UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

UM ANALISADOR SINTÁTICO PARA LOCALIZAÇÃO DE TEXTOS EM SÍTIOS WEB

RELATÓRIO DE PROJETO ORIENTADO EM COMPUTAÇÃO I

Lorena Coelho Vivian
Orientadora: Mariza Andrade da Silva Bigonha

18 DE JUNHO DE 2001

1 - INTRODUÇÃO

Atualmente a Internet é um meio global de comunicação e busca de informações. A Internacionalização de sites se torna uma necessidade básica para a transmissão do conteúdo de um sítio para outras comunidades que não utilizam-se da língua padrão do mesmo.

O objetivo de qualquer companhia que usa a Web como um veículo para vendas é alcançar todos os clientes em potencial globalmente. Este objetivo não pode ser atingido se muitos dos clientes não recebem uma língua que eles entendam. Além disso, mostrar respeito cultural distingüe uma companhia das outras.

Mas traduzir um sítio pode se tornar muito difícil quando se tem uma infinidade de arquivos a serem traduzidos. Arquivos .asp ou .jsp contém codificação interna em diversas linguagens de programação diferentes e precisam ser totalmente analizados até que se identifique o texto que deverá ser traduzido propriamente. Neste projeto serão analisados apenas os arquivos ASP(Active Server Pages).

Uma ferramenta para a localização e extração desses textos é de muita utilidade, principalmente se tratando de um sítio contendo dezenas, centenas ou até milhares de arquivos.

Uma pesquisa intensa na Web foi feita e nada foi encontrado sobre alguma implementação já feita nesse sentido que esteja disponível à comunidade.

2 – REVISÃO

JavaScript é uma linguagem de script projetada pelo Nestcape Communications para programação de páginas Web. É similar em sintaxe ao Java. O código de JavaScript é embutido em arquivos HTML e é interpretado pelo navegador do cliente. Com ele é possível uma maior interação para as páginas web.

Visual Basic Scripting or VBScript é um subconjunto do Microsoft Visual Basic. A edição em script é usada em documentos HTML e pode ser

executada no cliente(navegador) usando *tags* de *script* padrão, ou no servidor usando um tipo especial de documento HTML chamado Active Server Page (ASP).

Ambos VBScript e JavaScript podem ser usados em documentos HTML e ASP.

Códigos script entre os delimitadores <!-- e --> só são executados se o navegador reconhecer scripts, caso contrário eles são considerados comentário.

2.1 JavaScript

JavaScript é uma linguagem de programação simples que pode ser escrita diretamente dentro de documentos HTML para permitir o crescimento da interatividade com o usuário. Usando JavaScript é possível personalisar documentos HTML na hora, validar data e hora no lado do cliente e executar outras computações no lado do cliente.Por exemplo, JavaScript pode ser usado para criar calculadoras online ou mostras a data e hora atual.

A sintaxe de JavaScript é descrita a seguir.

	JavaScript
Array	New Array() new Object()
Atribuição	=
Comentários	// /* comment */

Fluxo de Controle	do {
	statements
	} while (condition)
	
	for (expression; condition; [expression])
	{
	statements
	}
	
	for (variable in object) {
	statements
	}
	
	while (condition) {

```
    statements
}

if (condition) {
    statements1
}
else {
    statements2
}

switch (expression){
    case label :
        statement;
        break;
    case label :
        statement;
       break;
    ...
    default : statement;
}
```

Captura de Erro	Window.onError = handler-func window.onError(message,url,line)
Literais	NaN null true, false User Defined Literals(e.g., 123.456, "test")
Operadores	Arithmetic +, ++, -, --, *, / % String + += Logical && || ! Bitwise & ^ | ~ << >> >>>> Assignment = += -= *= /= %= &= ^= |= <<= >>= >>>= Comparison == |= > >= < <= Special ?: , delete new this typeof void
Opções	N/A
Declaração de Procedimentos	Function function-name(arg1, arg2, ... argN)
Chamada de Procedimentos	Function function-name(arg1, arg2, ... argN) { statements; }
Saída de Procedimentos	
Parametros para Procedimentos	
Variáveis locais	Var variable-name
Variáveis Globais	Var variable-name

2.2 VBScript

VBScript é um subconjunto da linguagem Visual Basic, porém bem menor e

mais simples. Apesar de com JavaScript ser possível fazer tudo que é possível com VBScritpt e ainda mais, VBScript é muito usado por ser muito mais simples que JavaScript.

A sintaxe de VBScript é descrita a seguir.

