UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE CIÊNCIAS DA COMPUTAÇÃO

(Bacharelado)

APLICATIVO PARA ATUALIZAÇÃO DE BANCO DE DADOS UTILIZANDO ACTIVE SERVER PAGES (ASP)

TRABALHO DE CONCLUSÃO DE CURSO SUBMETIDO À UNIVERSIDADE REGIONAL DE BLUMENAU PARA A OBTENÇÃO DOS CRÉDITOS NA DISCIPLINA COM NOME EQUIVALENTE NO CURSO DE CIÊNCIAS DA COMPUTAÇÃO — BACHARELADO

ADRIANO DIAS

BLUMENAU, JUNHO/2002.

2002/1-01

APLICATIVO PARA ATUALIZAÇÃO DE BANCO DE DADOS UTILIZANDO ACTIVE SERVER PAGES (ASP)

ADRIANO DIAS

ESTE TRABALHO DE CONCLUSÃO DE CURSO FOI JULGADO ADEQUADO PARA OBTENÇÃO DOS CRÉDITOS NA DISCIPLINA DE TRABALHO DE CONCLUSÃO DE CURSO OBRIGATÓRIA PARA OBTENÇÃO DO TÍTULO DE:

BACHAREL EM CIÊNCIAS DA COMPUTAÇÃO

_____

Prof. Wilson Carli — Orientador na FURB

_____

Prof. José Roque Voltolini da Silva — Coordenador do TCC

BANCA EXAMINADORA

_____

Prof. Wilson Carli

_____

Prof. Maurício Capobianco Lopes

_____

Prof. Marcel Hugo

SUMÁRIO

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS

ASP - *Active Server Pages*

B2B - *Bussines-to-Bussines*

B2C - *Bussines-to-Consumer*

ODBC - *Open Database Connectivity*

SQL - *Structuded Query Language*

WWW - *World Wide Web*

HTML - *HyperTex Markup Language*

IIS - *Internet Information Server*

COM - *Component Object Model*

ADO – *Active Data Object*

DSN - *Data Source Name*

MER - *Modelo Entidade Relacionamento*

OLE DB - *Object Linking and Embedding Database*

RESUMO

Este trabalho consiste no desenvolvimento de uma ferramenta para criação de páginas *Active Server Pages* (*ASP*) que permitem atualizar o conteúdo de um banco de dados. Através da linguagem *ASP*, utilizando um *template* pré-definido são criadas as principais páginas necessárias para pesquisar, modificar e criar novos registros em um banco de dados.

ABSTRACT

This work consists the development of a tool for generation pages Active Server Pages (ASP) that they allow to bring up to date the content of a data base. Through language ASP using one template daily pay-defined is generated the main pages necessary to search, to modify and to create new registers in a data base.

1 INTRODUÇÃO

O desenvolvimento de tecnologias para Internet é uma das áreas mais promissoras na atualidade, já que cada vez mais as empresas estão buscando a elaboração de *sites* com conteúdo dinâmico que possam aumentar a interação entre empresas *Bussines-to-Bussines (B2B*) ou entre empresa e cliente *Bussines-to-Consumer (B2C)*.

Segundo Homer (2001) os *sites* estáticos, onde o usuário sempre visualiza a mesma informação a cada visita, estão sendo substituídos por *sites* dinâmicos que utilizam banco de dados que são atualizados através de páginas dinâmicas. Na construção destes *sites* dinâmicos é necessária a elaboração de páginas que possibilitem a atualização da base de dados remotamente através da *Internet*.

Para criar as páginas necessárias para a atualização dos dados dos produtos de uma empresa, ter-se-ia uma página de procura, onde o usuário digita o código ou uma palavra chave relacionada com o produto, e em seguida seria mostrado uma página com a listagem dos produtos encontrados. Clicando no código do produto assestar-se-ia uma página de detalhes deste produto, onde seria possível fazer alterações no produto.

O aplicativo apresentado visa automatizar o processo de criação dessas páginas. O aplicativo baseia-se em um modelo pré-definido, onde o usuário irá fornecer ao sistema, quais as tabelas e quais os campos são necessários e o sistema criará automaticamente o código em *Active Server Pages (ASP)* das páginas que serão utilizadas para a atualização do banco de dados.

O aplicativo foi desenvolvido com a principal linguagem do mercado para desenvolvimento de *sites* dinâmicos, *Active Server Pages (ASP)* e é compatível com os bancos de dados *ORACLE* e *SQL Server*, bem como permite o acesso a base de dados via conexão *Open Database Connectivity (ODBC)*. Segundo Wille (1999), o *ASP* é um ambiente de *script* de servidores que permite que se crie um *site* da *Web* dinâmico, rápido e interativo, sem precisar preocupar-se com os recursos de *browser* dos usuários.

A principal justificativa para o desenvolvimento do aplicativo está relacionado ao alto ganho de produtividade que será obtido no desenvolvimento das páginas, pois se tem a automatização na criação das páginas e formulários *ASP*. Essas páginas são criadas de forma parametrizada através de telas de configuração, dispensando a intervenção do programador no código criado.

1.1 OBJETIVOS

O objetivo principal é fornecer uma ferramenta que auxilie na criação de páginas para atualização de banco de dados utilizando Active Server Pages.

Os objetivos específicos do trabalho são:

a) criar código *ASP* para atualização de dados permitindo pesquisar, editar, inserir, alterar e excluir registros;
b) ser compatível com os bancos de dados *SQL Server*, *ORACLE*, *Microsoft Access* e permitir também conexões via *ODBC*;
c) fornecer opções para personalização das páginas criadas, onde o usuário irá indicar os atributos de cada página;
d) fornecer várias opções de cores para garantir que as páginas criadas não sejam sempre visualmente parecidas.

1.2 ESTRUTURA DO TRABALHO

O primeiro capítulo apresenta uma introdução sobre o assunto e objetivos do trabalho.

O segundo capítulo contextualiza os conceitos gerais relacionados com o trabalho, sobre a Internet, banco de dados e principalmente sobre a linguagem *ASP*, que é o foco do trabalho.

O terceiro capítulo descreve a estruturação necessária para o desenvolvimento do aplicativo até seu estágio final, destacando a metodologia utilizada, a modelagem do sistema e considerações sobre a codificação.

O quarto capítulo conclui o trabalho, apresenta limitações e também sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo apresenta-se os principais conceitos sobre os assuntos abordados neste trabalho.

2.1 INTERNET

A *Internet* é uma rede global de computadores que se comunicam usando uma linguagem comum. Quando se conecta a um *site* da *web* está-se conectado a *Internet*. É como usar o sistema telefônico internacional - ninguém é dono ou controla o sistema como um todo, mas as conexões são feitas de tal maneira que ele funciona como uma grande rede. A *World Wide Web* (a *web* ou *WWW*) oferece uma interface gráfica na qual é fácil navegar para observar documentos na *Internet*. Esses documentos, assim como os *links* existentes entre eles, compõem uma "teia" de informações.

Segundo Microsoft (2002), a *World Wide Web* está mudando o modo como as pessoas se comunicam no mundo todo. Essa nova mídia global vem aumentando de popularidade mais rapidamente que qualquer outro meio de comunicação em toda a história. Nos dois últimos anos, a w*eb* cresceu e passou a incluir uma enorme variedade de informações, de ações nas bolsas de valores a oportunidades profissionais, de serviços de mensagens a notícias, lançamentos da indústria cinematográfica, revistas literárias e jogos. Os tipos de informações variam dos mais irrelevantes até os mais importantes em nível mundial. Muitas vezes as pessoas falam em "surfar" pela *web* e visitar novos *sites*. "Surfar" significa seguir os *hyperlinks* para páginas e assuntos sobre os quais jamais se ouviu falar, encontrar novas pessoas, visitar novos lugares e aprender coisas do mundo inteiro.

2.2 CONSTRUÇÃO DE *SITES* E EXTRANET

A principal diferença entre *sites* estáticos e *sites* dinâmicos é a capacidade adicional que o *site* dinâmico tem de interatividade, permitindo o uso de informação fluindo em ambos os sentidos em tempo real. Os *sites* dinâmicos são *sites* que possuem capacidade de receber e processar informação de seus visitantes. A figura 1 ilustra a diferença que existe no fluxo de informação entre *sites* dinâmicos e *sites* estáticos.

Figura 1 – *Site* Estático x *Site* Dinâmico

Fonte: Microsoft (2002)

O custo de produção de *sites* dinâmicos é mais elevado, mas quando computado com o custo de manutenção de *sites* estáticos por um período de um ano, a redução do custo final é extraordinária. Com um site estático, para se mudar qualquer informação, é necessário o uso de profissionais qualificados, enquanto com o site dinâmico, mudanças são feitas simplesmente editando-se textos no banco de dados do site. Os *sites* dinâmicos tem seu conteúdo atualizado através de um banco de dados.

Segundo Pfaffenberger (2000) pode-se definir a *Extranet* como sendo uma área do site de uso exclusivo da empresa e/ou de seus clientes. Por exemplo, uma empresa com várias filiais precisando de comunicação urgente com uma outra localizada a vários milhares de quilômetros; usa-se, então, um provedor de acesso local para a interligação entre as duas filiais. Normalmente o acesso a *Extranet* de uma empresa é limitado por um *login* e senha, permitindo o acesso às informações somente por pessoas autorizadas.

2.3 ACTIVE SERVER PAGES - ASP

Neste item apresentam-se as principais características do *ASP*, linguagem que será utilizada na implementação do aplicativo e também será a linguagem encontrada nas páginas criadas pelo sistema.

Segundo Mitchell (2000), as *Active Server Pages* são a solução da *Microsoft* para criar páginas da *web* dinâmicas. Inicialmente os primeiros *sites* da internet consistiam basicamente de páginas fáceis de ler com imagens gráficas, todas estáticas, ou seja, o seu conteúdo era sempre o mesmo a cada visita. Com a evolução da *Internet* surgiu a necessidade de criar *sites* interativos e mais programáveis.

O *ASP* é um ambiente de programação e não uma linguagem de programação, onde se pode combinar *HyperText Markup Language (HTML)*, *scripts* e outros componentes (em qualquer linguagem), para se criar aplicações dinâmicas e poderosas no servidor *web*. Este ambiente foi criado pela *Microsoft* para o *Internet Information Server* (IIS). Os arquivos criados com *ASP* têm a extensão ".asp" e devem ser executados em um servidor *web* compatível com *ASP*.

Segundo Russell (2001), o *ASP* permite usar o poder de um servidor *web* para processar solicitações de usuários e fornecer conteúdo dinâmico e individualizado baseado em dados lógicos, de arquivos e de banco de dados, permitindo ainda, processar dados individualizados do usuário. O *ASP* permite, também, tratar os usuários como entidades exclusivas, mesmo que todos eles estejam executando o mesmo programa, na mesma máquina.

Resumidamente pode-se afirmar que o *ASP* fornece:

a) uma maneira de salvar dados individualizados para cada usuário;

b) acesso a sistema de arquivos;

c) acesso a banco de dados;

d) um meio de disparar e controlar qualquer componente *Component Object Model*, modelo de objeto componente *(COM)*.

O *ASP* é basicamente dividido em duas partes: o código programático e o *HTML*, sendo que o código programático pode ser escrito em linguagens de *script*, como *Vbscript* e *Jscript*.

A parte programática do código *ASP* fica delimitada pelas *tags* <% e %>, que é designado como código *ASP*. Isso é o que o servidor processa antes de enviar a página para o navegador da *web*.

2.3.1 MODELO CLIENTE-SERVIDOR

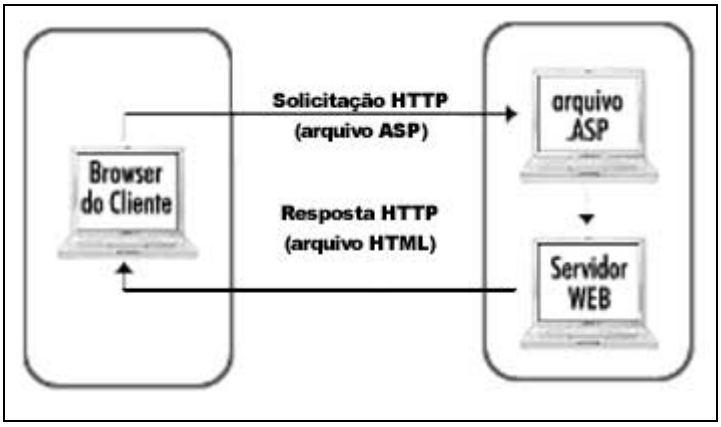
Segundo Mitchell (2001), "Em um modelo cliente-servidor, dois computadores trabalham juntos para realizar uma tarefa. Um computador cliente solicita algumas

informações necessárias de um computador servidor. O servidor retorna essas informações e o cliente interage com elas.”

