREATE TABLE "alencar"."produto"
(       "codProduto"        integer NOT NULL UNIQUE,
        "quantos"           integer NOT NULL,
        "subgrupo"          char(15) NULL,
        "apresentacao"      char(15) NULL,
        "marca"             char(15) NULL,
        "embalagem"         char(15) NULL,
        "unidade"           char(15) NULL,
        "descricao"         char(50) NULL,
        "grupo"             char(15) NOT NULL,
        "composicao"        varchar(70) NULL,
        PRIMARY KEY ("codProduto"),
);
```

```
CREATE TABLE "alencar"."ListaSugerida"
(       "cliente"               integer NOT NULL,
        "produto"               integer NOT NULL,
        "quantidade"            integer NOT NULL DEFAULT 1,
        "TabelaReferencia"  char(20) NOT NULL,
        PRIMARY KEY ("cliente", "produto", "TabelaReferencia"),  );
CREATE TABLE "alencar"."Subgrupo"
(       "grupo"                 char(15) NOT NULL,
        "subgrupo"              char(20) NOT NULL,
        "quantos"               integer NOT NULL DEFAULT 0,
        PRIMARY KEY ("grupo", "subgrupo"),
);

CREATE TABLE "alencar"."InfoRamo"
(
        "Ramo"                  char(20) NOT NULL,
        "tabela"                char(10) NOT NULL,
        "quantos"               integer NULL,
        PRIMARY KEY ("Ramo", "tabela"),
);

CREATE TABLE "alencar"."produto_tmp"
(
        "subgrupo"              char(15) NULL,
        "apresentacao"          char(15) NULL,
        "marca"                 char(15) NULL,
        "embalagem"             char(15) NULL,
        "unidade"               char(15) NULL,
        "descricao"             char(50) NULL,
        "grupo"                 char(15) NOT NULL,
        "composicao"            varchar(70) NULL,
);

CREATE TABLE "alencar"."Fornecimento"
(
        "codProduto"            integer NOT NULL,
        "codEmpresa"            integer NOT NULL,
        "Valor"                 numeric(6,2) NOT NULL,
        "ValorPromocao"         numeric(6,2) NULL,
        "AtendeuMaiorPreco"     integer NULL,
        "AtendeuMenoPreco"      integer NULL,
        PRIMARY KEY ("codProduto", "codEmpresa"),
        FOREIGN KEY "empresa" ("codEmpresa")
                REFERENCES "alencar"."empresa" ("codEmpresa")
                        on delete cascade,
);


% - mantém uma relação de produtos que foram procurados e
% - não encontrados em um estabelecimento em determinada data
CREATE TABLE "alencar"."ProdutoFalta"
(
        "codProduto"            integer NOT NULL,
        "codEmpresa"            integer NOT NULL,
        "data"                  date NOT NULL,
        PRIMARY KEY ("codProduto", "codEmpresa", "data"),
);
```

% - tabela que serve apenas para disparar um trigger que insere % - produtos a partir do alimentador de Dados via arquivo.

```
CREATE TABLE "alencar"."nada"
(       "nada"
                        integer NULL,
);

CREATE TABLE "alencar"."composicao"
(
        "codProduto"            integer NOT NULL,
        "componente"            char(15) NOT NULL,
        PRIMARY KEY ("codProduto", "componente"),
        FOREIGN KEY "codProduto" ("codProduto")
                REFERENCES "alencar"."produto" ("codProduto")
                                        on delete cascade,

);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Criação de Índices
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

CREATE UNIQUE INDEX "nome_index" ON "alencar"."empresa"
(       "nome" ASC          );

CREATE UNIQUE INDEX "cod_cli" ON "alencar"."cliente"
(       "codCliente" ASC    );

CREATE INDEX "nome_cli" ON "alencar"."cliente"
(       "nome" ASC          );

CREATE UNIQUE INDEX "indGrupo" ON "alencar"."Grupos"
(       "grupo" ASC         );

CREATE INDEX "grupoIndex" ON "alencar"."Grupos"
(       "grupo" ASC         );

CREATE INDEX "Subgrupo_Marca" ON "alencar"."produto"
(       "subgrupo" ASC, "marca" ASC     );

CREATE INDEX "indSub" ON "alencar"."Subgrupos"
(       "grupo" ASC,    "subgrupo" ASC     );

CREATE INDEX "indForn" ON "alencar"."Fornecimento"
(       "codProduto" ASC, "codEmpresa" ASC   );

CREATE INDEX "indVal" ON "alencar"."Fornecimento"
(       "Valor" ASC             );