	VBScript
Array	Declaration (e.g., Dim, Static)
	Lbound, Ubound
	ReDim, Erase
Atribuição	=
Comentários	REM
	Single Quote (')
Fluxo de Controle	Do ....Loop
	For ...Next, For Each ...Next
	While ...Wend
	If ... Then ...Else
Captura de Erro	On Error
	Err Object
Literais	Empty
	Nothing
	Null
	True, False
	User Defined Literals(e.g., 123.456, "test")
Operadores	Arithmetic +, -, *, /, \, ^, Mod, Negation (-)
	Strings concatenation (&)
	Comparison =, <, >, <>, <=, >=, Is
	Logical Not, And, Or, Xor, Eqv, Imp
Opções	Option Explicit
Declaração de Procedimentos	Functions
	Sub
Chamada de Procedimentos	Call
Saída de Procedimentos	Exit Function
	Exit Sub
Parametros para Procedimentos	ByVal, ByRef
Variáveis locais	Dim
	Static
Variáveis Globais	Private
	dim

Arrays fixos:

Um array de uma dimensão é uma maneira de agrupar itens de modo que pode-se acessar cada item através de seu índice:

Ex. : Dim MyButton(4)

De modo que este array possui 5 elementos.
Arrays dinâmicos:
Em arrays dinâmicos, o índice ainda não é conhecido. Portanto o array é declarado sem um número fixo de itens, e então é redimensionado quanto seu tamanho é conhecido.

```
Ex.: Dim MyButtons()
NumberOfButtons = InputBox("How many Buttons")
ReDim MyButtons(NumberOfButtons)
```

2.3 ASP

ASP ou Active Server Pages é uma tecnologia que funciona no lado do servidor, ou seja, pode ser executado em qualquer navegador web, porque todo o seu trabalho é feito no servidor web.

Essencialmente páginas ASP são apenas documentos HTML normais com scritps embutidos neles. Esses scripts podem ser escritos em VBScripts, JavaScript ou qualquer linguagem compatível com ActiveScript.
O começo de de um script é indicado com o símbolo <%, e o fim do script com um %>. Por exemplo:

```
<% for a = 1 to 5 %>
        <font size= <% = a %> > Hello World </font> <br>
<% next %>
```

Será executado pelo servidor antes da página ser enviada para o navegador. O código em <%...%> é executado e resulta no seguinte código html:

```
<font size= 1 > Hello World </font> <br>
<font size= 2 > Hello World </font> <br>
<font size= 3 > Hello World </font> <br>
<font size= 4 > Hello World </font> <br>
<font size= 5 > Hello World </font> <br>
```

A vantagem real de se usar ASP é a abilidade de se usar banco de dados ODBC com as páginas web.

Objetos ASP básicos

Usando VBScript, tem-se acesso a todos os objetos VB script padrão, qualquer DLL ActiveX que pode ser feita no VB5 - e os objetos ASP padrão. A seguir estão listados os objetos ASP básicos:

Application:

Permite que você compartilhe informações por todo seu aplicativo, utilizando variáveis globais.

Ex.: Applicatio(“strConnection”) = strConn
Neste caso a váriavel “strConnection” pode ser acessada por todos os arquivos num mesmo domínio.

Request:

Nos permite retornar dados do browser.

Exemplos:

request.querystring(”key”)
- Permite que se interrogue a linha de comando da.

reques.form(“inputname”)
- Para retornar informação de um formulário.

request.cookies(“key”)(“data”)=”string”
- Para retornar cookies (dados armazenados na máquina do cliente).

Collections: Cookies, Form, QueryString, ServerVariable.

Response:

Retorna dados da aplicação para o cliente. Permite que se escreva no arquivo HTML

Exemplos (Similares aos do Request) :

Response.write "string"
- para escrever “string” no arquivo HTML resultante.

response.cookies("adv")("data")="string".
  - Para escrever um cookie.

Server:

A objeto server possui métodos e propriedades para acessar recuros do servidor.

Ex.: set conn=Server.CreateObject("ADODB.Connection")

Session

Semelhante ao componente “Application”.