A *Internet* também utiliza o modelo cliente-servidor, onde o servidor é um servidor da *web* particular que contém as páginas do site e tem um *software* especial instalado para enviar as páginas para os navegadores *web* que as solicitaram.

Com a utilização do *ASP* todo o processamento da parte programática é feito no lado servidor sendo retornado ao cliente somente o código *HTML*. A figura 2 ilustra a arquitetura cliente-servidor utilizando *ASP*:

Figura 2 – Arquitetura cliente-servidor



Fonte: Mitchell (2001)

Quando o servidor de *web* recebe a solicitação para a uma página *ASP*, invoca o mecanismo do *ASP* que, por sua vez, invoca a linguagem de *script* utilizada e executa o código no servidor. A linguagem de *script* por si só, não oferece recursos suficientes para interagir com o cliente. Para isto o *ASP* oferece os objetos embutidos que possibilitam uma maior interação com o *browser* do cliente, acessando recursos do servidor e muito mais.

Segundo Mitchell (2001) o *ASP* não é uma linguagem. É uma tecnologia de servidor. A linguagem que se utiliza para programar é na realidade o *VBScript*. Contudo, *VBScript* não é a única linguagem que se pode utilizar com *ASP*. A definição da linguagem utilizada é configurada na primeira linha do código *ASP* através da palavra chave *Language*.

2.3.2 OBJETOS DO ASP

O *ASP* possui alguns objetos básicos para criar suas aplicações. Através destes objetos pode-se executar diversas funções que vão desde variáveis até acesso a bancos de dados.

O objeto *Session* é ideal para aplicações dinâmicas como, por exemplo, loja virtual. Cada vez que uma aplicação é acessada por um usuário, é aberta uma seção no servidor, a qual é encerrada após o termino da aplicação.

Ele define variáveis para usuários individuais, mantendo a aplicação aberta por um período de tempo. Por exemplo, se o cliente estiver fazendo um pedido em uma loja virtual e ele mantiver a aplicação aberta por um período de tempo, sem enviar as informações ao banco de dados, todas as informações por ele adicionadas serão automaticamente apagadas.

O objeto *Response* é o mais utilizado em *ASP*. Ele controla os dados a serem enviados para o cliente. Estes dados podem ser *HTML* (Método *Write*), *cookies* (*response.cookies*), validade de um *cookie* (Atributo *Expires*), etc..Exemplo:

```
<% Response.Write “Bom dia”>
```

Através do objeto *Server*, pode-se carregar diversos componentes, podendo-se, inclusive, acessar bancos de dados.

2.3.3 ASP E BANCO DE DADOS

Segundo Wille (1999), um banco de dados é uma coleção de informações que podem ser facilmente consultadas e modificadas. Um banco de dados possui as seguintes propriedades:

a) é uma coleção lógica coerente de dados com um significado inerente; uma disposição desordenada dos dados não pode ser referenciada como um banco de dados;

b) um banco de dados é projetado e construído com dados para um propósito específico, possuindo um conjunto pré-definido de usuários e aplicações;

c) um banco de dados representa algum aspecto do mundo real, o qual é chamado de “mini-mundo”; qualquer alteração efetuada no mini-mundo é automaticamente refletida no banco de dados.

O *ASP* dispõe de recursos para ler e modificar o conteúdo de um banco de dados, através de uma coleção de objetos *Activex Data Object (ADO)*.

2.3.4 BANCOS DE DADOS RELACIONAIS

Quando se utiliza um banco de dados, pode-se fazer basicamente quatro coisas: recuperar dados, inserir dados, atualizar dados existentes ou excluir dados existentes. O principal propósito de um banco de dados é armazenar informações. Embora cada sistema de banco de dados possa ter suas próprias características, cada banco de dados utiliza tabelas para armazenar informações. Segundo Mitchell (2000) uma tabela é uma matriz bidimensional que é utilizada para armazenar informações em um banco de dados.

A tabela é constituída por uma combinação de linhas e colunas no formulário de uma matriz. As colunas de uma tabela descrevem as propriedades do objeto, enquanto cada linha é uma instância única do objeto. As linhas são freqüentemente chamadas de registros e as colunas de campos.

Normalmente os bancos de dados constituem-se de várias tabelas e cada tabela representa um único objeto. Mas e se um objeto contiver instâncias de tipos de objetos diferentes? Por exemplo, se uma das colunas de uma tabela constituir de um objeto, pode-se criar uma nova tabela com as características do novo objeto e fazer o relacionamento entre as tabelas. Segundo Wille (1999) um banco de dados relacional representa cada objeto, relacionado ou não como sua própria tabela de banco de dados.

Muitos sistemas comerciais de banco de dados relacional estão disponíveis no mercado. Segundo Mitchell (2000) entre os mais populares encontra-se o *Microsoft SQL Server*, *Microsoft Access*, *Oracle e Informix*. Com o *ASP* não se está limitado a utilizar somente esses quatro sistemas de banco de dados. Pode-se acessar qualquer banco de dados compatível com *ODBC*. Um banco de dados compatível com *ODBC* obedece a padrões de conectividade definidos pela *Microsoft*.

Os bancos de dados fornecem o melhor desempenho para armazenar informações longas e heterogêneas. Os bancos de dados são projetados para simplificar o processo de armazenar e recuperar informações. No entanto, a conexão com o banco de dados transporta com ela um alto custo de desempenho. Os bancos de dados somente devem ser utilizados para armazenar grandes quantidades de informações sobre períodos indefinidos de tempo.

2.3.5 TIPO DE CAMPOS

Os tipo de campos em bancos de dados definem o conteúdo que o campo armazena, por exemplo, um tipo *number* no *Access* poderá armazenar somente números. Os bancos de dados populares oferecem muitos dos mesmos tipos de dados, embora possam utilizar vocabulário diferente. No quadro 1 tem-se um exemplo de uma listagem de tipos de dados suportado pelo *Microsoft Access*, esses campos são relevantes, pois através do tipo de campo será criado código HTML diferente para cada campo como será mostrado na implementação. Para os bancos de dados *Microsoft SQL Server* e *Oracle* os campos são similares com algumas diferenças que foram tratadas no aplicativo.

Quadro 1 – Tipos de dados suportados pelo Access

Tipo de dados	Quando Utilizar
Text	Utilize quando precisar armazenar 255 caracteres alfanuméricos ou menos.
Memo	Utilize quando precisar armazenar uma quantidade grande de caracteres alfanuméricos. O tipo de dados Memo pode armazenar até 65.535 caracteres
Number	Utilize quando precisar armazenar valores numéricos
Data/Hora	Utilize quando precisar armazenar valores de data e hora. Um tipo de dados Date/Time no Access pode armazenar datas entre os anos 100 e 9999
Moeda	Utilize quando precisar armazenar valores de moeda. Os tipos de dados Currency somente suportam até quatro dígitos de precisão decimal
Sim/Não	Utilize para colunas que somente ter um de dois valores
Autonumber	Utilize essa variável de incremento automaticamente para identificar de maneira única cada linha em uma tabela. Os campos Autonumber são essenciais para projetar múltiplas tabelas relacionadas em um banco de dados relacional

Fonte: Mitchell (2000)

2.3.6 MODELO DE OBJETOS ADO

Segundo Wille (1999) o *ADO* é um conjunto de interfaces por meio do qual pode-se acessar os dados *Object Linking and Embedding Database* (*OLE DB)* com qualquer linguagem, incluindo *VBScript*. O *OLE DB* é um conjunto de interfaces de programação no nível de sistema que fornece acesso a diversas informações e fontes de dados. Como o *OLE DB* está no nível de sistema, o programador tem controle completo sobre todas as interfaces.

Entretanto, só pode ser utilizado a partir do *C++*. Já o *ADO*, é um objeto de acesso de dados no nível de aplicativo que é fácil de usar, neutro quanto à linguagem de programação, podendo ser acessado em *VBScript*, *JScript*, *Visual Basic* e quaisquer outras linguagens que possam tratar dos objetos.

A seguir tem-se os principais objetos do ADO:

a) *connection* – representa a conexão com uma fonte de dados;

b) *recordset* – consiste em registros retornados de uma consulta de banco de dados e um cursor para esses registros;

c) *field* - contém dados de uma única coluna e informações sobre esses dados. O objeto *recordset* fornece a coleção *fields*, que contém todos dos objetos *field* de um *recordset*;

d) *property* – contém informações dinâmicas sobre um objeto *ADO*, fornecido pelo provedor básico;

e) *error* – contém informações estendidas de erro retornadas pelo provedor;

f) *command* – fornece a possibilidade de definir comandos específicos que se pretende executar em um banco de dados várias vezes com parâmetros diferentes.

Antes de fazer qualquer coisa com um banco de dados é preciso conectar-se a ele. O objeto *connection* do *ADO* é utilizado para armazenar informações sobre os dados que se quer acessar. Ele é criado utilizando *Server.CreateObject*, conforme o quadro 2.

Quadro 2 – Criando objeto *Connection*

```
Set objConn = Server.CreateObject(“ADODB.Connection”)
```

O provedor de acesso mais fácil de utilizar com o *ASP* é o *ODBC*. Ele oferece a possibilidade de criar o chamado *DSN* (*Data Source Names*), por meio do qual pode-se conectar facilmente a qualquer tipo de banco de dados.

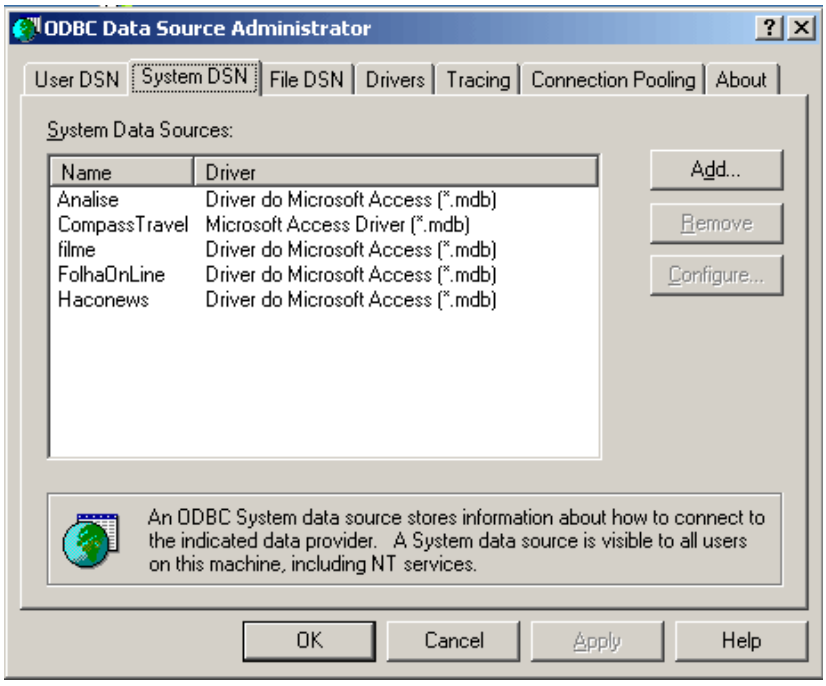
Para exemplificar, a seguir é explicado como criar uma *DSN* no *Windows* para conectar-se a um arquivo do *Microsoft Access*, com os bancos SQL Server e Oracle o também podemos estabelecer a conexão DSN, no entanto, neste caso precisa-se escolher o nome do banco, bem como configurar os dados para o usuário.

a) clica-se em *Iniciar*, *Configurações*, *Painel de Controle* para abrir o *Painel de Controle;*

b) abre-se o *ODBC Data Source Administrator*. Para tanto, clica-se duas vezes em *ODBC* ou *ODBC Data Sources* no *Windows NT* ou em *32-bit ODBC* para o *Windows 98;*

c) vá para a guia *System DSN*. Deve-se ver algo como a Figura 3. Há uma listagem das origens de dados de sistema atualmente configuradas. Há três botões do lado direito que permitem que se adicione um novo *System DSN*, remova o selecionado ou altere as configurações do atualmente selecionado;

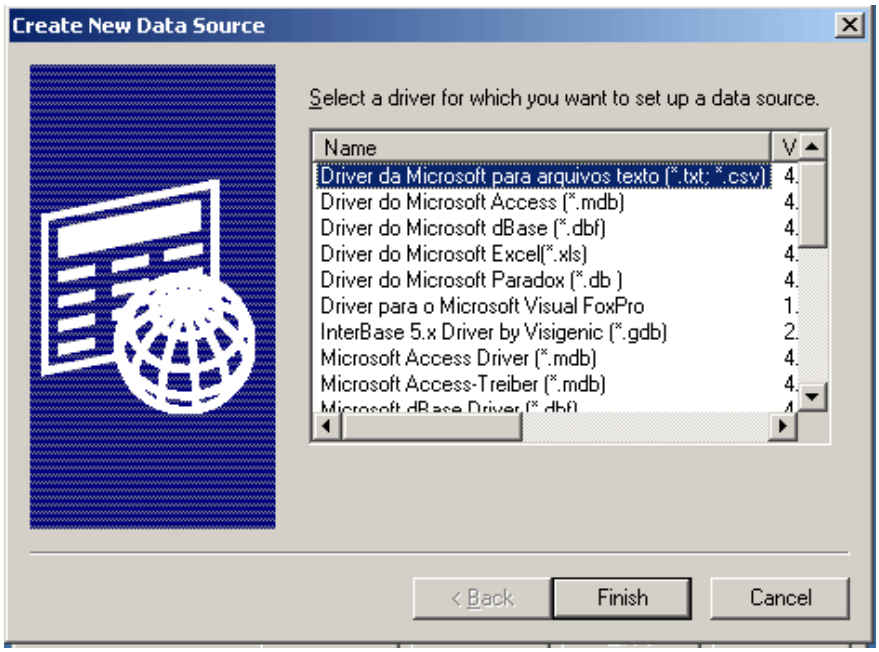
Figura 3 – Visualizando a guia *Data Source Administrator System DSN*



d) para criar um novo *DSN*, clica-se em *ADD*;

e) agora se tem uma lista de *drivers*, como na Figura 4. Escolha *Microsoft Access Driver (*.mdb)* e clica-se em *Finish*;

Figura 4 – Criando um novo *System DSN* para o banco de dados



f) agora a caixa de configuração se abre. Insere-se o *Data Source Name*, que é o nome que se utiliza para referenciar o *DSN* nas páginas *ASP*;

g) clica-se em *Select*. Agora se escolhe o arquivo *mdb*;

h) clica-se em *OK* para selecionar o banco de dados e novamente em *OK* para sair.