CREATE INDEX "indComp" ON "alencar"."composicao"
(       "componente" ASC     );
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   Criação de triggers e funções
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

create function GeraCodProd()
returns integer
begin
  declare codigo integer;
  declare retorno integer;
  declare quantos integer;
  % -------------------------------------------------------------------%
  %       pega o máximo valor do código de produtos       %
  % -------------------------------------------------------------------%
  select count(*) into quantos from produto;
  if quantos=0 then
    return 1
  else
    select MAX(codProduto) into codigo from alencar.produto
  end if
  ;return codigo
end

create trigger ItensListaProdutos after insert order 1
on alencar.ListaProdutos
referencing new as nova_lista
for each row
begin
  declare quantositens integer;
  declare i integer;
  declare inicio integer;
  declare quantos integer;
  declare CodigoCliente integer;
  declare CodigoProduto integer;
  declare CodigoLista integer;
  declare Qtde integer;
  declare c varchar(1);
  declare tamanho integer;
  declare listaitens varchar(1000);
  set listaitens=nova_lista.lista;
  set quantositens=LENGTH(listaitens);
  set i=1;
  set c='';
  set inicio=1;
  set quantos=0;
  set CodigoLista=nova_lista.codlista;
  set CodigoCliente=nova_lista.codCliente;
  set CodigoProduto=0;
  set Qtde=0;
  while i<=quantositens loop
    set c=SUBSTR(listaitens,i,1);
    if c='|' then
      set CodigoProduto = cast( SUBSTRING( listaitens, inicio,  quantos) as integer);
      set inicio=i+1;
      set quantos=-1
    end if ;
    % fim do registro ou fim da lista
    if c='!' then
      set Qtde=cast( SUBSTRING( listaitens, inicio, quantos) as integer);
```

```
    set inicio=i+1;
    set quantos=-1;
    insert into ItemLista values( CodigoCliente, CodigoLista,   codigoproduto, Qtde)
   end if
   ;set quantos=quantos+1;
   set i=i+1
  end loop
end

create trigger ItensListaSugerida after insert order 2
on alencar.ListaProdutos
referencing new as nova_ListaProdutos
for each row
begin
 % ------------------------------------------------------%
 %                        Declare variables
 % ------------------------------------------------------%
 declare CodigoProduto integer;
 declare Qtde integer;
 declare tmp_codigo integer;
 declare CodigoCliente integer;
 declare CodigoLista integer;
 % ------------------------------------------------------%
 %  Uma excessão SQL que é "disparada" quando já se
 %      passou do    último registro
 % ------------------------------------------------------%
 declare linha_nao_encontrada exception for sqlstate value '02000';
 % ------------------------------------------------------%
 %  Cursor ItemDaLista que vai percorrer um result set    %
 %  contendo o código do produto e a maior quantidade     %
 % desse     já solicitado pelo cliente                   %
 % ------------------------------------------------------%
 declare ItemDaLista
        dynamic scroll cursor for
                select codProduto,max(quantidade) from itemLista
                where codlista  = any(select codlista from listaProdutos
                                      where listaProdutos.codCliente =
                                            nova_ListaProdutos.codCliente
                                           and
                                       TabelaReferencia = nova_ListaProdutos.TabelaReferencia)
               group by codProduto;
 % ------------------------------------------------------%
 %  Inicializa as variáveis
 % ------------------------------------------------------%
 set CodigoProduto=0;
 set CodigoLista=nova_ListaProdutos.CodLista;
 set Qtde=0;
 set CodigoCliente=nova_ListaProdutos.codCliente;
 % ------------------------------------------------------%
 %   Abre o cursor
 % ------------------------------------------------------%
 open ItemDaLista;
 % ------------------------------------------------------------------------%
 %   Exclui a última lista sugerida desse cliente.                        %
 %  Para cada cliente é mantida somente uma lista sugerida    %
 % ------------------------------------------------------------------------%
 delete from ListaSugerida where cliente=CodigoCliente;
```

```
 % ------------------------------------------------------------%
 % Percorre as linhas do result set                           %
 % ------------------------------------------------------------%
 ItemLoop: loop
  fetch next ItemDaLista into CodigoProduto,Qtde;
  if sqlstate=linha_nao_encontrada then
   leave ItemLoop
  end if   ;
insert into ListaSugerida values( CodigoCliente, CodigoProduto,
                                  Qtde, nova_ListaProdutos.TabelaReferencia)
 end loop ItemLoop ;
 close ItemDaLista
end
go  % finaliza

create trigger InsereSubgrupos after insert order 2
on alencar.produto
referencing new as novo_produto
for each row
begin
 if exists(select quantos from Subgrupos
           where novo_produto.grupo=grupo and
                 novo_produto.subgrupo=subgrupo)
 then
  update subgrupos set quantos=quantos+1
  where novo_produto.grupo = grupo and
          novo_produto.subgrupo=subgrupo
 else
        insert into Subgrupos values(novo_produto.grupo,novo_produto.subgrupo,1)
 end if
end
go