Abrindo um banco de dados

Primeiramente é criado um banco de dados no Microsoft Access, ou qualquer banco de dados que provê um driver ODBC. Entao no Painel de Controle adiciona-se um outro arquivo de sistema criando-se um nome para referenciar o banco de dados criado no Access. O nome que foi dado ao banco de dados é usado para encontrá-lo quando um fonte ODBC é aberto como mostrado a seguir.

```
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open "string de conexao contendo também o nome dado ao banco de dados"
```

Então é necessárioN se estabelecer uma conexão como bando de dados, como por exemplo usando-se uma consulta SQL para retornar dados.O próximo exemplo constrói um conjunto de registros contendo os registros retornados do banco de dados referidos na consulta SQL:

```
set RS = Conn.Execute("SELECT * FROM [table] WHERE [field]=1")
```

Este conjunto de registros gerado pode ser navegado, e entao é possível escrevê-los na página web como no exemplo que se segue:

```
If not RS.eof then
        RS.movefirst
        Do
                Response.write "Algum dado " & RS("Field")
                Rs.movenext
        Loop until RS.eof
```

```
End if
```

No código abaixo, os dados que serão mostrados ná página são preparados e depois formatados usando comandos HTML.

```
<HTML>
<HEAD>
<TITLE> Exemplo </TITLE>
<SCRIPT LANGUAGE="VBScript">
<!--
Dim MyString, SpaceCount, Length, Position
MyString = "Esta é uma string de teste"
SpaceCount = 0
Length = Len(MyString)

For Position = 1 to Length
 If Mid(MyString, Position, 1) = Chr(32) Then
  SpaceCount = SpaceCount + 1
 End If
Next
MsgBox SpaceCount

'SpaceCount seria igual a 4
-->
</SCRIPT>
</HEAD>
</BODY>
</HTML>
```

3 – DESCRIÇÃO DO PROBLEMA E OBJETIVOS

Este projeto tem o objetivo de implementar uma ferramenta que possibilite um auxílio para a tradução de grandes sítios.

E por que não usar máquinas de tradução já existentes na Web?

Máquinas para traduzir idiomas instantaneamente tem sido prevista há cerca

de cinqüenta anos, e ainda a necessidade por traduções feitas por seres humanos é muito grande. Nada pode substituir um trabalho preciso que um tradutor humano produz.

Confiar em um computador para uma tradução precisa pode ter resultados desastrosos.
Seguem alguns exemplos:
1) O famoso slogan da Coca-Cola "Coca adiciona vida" é traduzido em chinês como "Coca traz seus ancetrais de volta à vida".
2) "Serviços sociais" é freqüentemente traduzido como "Limpeza de sanitários".
Ou seja, a tradução deve ser feita por pessoas e não por tradutores eletrônicos. Porém encontrar o texto a ser traduzido em meio a códigos em ASP é uma tarefa difícil, desde que arquivos em ASP podem conter códigos de diversas linguagens de programação diferentes. Essa tarefa pode se tornar muito monótona e propensa a erros.

O Analisador sintático neste caso tem a função de percorrer o arquivo fonte em busca do texto que será exibido na página final. Cada pedaço de texto encontrado é extraído e substituído por uma chamada de função que posteriormente irá recolocar o texto já traduzido no mesmo local.

COMPLEXIDADE

A complexidade dessa ferramenta consiste no contexto da estrutura interna de um arquivo ASP, que pode possuir vários componentes e códigos em diversas linguagens como HTML, JavaScript, VBScript e qualquer outra compatível com ActiveScript. Estes códigos são separados por identificadores que determinam também qual é a linguagem utilizada.

A busca de textos que serão exibidos na página Web deve ser então muito cautelosa, analizando cada caracter do arquivo que possa ser um separador/identificador. A implementação de um parser específico para o retorno de "tokens" é necessária, pois será um trabalho de compilação léxica e sintática. Todo o contexto da frase deve ser analisado para se indentificar um texto que deve ser exibido na tela final.

4 - SOLUÇÕES PROPOSTAS

Uma solução para o problema é definir todas as funções que manipulam strings de forma a especificar quais situações em que a string em questão é um texto a ser traduzido ou não.

Divide-se então essas funções em dois grupos, situações em que a string deve ser extraída e situações em que ao encontrarmos uma aspas, o texto em seguida deve ser ignorado até encontrar uma situação em que a string deve ser extraída.

4.1 Palavras-chave que denotam uma string a ser traduzida:

4.1.1 JavaScript

Expressões:

```
window.status='Hi there!'
document.myform.mytext.value="texto a extrair")    // neste caso value aparece
                                                      como propriedade  de objetos
alert('texto a extrair')
confirm("...");
prompt('texto a extrair', ' ');
document.write("…")
```

4.1.2 VBScript

Expressões:

```
confirm("texto a extrair")
document.write("…")
StaticNames(0) = "texto a  extrair"
```

Obs.: Ou qualquer variável que tenha sido declarada normalmente com o comando "Dim" e receber alguma string.