Para abrir a conexão deve-se utilizar o método *Open* do objeto *Connection*, conforme mostra o quadro 3. Para fechar a conexão ao objeto utiliza-se o método *close* e para liberar memória e configura-se o objeto como *nothing* conforme o quadro 3.

Quadro 3 – Exemplo de criação de conexão em ASP

```
<%
set objConn = Server.CreateObject("ADODB.Connection")
objconn.ConnectionString = "DSN=TCC"
objconn.Open
...
objconn.Close
set objConn = Nothing
%>
```

2.3.7 UTILIZANDO UMA CONEXÃO SEM DSN

Há uma alternativa a utilizar o *System DSN*. Em vez de colocar as informações da conexão no *DSN*, pode-se colocá-las na *string* de conexão conforme quadro 4.

Quadro 4 – Exemplo de criação de conexão em ASP sem DSN

```
ObjConn.ConnectionScring =”DRIVER={Microsoft Access Driver (*.mdb));
DBQ=C:\MeusDocumentos\arquivo.mdb”
```

A linha *“Driver=”* informa ao objeto *Connection* o tipo de banco de dados ao qual ele deve se conectar. A linha *“DBQ=”* indica o nome e o caminho do arquivo.

A diferença entre a conexão com e sem DSN, está no fato que com a DSN utiliza-se o driver ODBC de OLEDB, enquanto que a conexão sem DSN não é utilizado este driver fazendo uma conexão direta com o banco de dados.

Segundo Mitchell (2000) é difícil dizer com certeza qual é a maneira mais eficiente para se conectar a um banco de dados: com DSN ou sem DSN. A página de dicas de desempenho da Microsoft diz que utilizar um System DSN é mais rápido. Outras fontes discordam. Testes recentes parecem indicar, porém, que as conexões sem DSN são de fato ligeiramente melhores.

2.3.8 LENDO E ALTERANDO DADOS DE UM BANCO DE DADOS

Após a conexão com o banco de dados pode-se ler os dados do banco. Para isso precisa-se trabalhar com outros objetos.

Um dos principais objetos do ASP é o *recordset* que segundo Mitchell (2000) um *recordset* é simplesmente um conjunto de registros que pode ser utilizado para conter um subconjunto dos registros em uma tabela ou mesmo todos os registros da tabela. O quadro 5 mostra a sintaxe para criar e abrir um objeto *recordset*. O método “open” pode aceitar muitos conjuntos diferentes de argumentos e pode ser utilizado de maneiras diferentes. A sua sintaxe encontra-se no quadro 5.

Quadro 5 – Sintaxe do objeto *Recordset*

Recordset.Open origem, conexao, tipodecursor, tipodebloqueio, tipodecomando
origem é uma *string* contendo um comando reconhecido
conexão é um objeto *Connection* ou uma *string* contendo as informações de conexão.
tipodecursor indica a maneira como se pode mover no *recordset*.
tipodebloqueio determina se pode gravar na tabela e, se puder, como.
tipodecomando indica como o parâmetro *origem* dever ser avaliado.

O valor padrão do *tipodecursor* é a constante de *ADO adOpenForwardonly*, onde se pode ir somente para frente no *recordset*, ou seja, pode-se ir ao próximo registro, mas não se pode retornar ao registro anterior. Um outro tipo de cursor utilizado é o *Dynamic* (*adOpenDynamic*) – o movimento através do conjunto de registros é permitido tanto para frente quanto pra trás.

Para selecionar os registros de uma tabela pode-se utilizar a *Structuded Query Language* (*SQL*). Segundo Wille (1999), a *SQL* é usada para consultar, atualizar e gerenciar os dados dos banco de dados relacionais. Usando a *SQL*, pode-se recuperar, filtrar e classificar dados específicos de um banco de dados.

No quadro 6 mostra um exemplo completo utilizando o objeto *recordset*, neste exemplo pode-se visualizar a utilização do método *movenext* do *recordset* utilizado para fazer a varredura de todos os registros.

Quadro 6 – Exemplo utilizando o *Recordset*

```
<@ Language=VBScript%>
<!--#include virtual=”/adovbs.inc”-->
<%
‘Abre um conexão com a banco de dados
set objConn = server.CreateObject(“ADODB.Connection”)
objConn.ConnectionString = “DSN=TCC”
objConn.Open

‘Cria uma instância do objeto recordset e recuperar as informações
‘da tabela filmes
set objRS = Server.CreateObject(“ADODB.RecordSet”)
objRS.Open = “FILMES”, objConn,,,adCmdTable

‘Exibe o conteúdo da tabela
While not objRS.EOF
        Response.write objRs(“NOME”) & “<br>”
        ObjRS.MoveNext
  Wend

  ‘Limpa os objetos ADO
  objRS.Close
  set objRS = Nothing

objConn.Close
set objConn = Nothing
%>
```

Uma maneira mais direta para se criar um *recordset* utilizando um comando SQL é através da seguinte linha de comando no *ASP* do quadro 7:

Quadro 7 – Exemplo executando uma SQL

```
SET objRS = objConn.Execute(strQuery)
```

Nesta *strQuery* deve ser utilizado um comando SQL, como por exemplo: “Select * from Usuarios”.

Existem duas maneiras para inserir, modificar ou apagar os dados em um tabela, pode-se utilizar diretamente os comandos *SQL*, ou através dos métodos encontrados no *recordset*: *ADDNEW* para novo registro, *UPDATE* para alteração, *DELETE* para apagar. No quadro 8 tem-se as duas situações ilustradas.

Quadro 8 –Alterando o conteúdo da Tabela

```
<@ Language=VBScript%>
<!--#include virtual=”/adovbs.inc”-->
<%
‘Abre um conexão com a banco de dados
set objConn = server.CreateObject(“ADODB.Connection”)
objConn.ConnectionString = “DSN=TCC”
objConn.Open

‘Cria uma instância do objeto recordset e recupera as informações
‘da tabela filmes
set objRS = Server.CreateObject(“ADODB.RecordSet”)
objRS.Open = “FILMES”, objConn,,,adCmdTable

‘Criando um novo registro com o método ADDNEW
objRS.AddNew
objRS(“NOME”) = “O Ultimo dos Moicanos”
objRS.Update

‘Alterando o conteúdo do registro atual com o método UPDATE
objRS(“NOME”) = “O Ultimo dos Moicanos”
objRS.Update

‘Apagando o Registro atual com o método DELETE
objRS.Delete

‘Criando um novo registro utilizando SQL
objConn.EXECUTE(“INSERT INTO FILMES VALUES (‘O ultimo dos Moicanos’)”

‘alterando um registro utilizando SQL
objConn.EXECUTE(“UPDATE FILMES SET NOME = ‘O ultimo dos Moicanos’ _
      WHERE COD_FILME = 1”)

‘excluindo um registro utilizando SQL
objConn.EXECUTE(“DELETE FROM FILMES WHERE COD_FILME = 1”)

‘Limpa os objetos ADO
objRS.Close
set objRS = Nothing

objConn.Close
set objConn = Nothing
%>
```

2.4 SOFTWARES SIMILARES AO APLICATIVO

Existem alguns softwares no mercado com o mesmo propósito do aplicativo desenvolvido, porém encontram-se algumas diferenças bem significativas.

O Universal Table Editor, desenvolvido por Wellige (2002), permite a edição do banco de dados de uma forma bem simples, neste software é escolhido a tabela e o sistema configura

as páginas necessárias para fazer a atualização da tabela. Na figura 5 temos a página de listagem de registro do software:

Figura 5 – Universal Table Editor

Já o Table Editor 0.8 Beta desenvolvido por 2eNetWorx (2002), é um software mais completo. Neste caso temos a possibilidade de alterar a estrutura da tabelas, criando novos campos, por exemplo. Na figura 6 temos a página de detalhes do registro criado pelo Table Editor da 2eNetWorx:

Figura 6 – Table Editor

Um grande diferencial do aplicativo apresentado neste trabalho quanto a estes dois softwares, é no trabalho é permitido a criação de relacionamentos entre tabelas, característica esta não encontrada nos outros softwares. Outro ponto positivo está na melhor criação dos forms em HTML que no aplicativo são criadas de acordo com o tamanho do campo.

3 DESENVOLVIMENTO DO TRABALHO

Neste capítulo apresenta-se a descrição do aplicativo com suas etapas de desenvolvimento.

3.1 REQUISITOS PRINCIPAIS DO TRABALHO

O sistema está dividido em duas partes distintas, as páginas de administração, que são responsáveis pela configuração do sistema e as páginas criadas, que serão utilizadas para a atualização do banco de dados.

Através das páginas de administração é possível configurar o sistema, indicando quais campos serão mostrados nas páginas criadas e qual o modelo de cores que será utilizado. Os três tipos de páginas que são criados pelo sistema: página de procura, página de listagem e a página de detalhes. Estas páginas são o resultado do aplicativo, ou seja, o que será obtido com a ferramenta proposta.

3.1.1 PÁGINA DE PROCURA

A página de procura é utilizada como um filtro para a tabela do banco de dados. Nesta página deve-se fornecer as informações que serão filtradas de acordo com cada campo da tabela.

Esta página está dividida em quatro partes distintas:

a) os campos principais fazem pesquisa somente em seu próprio campo correspondente. Por exemplo, o campo *Nome* da tabela *Usuário*, quando preenchido com algum valor filtrará todos os usuários que apresentem o *Nome* digitado;

b) os campos “palavra chaves” são vários campos onde serão feitas as pesquisas. Por exemplo, pode-se configurar como campos para pesquisa por palavra chave os campos *Nome* e *Endereço* e qualquer texto que for incluído neste campo do formulário fará a pesquisas nos campos *Nome* **ou** *Endereço*, retornando os registros que possuam o texto nestes campos. Neste caso deve haver a utilização do conectivo “ou”;

c) os campos definidos como chave estrangeira serão mostrados através de um *list box (HTML)*, e conterá os valores dos registros do campo chave relacionado.

d) campos de ordenação são os campos que podem permitir que o resultado mostrado na página de listagem esteja ordenado de acordo com o campo especificado.

Nesta página quando for preenchido ou escolhido mais de um campo no formulário será considerado o critério “AND”, ou seja, só serão mostrados os registros que possuem tanto um campo quanto o outro. Exemplificando, caso seja preenchido o campo *Nome* com a palavra “*Paulo*” e o campo chave estrangeira *Profissão* seja escolhido “*Programador*”, serão retornados todos os registros que possuam a palavra “*Paulo*” no campo *Nome* e (“AND”) a Profissão seja “*Programador*”.

A página de procura criada em *HTML/ASP* mostrará os campos do formulários em tamanhos e tipos de acordo com o tipo de campo. Ou seja, um campo do tipo texto será mostrado através da seguinte *tag* HTML, conforme quadro 9.

Quadro 9 – Exemplo de form em HTML

```
<input type="text" name="textfield" size="10" maxlength="15">
```

As opções *size* e *maxlength* da tag *INPUT* também serão dimensionadas de acordo com o tamanho do campo.

3.1.2 PÁGINA DE LISTAGEM

A página de listagem é o resultado obtido através da página de procura. Nesta página são listados todos os registros encontrados de acordo com o formulário preenchido na página de procura. Nesta página é utilizado o recurso de paginação, para facilitar a navegação caso o número de registros obtidos seja muito alto.