create trigger InsereGrupos after insert order 1
on alencar.produto
referencing new as novo_produto
for each row
begin
 declare linha_nao_encontrada exception for sqlstate value '02000';
 declare tmp_grupo char(20);
 declare Qtde integer;
 declare GrupoProduto dynamic scroll cursor for
              select grupo,quantos from Grupos where grupos.grupo=novo_produto.grupo;
 set Qtde=0;
 set tmp_grupo=' ';
 open GrupoProduto;
 fetch next GrupoProduto into tmp_grupo,Qtde;
 if sqlstate=linha_nao_encontrada then
  insert into Grupos values(novo_produto.grupo,'produto',1)
 else
  update Grupos set quantos=quantos+1 where grupo=novo_produto.grupo
 end if;
 close GrupoProduto
end
go
```

```
create trigger InsereProduto_AD after insert order 1
on alencar.nada
for each row
begin
 % ----------------------------------------------------%
 %  Declaração das variáveis
 % ----------------------------------------------------%
 declare quant integer;
 declare numProd integer; % para checar se o produto já existe
 declare codProd integer;
 declare subgrp char(15);
 declare apres char(15);
 declare marc char(15);
 declare emba char(15);
 declare unid char(15);
 declare descr char(50);
 declare grup char(15);
 declare comp char(70);
 % ----------------------------------------------------------------------------------------------------%
 %  Uma excessão SQL que é "disparada" quando já se passou do último registro %
 % ----------------------------------------------------------------------------------------------------%
 declare linha_nao_encontrada exception for sqlstate value '02000';
 % ----------------------------------------------------------------------%
 % Cria um cursor para percorrer a tabela produto_tmp          %
 % ----------------------------------------------------------------------%
 declare prodTemp dynamic scroll cursor for select* from produto_tmp;
 % ----------------------------------------------------------------------%
 % Gera um novo código para o produto                          %
 % ----------------------------------------------------------------------%
 set codProd=GeraCodProd(*);
 % ----------------------------------------------------%
 %  Abre o cursor prodTemp
 %----------------------------------------------------%
 open prodTemp;
 prodLoop: loop
  fetch next prodTemp into subgrp,apres,marc,emba,unid, descr,grup,comp;
  if sqlstate=linha_nao_encontrada then
   leave prodLoop
  end if
  % ----------------------------------------------------------------------------------------------------%
  %  Obtem o número de colunas a considerar quando da  solicitação de um serviço  %
  % ----------------------------------------------------------------------------------------------------%
  select quantos into quant  from InfoRamo where tabela='produto';
%----------------------------------------------------%
%  Usado para verificar se o produto que está sendo
%  Inserido ainda não consta na tabela de produtos
%----------------------------------------------------%
  select count(*) into numProd from produto
       where subgrupo=subgrp and marca=marc and embalagem=emba  and unidade=unid;
  % ----------------------------------------------------%
  %  caso o produto não tenha sido encontrado
   ----------------------------------------------------%
  if numProd=0 then
   insert into produto values( codProd,quant,subgrp,apres,marc,emba, unid,descr,grup,comp);
   set codProd=codProd+1
  end if
 end loop prodLoop
 ;delete from nada;
```

```
 close prodTemp
end
go

create trigger ExcluiSubgrupo after delete order 2
on alencar.produto
referencing old as produto_excluido
for each row
begin
  update Subgrupos set quantos=quantos-1
    where grupo=produto_excluido.grupo
    and subgrupo=produto_excluido.subgrupo
    and quantos>0
end
go

create trigger CadastraRamo after insert order 2
on alencar.empresa
referencing new as nova_empresa
for each row
%-------------------------------------------------------------------------%
%   Se o ramo comercial dessa empresa ainda não existe ...  %
%-------------------------------------------------------------------------%
    when(not exists(select Ramo
          from InfoRamo where Ramo=nova_empresa.ramocomercial))
   begin
            declare tabelatemp char(20);
            declare quantos integer;
            set quantos=0;
            set tabelatemp='';
            if nova_empresa.ramocomercial='Supermercado' then
                    set tabelatemp='produto';
                    set quantos=6
            end if
            ;if nova_empresa.ramocomercial='Farmácia' then
                    set tabelatemp='prodFarm'
            end if  ;
            if nova_empresa.ramocomercial='Posto de Gasolina' then
                    set tabelatemp='prodPosto'
            end if  ;
            if nova_empresa.ramocomercial='Livraria' then
                     set tabelatemp='prodLiv'
            end if  ;
            insert into InfoRamo values(nova_empresa.ramocomercial,tabelatemp,quantos)
    end
go

create trigger ExcluiGrupo after delete order 1
on alencar.produto
referencing old as produto_excluido
for each row
begin
  update Grupos set quantos=quantos-1
    where grupo=produto_excluido.grupo    and quantos>0
end
go
```