Clicando no nome do campo em cada coluna é possível reordenar a coluna de acordo com este campo. Clicando duas vezes no mesmo campo, inverte-se o sentido de ordenação.

Clicando no código do registro (primeiro campo no lado esquerdo da tabela) o usuário é redirecionado para a página de detalhes do registro, onde é possível visualizar todos os campos e, se for o caso, fazer as alterações no registro.

3.1.3 PÁGINA DE DETALHES

Nesta terceira página onde se tem os detalhes do registro, com todos os campos. Assim como ocorre na página de procura, os campos são mostrados em *tags HTML* de acordo com o tipo de campo.

Esta página apresentará as opções:

a) salvar – salva as alterações no registro;

b) excluir – apaga o registro da tabela, é feito a confirmação da exclusão;

c) novo – cria um novo registro, a página será recarregado com todos os campos do formulário em branco;

3.2 ESPECIFICAÇÃO

O aplicativo disponibiliza uma ferramenta para configuração e criação das páginas descritas anteriormente em *ASP* utilizando recursos de programação *ASP*.

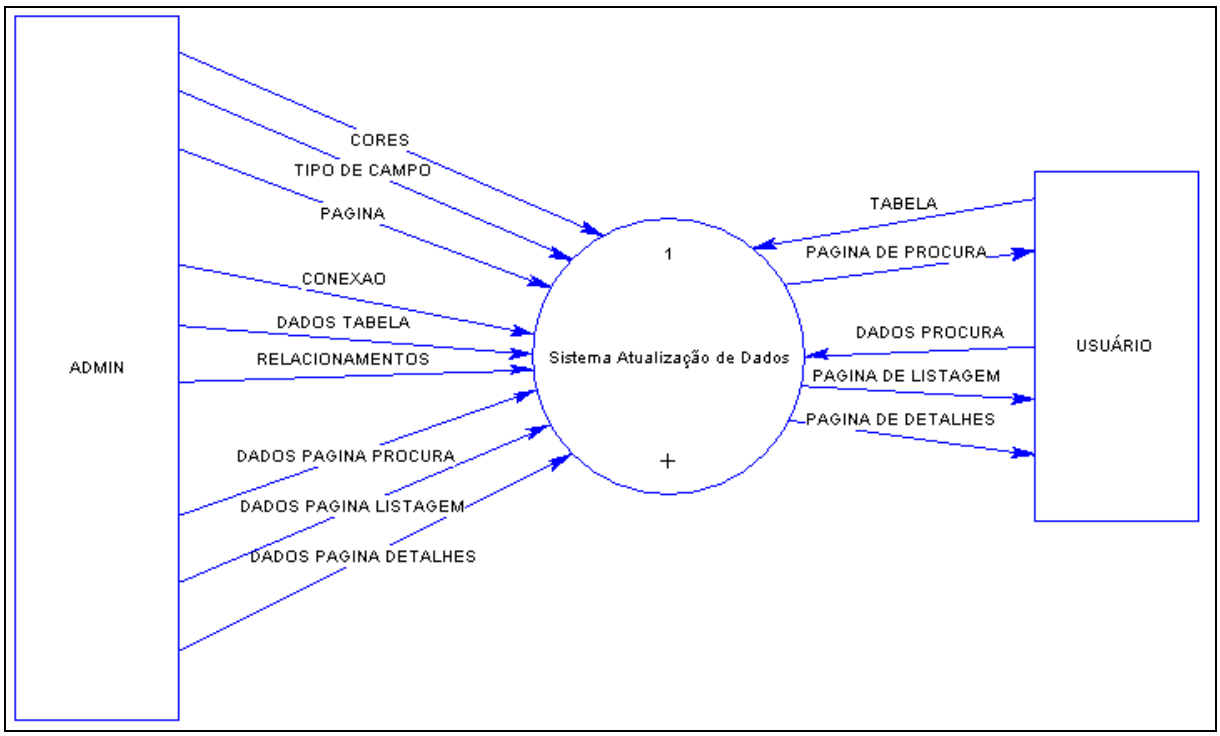
A metodologia utilizada no desenvolvimento da especificação do sistema foi a análise estruturada.

Para especificação do aplicativo foi utilizada a ferramenta Power Designer. Foram criados o diagrama de contexto e o modelo entidade relacionamento (*MER*) das tabelas da base de dados que será utilizada para configurar o sistema.

3.2.1 DIAGRAMA DE CONTEXTO

No diagrama de contexto (figura 7) se apresenta duas entidades externas envolvida no sistema de atualização de dados. Tem-se o administrador (Admin) que configura as páginas criadas e na outra entidade temos o usuário final, que irá interagir com o aplicativo fazendo a atualização do banco de dados.

Figura 7 - Diagrama de Contexto

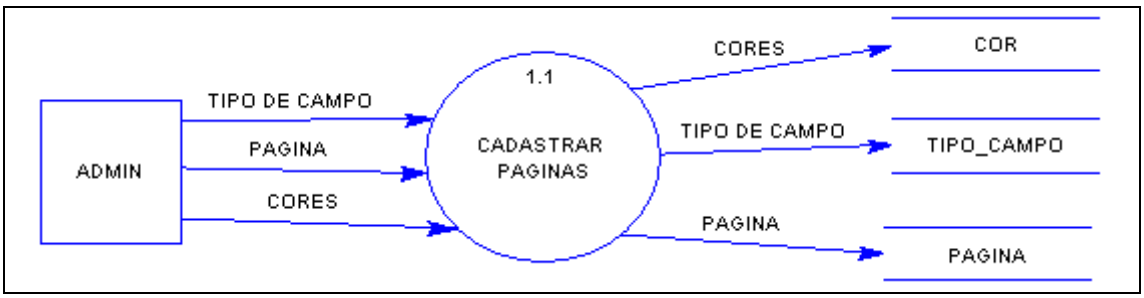


3.2.2 DIAGRAMAS DE FLUXO DE DADOS

Nos diagramas de fluxo de dados, pode-se observar as principais funcionalidades do sistema.

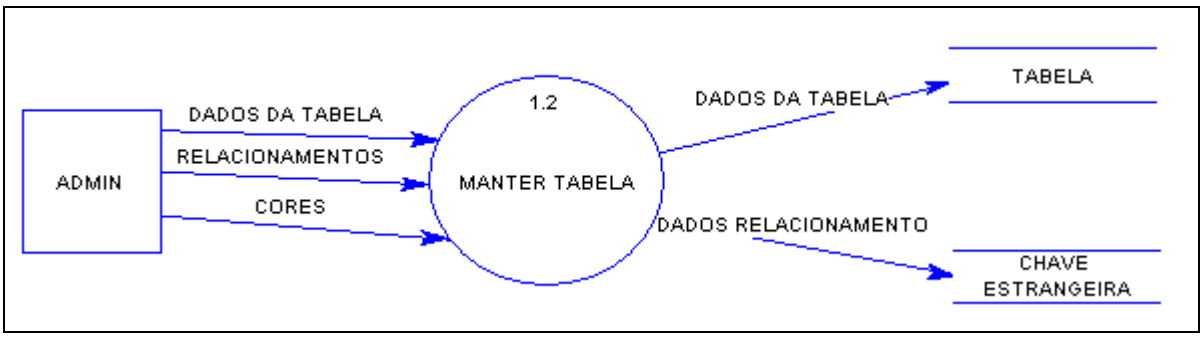
Na figura 8 tem-se a rotina para manter os dados das tabelas COR, TIPO_CAMPO E PAGINA, estes dados são cadastrados no sistema pelo administrador.

Figura 8 – Cadastrar páginas.



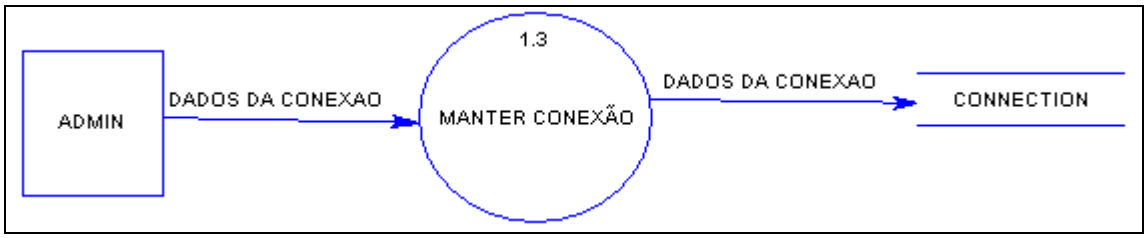
Na figura 9 tem-se a rotina para manter os dados das tabelas TABELA e CHAVE ESTRANGEIRA, estas são as principais informações do aplicativo que são salvas durante a configuração do sistema.

Figura 9 – Manter Tabela.



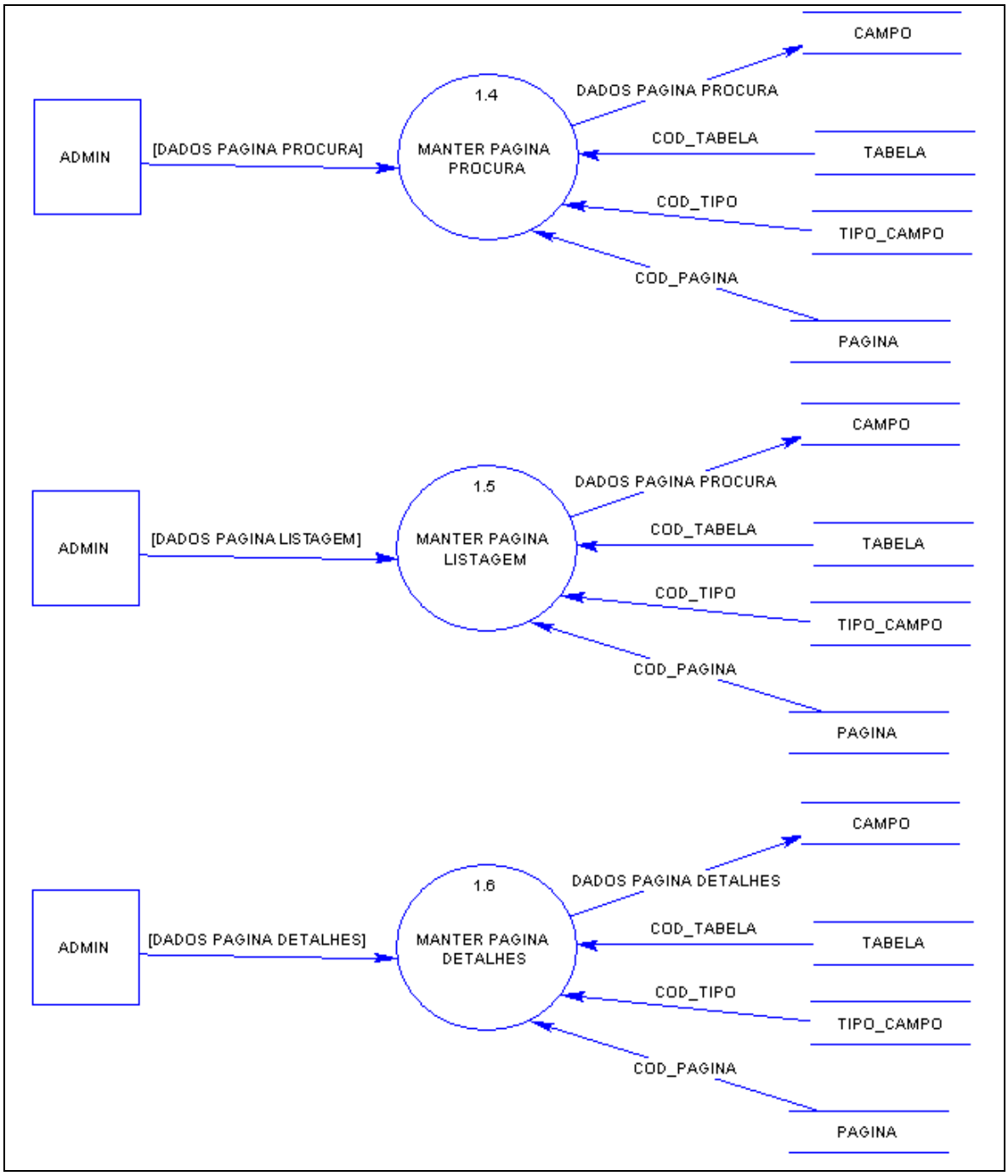
Na figura 10 tem-se a rotina para manter os dados das tabela CONNECTION, nesta tabela são salvos os dados da conexão com o banco de dados.

Figura 10 – Manter Conexão.



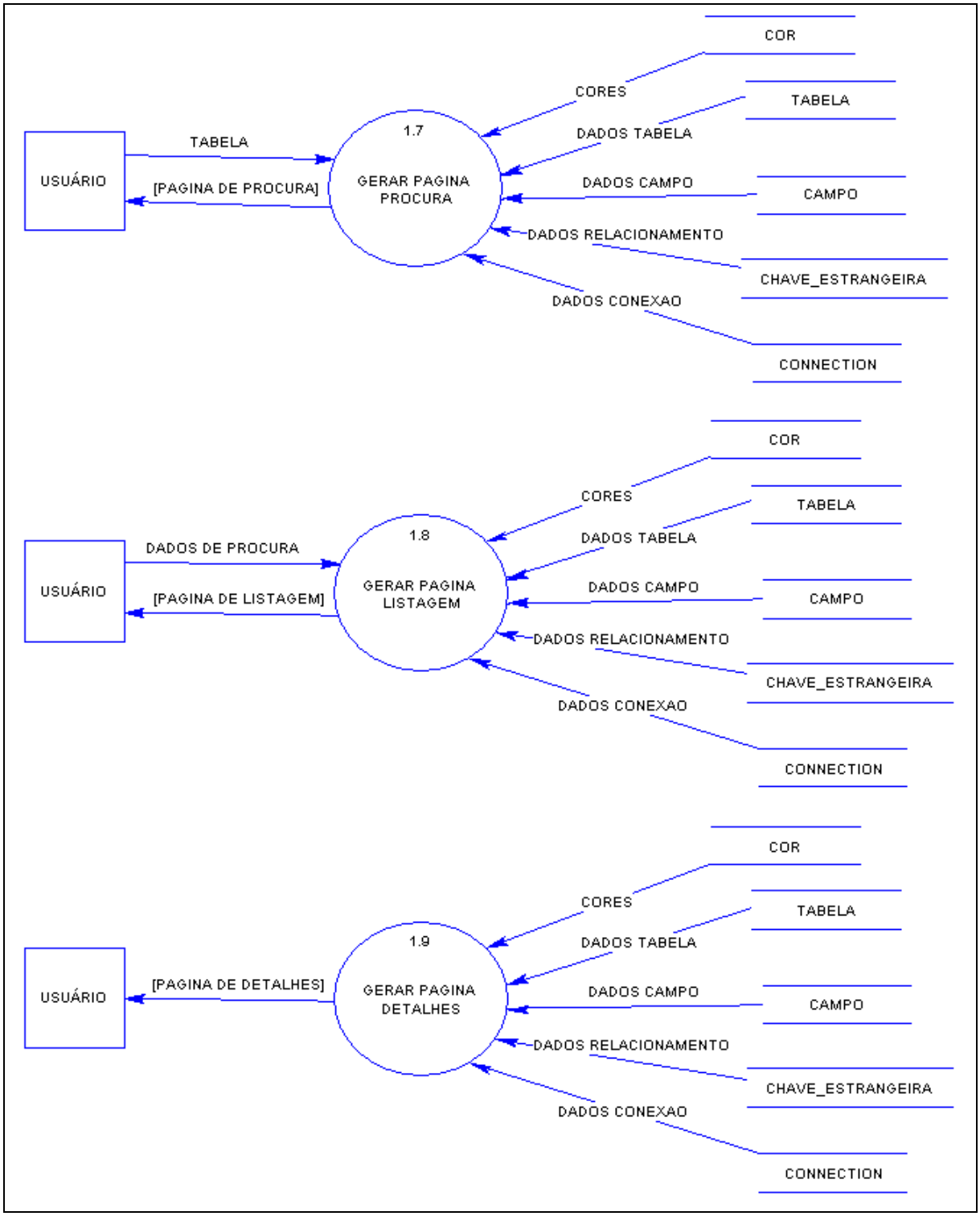
Na figura 11 tem-se as rotinas para manter página de procura, listagem e detalhes. O administrador passa os dados referentes a configuração da página, o sistema busca as informações necessárias em TABELA, TIPO_CAMPO e PAGINA, e salva os dados na tabela CAMPO.

Figura 11 – Manter páginas de procura, listagem e detalhes.



Na figura 12 tem-se as rotinas para criar páginas de procura, listagem e detalhes. Quando o usuário clica em visualizar, o sistema busca as informações necessárias para criar as páginas e retorna para o usuário as páginas criadas de acordo como foi configurado.

Figura 12 - Criar páginas de procura, listagem e detalhes

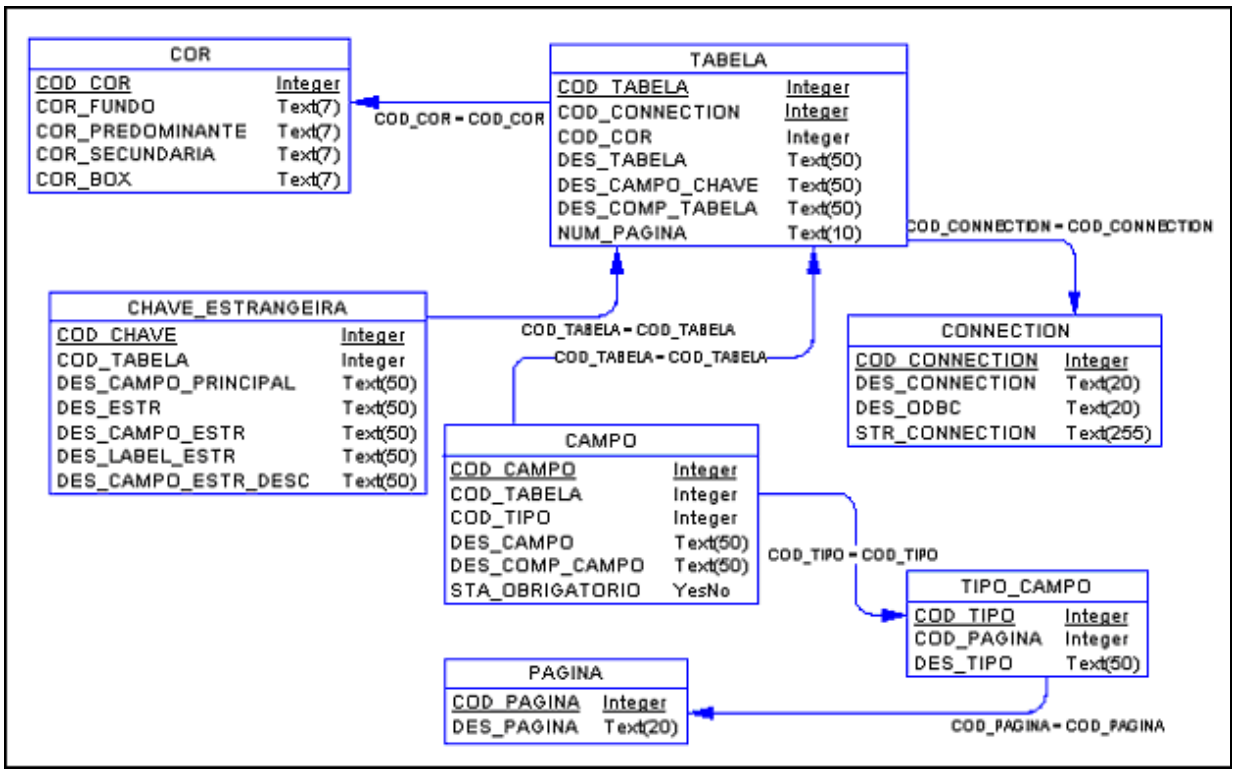


3.2.3 MODELO ENTIDADE RELACIONAMENTO

Na figura 13 apresenta-se o modelo entidade relacionamento (MER) das tabelas utilizadas no sistema.

Para a base de dados do sistema utilizou-se o *Microsoft Access* versão 2000.

Figura 13 - Modelo entidade relacionamento (MER)



A estrutura das tabelas da figura 13 é composta pelos seguintes módulos:

a) TABELA: esta é tabela principal do sistema, onde são armazenados os dados principais como nome e descrição da tabela, conexão com o banco de dados e também os campos relacionados com a chave estrangeira. Veremos a descrição dos campos:

- COD_TABELA – campo auto-incremento que é a chave principal da tabela;
- COD_CONNECTION – código da conexão com o banco de dados, relacionado com a tabela CONNECTION;
- COD_COR – relacionamento com a tabela de cores, que possui as cores que serão utilizadas nas páginas.
- DES_TABELA – descrição do nome da tabela. Este campo é escolhido através da listagem de tabelas encontradas na conexão estabelecida. Este campo só

poderá ser configurado após a escolha da conexão;

- DES_CAMPO_CHAVE – descrição do campo que é o campo chave da tabela;
- DES_COMP_TABELA – uma descrição mais completa do campo chave da tabela. Ex.: o DES_CAMPO_CHAVE pode ser *cod_usuario* já o campo DES_COMP_TABELA poderia ser *Código de Usuário* que será o campo que aparecerá na página criada;
- NUM_PAGINA – contém uma lista com o código dos tipos de página que o sistema permite. Por exemplo, com o conteúdo "1,2,3" se tem as três páginas: procura, listagem e detalhes, já "1,2" somente procura e listagem. As informações sobre os tipos de páginas estão na tabela PAGINA.

b) CONNECTION: nesta tabela são armazenados os tipos de conexões com a base de dados. Veja os detalhes de cada campo:
   - COD_CONNECTION – identificador da tabela;
   - DES_CONNECTION – descrição da conexão;
   - DES_ODBC – nome da conexão ODBC (optativo);
   - STR_CONNECTION – caso não seja feita uma conexão ODBC, pode-se configurar uma *string* de conexão neste campo.

c) CHAVE_ESTRANGEIRA: nesta tabela são armazenadas as informações referentes as chaves estrangeiras da tabela, como é o relacionamento entre as tabelas e descrição de campos. Os detalhes de cada campo:
   - COD_CHAVE – identifica a chave estrangeira;
   - COD_TABELA – relacionamento com a TABELA.
   - DES_CAMPO_PRINCIPAL – nome do campo na tabela principal, que possui o relacionamento com a tabela estrangeira, definida no campo abaixo;
   - DES_ESTR – nome da tabela onde se encontra a chave estrangeira;
   - DES_CAMPO_ESTR – nome do campo na tabela estrangeira;
   - DES_LABEL_ESTR – descrição que aparecerá para o usuário referente ao campo da chave estrangeira;
   - DES_CAMPO_ESTR_DESC – nome do campo que será mostrado para o usuário.

   Para exemplificar a utilização desta tabela, apresenta-se um exemplo:

   Tabela principal: USUARIO, campos: cod_usuario, nome, cod_profissao e tabela com chave estrangeira: PROFISSAO, campos: cod_profissao, profissao

   Com a configuração acima se tem os seguintes possíveis valores na tabela

CHAVE_ESTRANGEIRA:

```
COD_CHAVE = 1
COD_TABELA = 2
DES_CAMPO_PRINCIPAL = cod_profissao
DES_ESTR = profissao
DES_CAMPO_ESTR = cod_profissao
DES_LABEL_ESTR = Profissão
DES_CAMPO_ESTR_DESC = profissao
```

d) CAMPO: esta tabela é usada para salvar as informações referentes aos campos nos diferentes tipos de páginas. Contém os seguintes campos:
   - COD_CAMPO – identifica o campo;
   - COD_TABELA – relacionamento com TABELA;
   - COD_TIPO – relacionamento com a tabela TIPO_CAMPO;
   - DES_CAMPO – nome do campo;
   - DES_COMP_CAMPO – descrição do campo que será mostrado para o usuário final.
   - STA_OBRIGATÓRIO – define se o campo é obrigatório ou não.

e) TIPO_CAMPO: salva os diferentes tipos de campos encontrados em cada página. Descrição dos Campos:
   - COD_TIPO – identificador da tabela;
   - COD_PAGINA – relacionamento com o tipo de página;
   - DES_TIPO – uma breve descrição do tipo de campo.

   Para a versão atual do aplicativo esta tabela possui os valores fixos, com os seguintes registros (exceto a coluna comentário), conforme quadro 10.

Quadro 10 – Valores da tabela TIPO_CAMPO

COD_TIPO	DES_TIPO	COD_PAGINA	COMENTÁRIO
1	Formulário	1	Campo utilizado nos formulários da página de procura
2	Palavra Chave	1	Campo utilizado no formulário na opção Palavra Chave
3	Ordenação	1	Também utilizado na página de procura, na opção ordenação.
4	Colunas	2	Utilizado na página de listagem, onde será armazenado as colunas que serão mostradas na página.
5	Campos detalhes	3	Utilizado na página de detalhes, define os campos da tabela.

f) PAGINA – possui a descrição das páginas. Também possui os registros pré-definidos para esta versão:

COD_PAGINA	DES_PAGINA
1	Procura
2	Listagem
3	Detalhes

3.3 IMPLEMENTAÇÃO

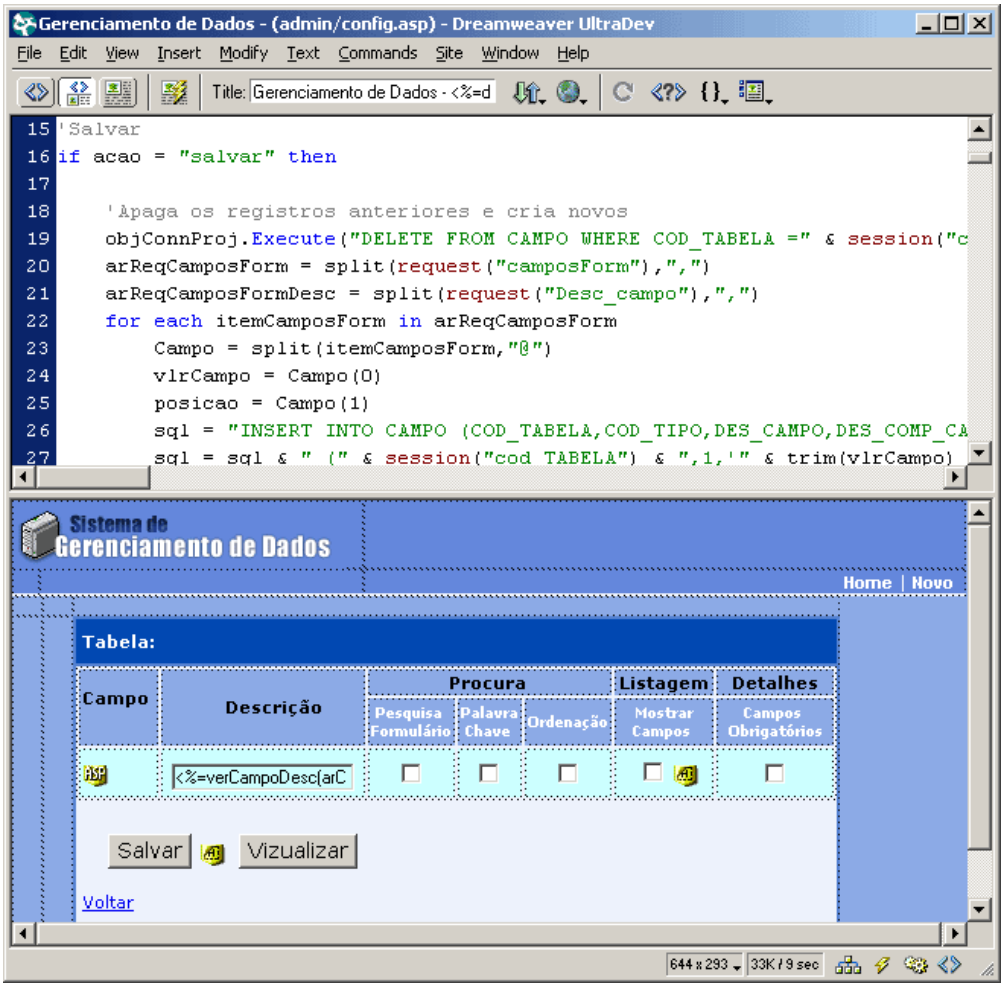
3.3.1 TÉCNICAS E FERRAMENTAS UTILIZADAS

Para implementação do aplicativo foram utilizadas as ferramentas *Macromedia Ultradev 4.0*, servidor *IIS* do *Windows 2000* e as linguagens de programação *ASP, HTML e Javascript*. Todos os testes e desenvolvimento do sistema foram feitos utilizando o *Microsoft Internet Explorer 5*.

O servidor *IIS* do *Windows 2000* foi utilizado como servidor *web* para executar os *scripts ASP*.

O *Macromedia Ultradev 4.0* é uma excelente ferramenta para a edição de código *ASP* e *HTML*, pois facilita a visualização do código (figura 14 – parte superior) e mostra em modo gráfico como que o *HTML* será visualizado no Navegador *web* (figura 13 – parte inferior).

Figura 14 – Macromedia Ultradev 4.0

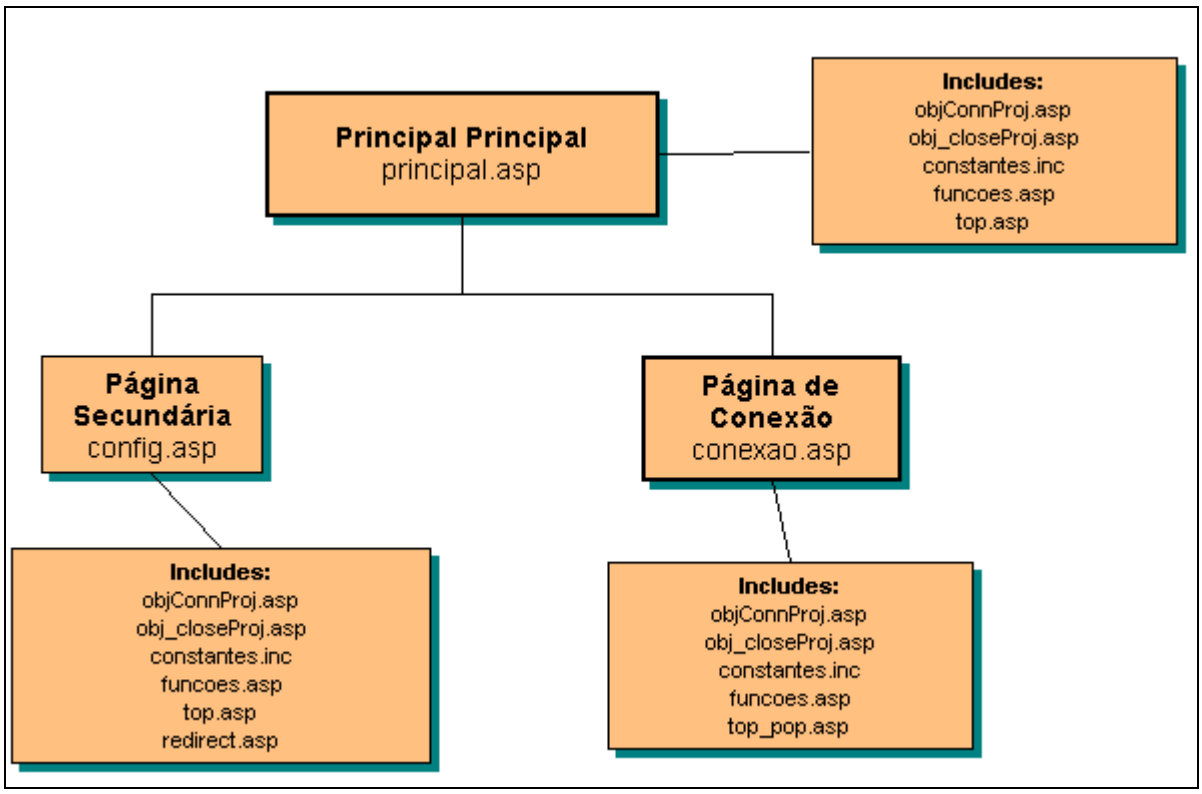


3.3.2 OPERACIONALIDADE DA IMPLEMENTAÇÃO

Este item trata da operacionalidade da implementação e como estão organizados os arquivos.

Na figura 15 é mostrado o macro fluxo do sistema.

Figura 15 – Macro fluxo do Sistema



A página principal é a página de configuração onde o usuário entrará com os parâmetros principais para configurar o sistema.

Na página de conexão é possível criar uma nova conexão com base de dados. Essa página é acessível clicando no botão “Nova” encontrado ao lado do campo Conexão na página principal (figura 16).

Clicando no botão *“continuar”* da página principal o usuário é direcionado para a página secundária de configuração dos detalhes de cada página criada.

Os *includes* são utilizados para compartilhar o mesmo segmento de código por várias páginas. Os principais *includes* utilizados são os seguintes:

a) objConnProj.asp – abre a conexão com a base de dados onde serão armazenados os dados do projeto;

b) obj_CloseProj.asp – fecha a conexão com a base de dados;

c) constantes.inc – constantes ADO. São as constantes utilizadas como parâmetros nos objetos ADO: *recordset e conection*;

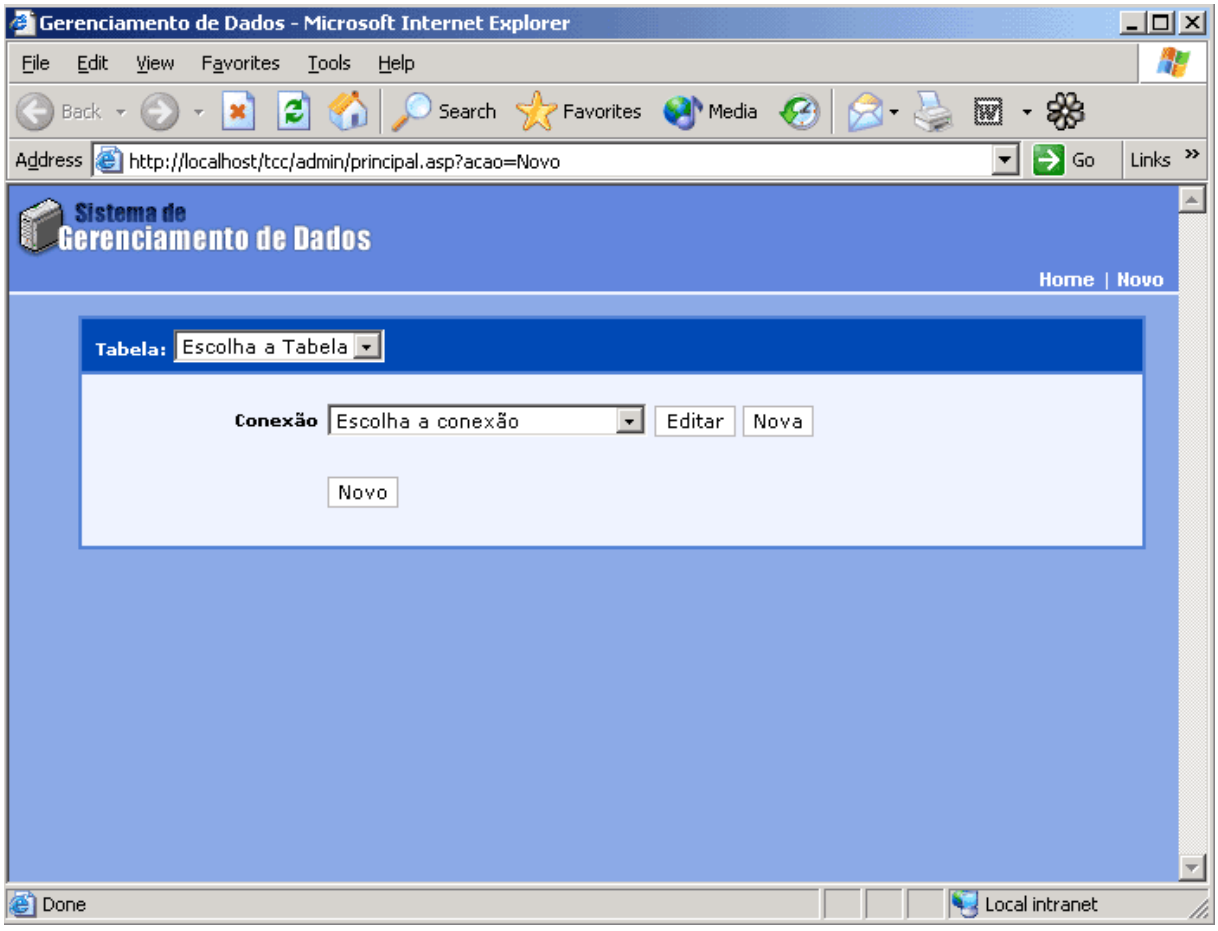
d) funcoes.asp – conjuntos de funções comuns que são compartilhada pelas páginas *ASP*;

e) top.asp – cabeçalho utilizado nas páginas.

3.4 APRESENTAÇÃO DO APLICATIVO

Este item apresenta a visualização do aplicativo desenvolvido, com uma seqüência coerente de utilização do mesmo passo a passo.

Figura 16 - Tela Principal (etapa nº 1)



Na primeira vez que o sistema é acessado é mostrada a tela da figura 16, nesta tela tem as seguintes opções:

a) tabela na barra superior – escolher uma tabela já cadastrada no sistema. Após a escolha automaticamente a página é recarregada mostrando os dados já cadastrados na tabela escolhida;

b) conexão – Pode-se escolher a conexão com a base de dados. O *list box* mostra as conexões já cadastradas no sistema. Após a escolha, a página e recarregada e é mostrada a tela da figura 15;

c) botão *Nova (ao lado da conexão)* – clicando no botão *nova* é aberto um popup onde o usuário poderá cadastrar uma conexão (figura 17);

d) botão *Editar* – clicando no botão *editar* é aberto um popup para que o usuário possa modificar a conexão escolhida no *list box* já cadastrada.

Figura 17 - Tela de Conexão

Na tela mostrada na figura 17 é possível cadastrar novas conexões com banco de dados. Esta tela tem as seguintes opções:

a) descrição – um nome para a conexão;

b) ODBC – nome da conexão ODBC (opcional);

c) *string* de conexão – define um *string* para a conexão com o banco de dados (opcional).

Para o exemplo utilizá-se o campo *Descrição* com o nome “TCC”, e a *string* de conexão conforme quadro 11:

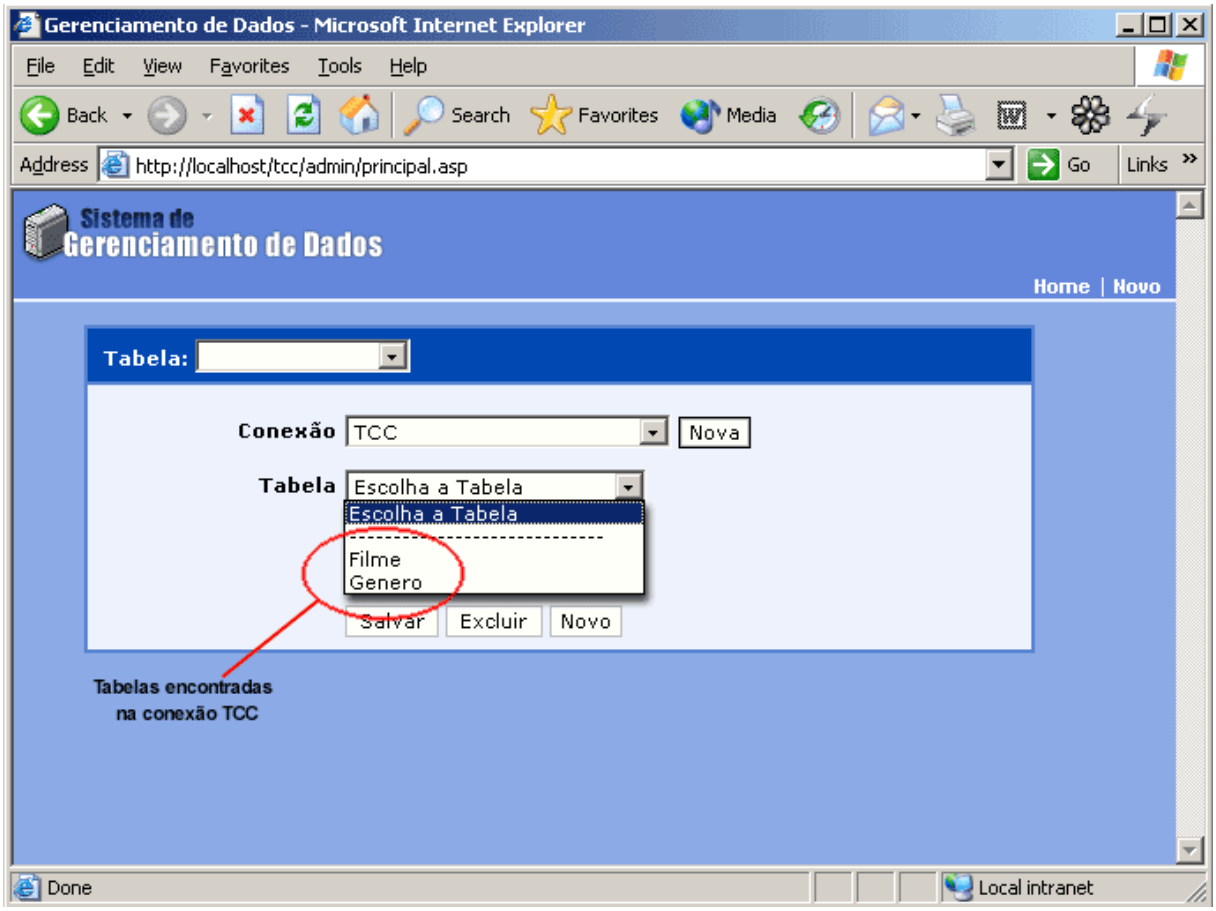
Quadro 11 – Conexão com Banco de Dados

```
    Driver=Microsoft Access Driver (*.mdb);
DBQ=d:\sites\tcc\dados\tcc.mdb
```

O campo *ODBC* não será utilizado.

Após configurar a conexão correta pode-se fechar a janela do cadastro de conexão (figura 16). Em seguida deve-se escolher uma conexão na tela principal, após a escolha da conexão é mostrada a tela da figura 18.

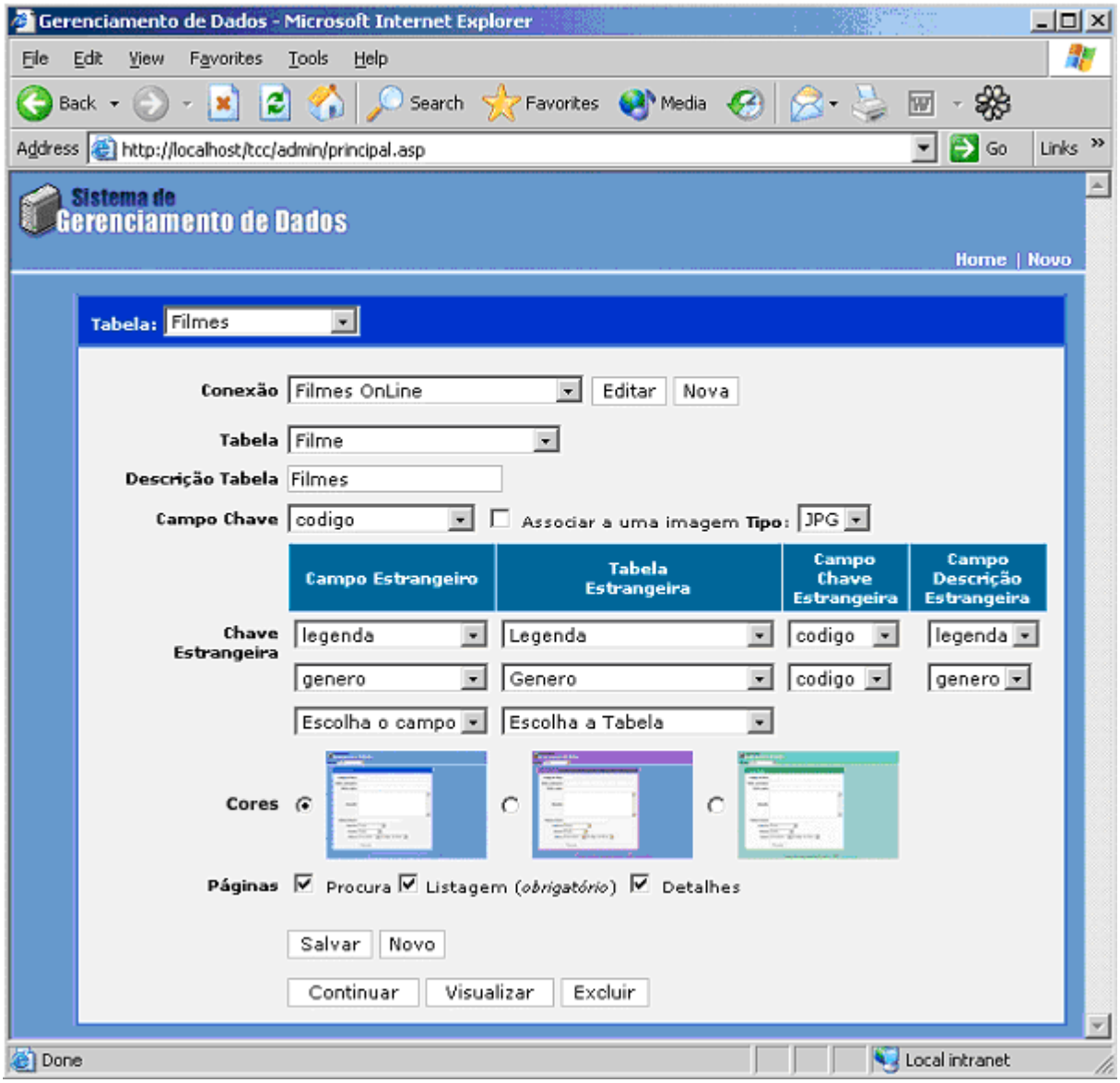
Figura 18 - Tela Principal (etapa nº 2)



Nesta tela tem-se um novo campo Tabela onde no *list box* é mostrado uma listagem de todas as tabelas encontradas na conexão previamente escolhida. No exemplo em questão são mostrados nas tabelas *Filme* e *Genero* que são as tabelas cadastradas no arquivo *tcc.mdb.*

Após a escolha da tabela a página é novamente recarregada é mostrada a tela conforme a figura 19.

Figura 19 – Tela Principal (etapa nº 3)



Tem-se a seguinte descrição dos campos relativo a figura 19:

a) descrição tabela – um nome para a tabela;

b) campo chave – o campo que é a chave principal da tabela, o seu identificador;

c) chave estrangeira – esta opção por sua vez está dividido em quatro campos:

1. Campo Estrangeiro – campo na tabela principal que possui a chave estrangeira;

2. Tabela Estrangeira – tabela que possui o relacionamento com o campo da opção anterior;

3. Campo chave estrangeira – campo na tabela chave estrangeira que está relacionado com a tabela principal;

4. Campo descrição estrangeira – campo que possui a descrição do campo da chave estrangeira.

   Estas últimas duas opções só serão mostradas após a escolha da tabela estrangeira, pois os campos mostrados estão relacionados com esta escolha;

d) layout – os modelos de layout disponíveis no sistema;

e) páginas – são as páginas que o sistema irá criar.

Após salvar as alterações pode-se clicar no botão *Continuar* para se fazer a configuração das páginas.

Na tela mostrada na figura 20 é feita a configuração das páginas criadas pelo sistema. Esta tela está organizada em linhas e colunas. Nas linhas tem-se os vários campos e as opções para cada página (colunas). Algumas considerações sobre esta tela:

a) descrição – nesta coluna é possível alterar o nome do campo para um nome mais completo que será mostrado para o usuário final. Quando a página é carregada pela primeira vez são assumidos como valores *default* os nomes dos campos;

b) pesquisa formulário (Procura) – nesta coluna devem ser assinaladas todas as opções de procura que devem aparecer no formulário da página de procura;

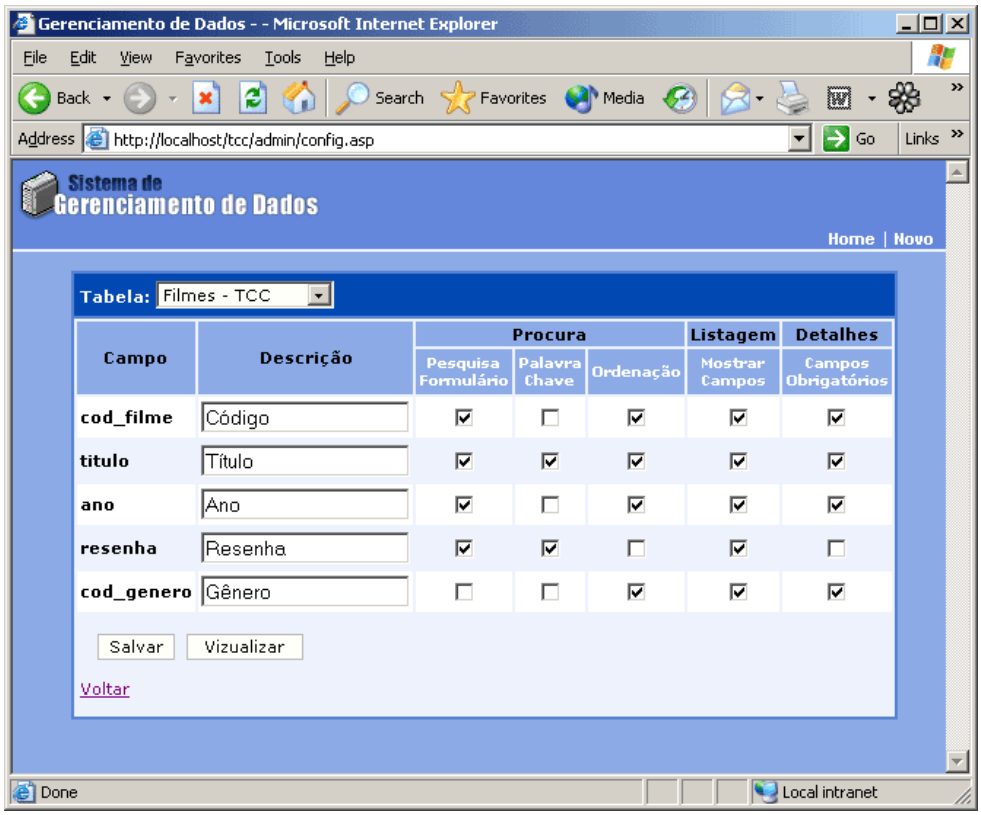
c) palavra chave (Procura) – são os campos que serão pesquisados quando o campo “Palavra chave” for preenchido no formulário de procura;

d) ordenação (Procura) – são os campos que aparecerão no *list box* de ordenação;

e) listagem – todos as campos que serão mostrados nas colunas da página de listagem;

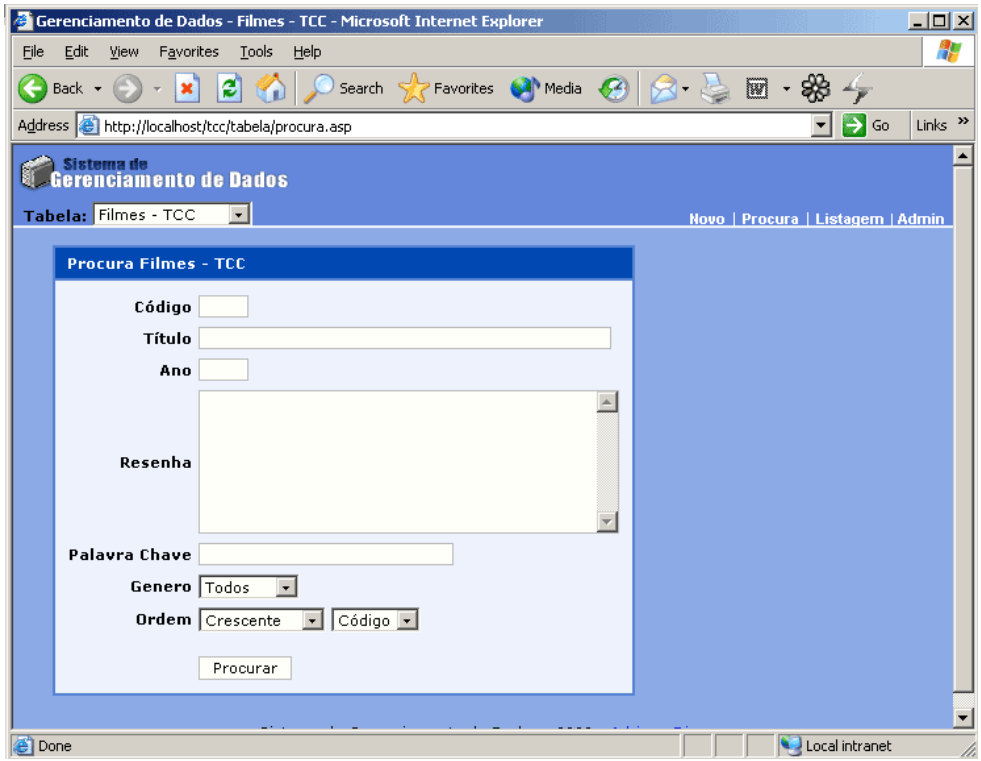
f) detalhes – os campos que são obrigatórios o preenchimento na página de detalhes.

Figura 20 – Página de configuração



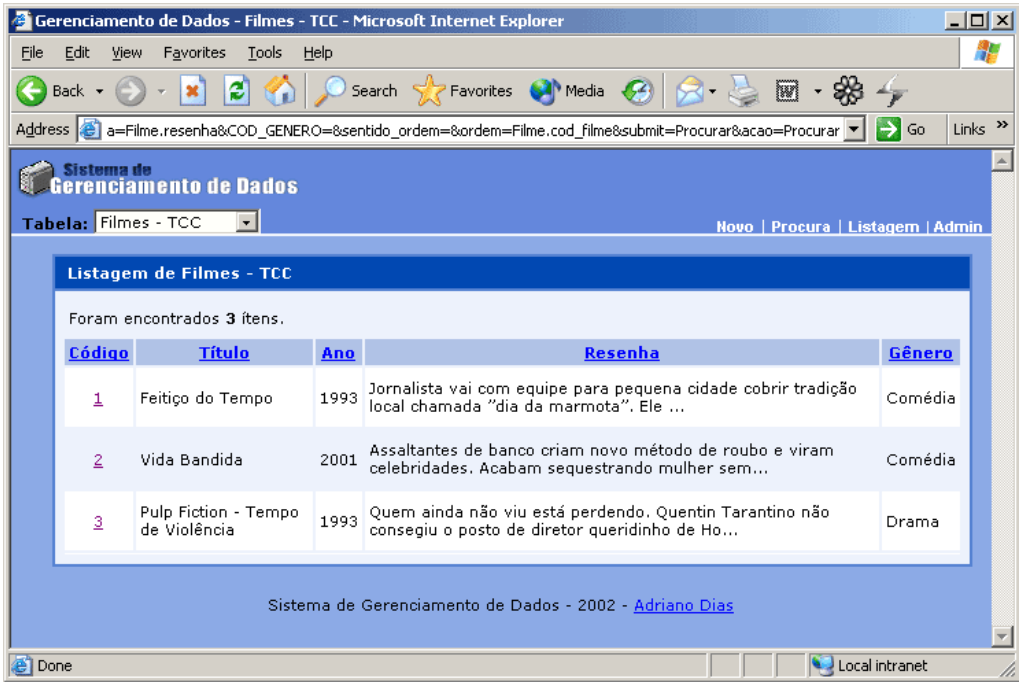
Clicando em visualizar é possível navegar pelo sistema criado. A primeira tela que é mostrada é a página de procura (caso tenha sido assinalada na página principal de configuração, senão será mostrada a listagem de registros). Esta tela é mostrada na figura 21.

Figura 21 – Página de Procura



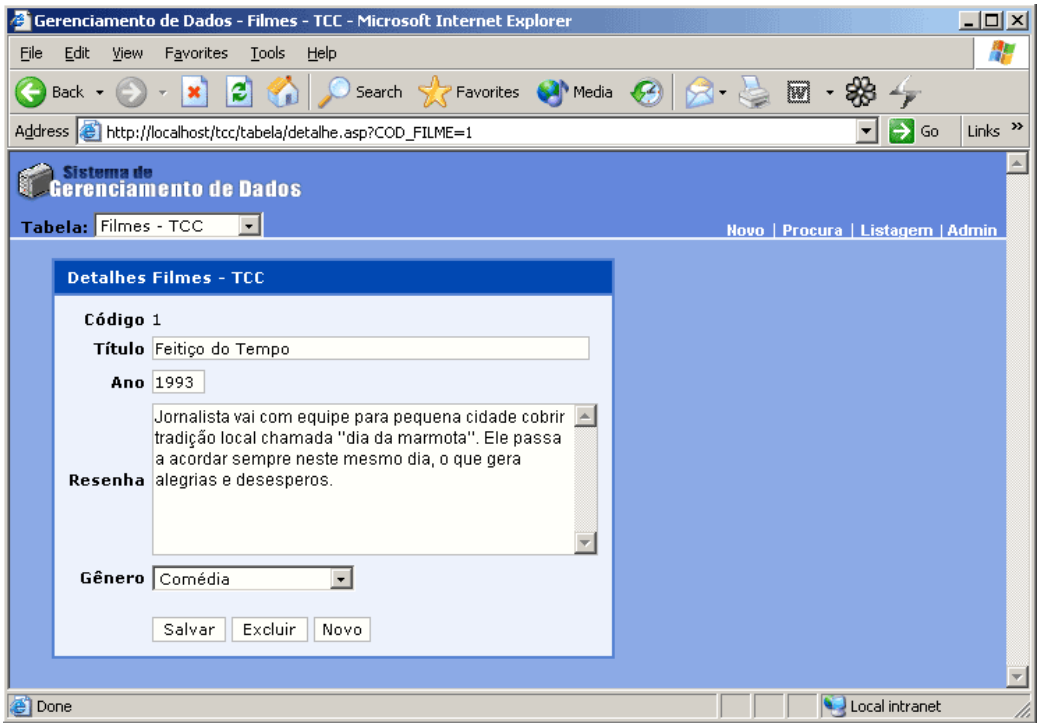
Em seguida clicando-se em procurar, será mostrada a listagem dos registros encontrados, conforme a figura 22.

Figura 22 – Página de listagem



E finalmente pode-se clicar no *link* do código do registro e fazer a edição dos dados conforme a figura 23.

Figura 23 – Página de detalhes



3.5 TESTES COM BANCOS DE DADOS

A demonstração de funcionalidade do aplicativo no item anterior foi feita utilizando o Microsoft Access, também se realizou testes utilizando-se o Microsoft SQL Server e o Oracle 8.0. Nos três casos o aplicativo criou as páginas normalmente, bastando para isto que fosse estabelecida a conexão com o banco corretamente. Pequenas diferenças entre tipos de campos foram tratadas no aplicativo para que funcionassem com estes bancos de dados, já que as constantes *ADO DataTypeEnum* que são retornadas como propriedades do campo *field* do *recordset* diferem em alguns casos de acordo com o tipo de campo e do banco utilizado.

4 CONCLUSÕES

Pode-se considerar de grande valia o estudo sobre a tecnologia ASP e o conhecimento sobre ferramentas que auxiliam na geração de código e gerenciamento de conteúdo. Acredita-se, segundo a literatura e alguns autores, que o ASP é uma das tecnologias mais eficientes e de fácil aprendizado e vem crescendo em número de aplicações para *web*. Através do aplicativo apresentado criou-se um ambiente de fácil utilização, onde os usuários podem configurar páginas para realizar a atualização de tabelas nos bancos de dados.

O aplicativo permite a criação das páginas necessárias para atualizar o conteúdo de tabelas: pesquisar, editar, inserir e excluir registros. Todas as funções são compatíveis com os bancos de dados citados e com os diferentes tipos de tabelas e campos.

O aplicativo permite a configuração das páginas criadas, onde o usuário indica quais campos devem ser mostrados e características das páginas, bem como a escolha das cores mais adequadas.

Quanto à tecnologia envolvida, acredita-se que este tipo de aplicação passará a fazer parte do dia-a-dia dos programadores, uma vez que cada vez mais se busca a construção de aplicações que possam gerar código a partir de dados parametrizados.

Em relação às ferramentas utilizadas, o Power Design, mostrou-se eficiente no suporte à metodologia estruturada. O Macromedia Ultradev 4.0 facilitou a edição de códigos ASP, HTML e Javascript, pois apresenta várias *tags* HTML visuais. Com relação ao servidor IIS do Windows 2000, o mesmo apresentou estabilidade na execução de *scripts*, tornando sua utilização eficiente para esse tipo de aplicação.

4.1 LIMITAÇÕES

Como limitações têm-se os seguintes itens:

a) a exclusão de registros é permitida somente na página de edição do registro;
b) na procura não é possível configurar faixas de pesquisa, por exemplo, definir uma data inicial e uma data final para filtrar na pesquisa;
c) o aplicativo trabalha com tabelas que possuam somente uma chave primária.

4.2 SUGESTÕES

Como sugestões para trabalho futuros, tem-se as seguintes possibilidades:

a) permitir a edição do registro também na listagem;
b) permitir o *upload* de imagens associadas ao registro da tabela;
c) implementar a confirmação na exclusão;
d) criar um sistema de usuários através de *login* e senha, onde cada usuário teria privilégios diferentes.

REFERÊNCIAS BIBLIOGRÁFICAS

2ENETWORX. [2002?]. Disponível em: < http://www.2enetworx.com >. Acesso em: 20 mar. 2002.

HOMER, Alex. **Professional Active Server Pages 3.0.** São Paulo**:** Wrox, 2000

MICROSFOT INC. **Microsoft Corporation,** São Paulo [2002?]. Disponível em: <http:// http://lg.msn.com/intl/br/ >. Acesso em: 15 mar. 2002.

MITCHELL, Scott; Atkinson, James. **Aprenda em 21 dias ASP 3.0 (Active Server Pages).** São Paulo: Campus, 2000.

PFAFFENBERGER, Bryan. **Estratégias de Extranet.** São Paulo: Editora Berkeley, 2000.

RUSSELL, Jones A. **Dominando ASP 3 ACTIVE SERVER PAGE 3 "A BÍBLIA".** São Paulo: MAKRON Books, 2001.

SAVOLA, Tom. **Usando HTML: guia de referência mais completo.** Rio de Janeiro: Campus, 1996.

WEISSINGER, Keyton A. **ASP guia completo.** São Paulo: Ciência Moderna, 1999.

WELLIGE, TOM. Universal Table Editor. [2002?]. Disponível em: < http://www.codeproject.com/asp/ute.asp >. Acesso em: 20 mar. 2002.

WILLE, Cristoph. **Aprenda em 24 horas Active Server Pages.** Rio de Janeiro: Campos, 1999.