4.1.3 HTML

Qualquer coisa que estiver entre tags HTML, ou seja, entre os símbolos > e < , desd que estes não estejam dentro de um script ou não sejam caracteres de escape.

Ex.: value="Cancelar" em <input type=button name = “cmdCancel” value=”Cancelar”>
// neste caso *value* aparece dentro de uma *tag* html, como propriedade de um componente

4.1.4 ASP

Todas as varíaves às quais são atribuídas string de texto.

Ex.: Application (“var1”) = “string1”
Session (“var2”) = “string2”

Obs.: Nestes dois exemplos as variáveis a serem traduzidas são string1 e string2, pois Application(“var1”) e Session(“var2”) funcionam como variáveis comuns.

4.2 Palavras-chave e situações onde uma string não deve ser traduzida:

4.2.1 JavaScript

Funções e propriedades:

```
language="JavaScript"
document.bgColor='yellow'
onMouseout=" "
onMouseover=" "
window.location='http..'
top.location="http://someplace.com";
setTimeout('getgoing()',)
window.open('url to open','window name','attribute1,attribute2')
style="..."
eval(imgName + "off.src");
version="n2";
mytool.split("/");
```

Situações

1) O método onClick define operações que irão ocorrer a um clique do mouse. Deve-se ignorar o que vêm após as aspas até que se encontre uma expressão onde deve-se extrair string texto:

Ex.: onClick="window.status='You clicked the button!'; return true"
onClick="history.back()"
onClick="parent.right_frame.document.form1.text1.value='Me!'"

2) O método onMouseOver é um caso semelhante ao de onClick :

Ex. : onMouseover="alert('Hey! Dont click on this link!')">
// neste exemplo, após as aspas encontra-se a função “alert”, onde tudo que vier depois de (“ deve ser extraído

3) Expressões booleanas de comparação:

Ex. :if (browserName=="Netscape")

4) Índices de *arrays* dinâmicos:

Ex. : my_cars["cool"]=

4.2.2 VBScript

Funções de manipulação de string:

Left(), Mid(), Right(), Instr(), Replace(), Split(), UCase(), LCase(), Len() and Trim().

Situações:

1) Variáveis contendo consultas SQL:

```
sql = “SELECT “ & _
        “UPDATE “ & _
        “INSERT “ & _
        ‘INCLUDE “ & _


sql = “SELECT
        “UPDATE
        “INSERT
        ‘INCLUDE
```

obs.: Em VBScript como em Visual Basic, quando uma string começa em uma linha e termina em outra, os símbolos ‘”& _ ‘ devem ser incluídos ao final da linha de cima para concatenação das linhas.

4.2.3 HTML

Expressões de atribuição:

<form method="post" action="formProcessingScript">
// outras propriedades de formulário:
name, id, size e value(positiva), class e Maxlenght como descrito abaixo

<IMG SRC="scare.jpg" border="0">
<AREA SHAPE="RECT"
<IMG SRC="eximap1.gif" width="200" height="40" border="0" alt="imagemap" USEMAP="#mymap">
<HREF="jmouse.htm"
<input type="button" name="button6">
<BODY text="lime">
<BODY onLoad="...">’

Situações:

1) A propriedade *src* define que a string entre aspas é uma referência

Ex.: pic1.src="http://someplace.com/image1.gif"

2) Quando for uma tag começando com “OPTION SELECTED”, ocorre uma exceção no caso de *value:*

Ex.: <OPTION SELECTED value="jex6.htm">

4.2.4 ASP

Expressões:

Request.form("txtUserID")

Server.ScriptTimeout
Server. CreateObject
Server. HTMLEncode

Server. URLEnconde
Server. Execute
Response.Clear
Response.End
Response.Flush
Response.Redirect
Application (“var1”) = “string1”
Session (“var2”) = “string2”
// var 1 e var2 não devem ser traduzidas, apenas string1 e string2

CreateParameter(“…”, …)
Case “…”

Situações:

1) String de conexão para o banco de dados:

```
StrConn = "Provider=SQLOLEDB;UserID=sa;Password=;Data
Source=(local);Initial Catalog=Asp101"
```

Exemplos de utilização dessa string de conexão:

```
 - Conn.Open   strConn
- cn.Open "Data Source=(local);Initial Catalog=Asp101" & _
          "Provider= SQLOLEDB;UserID=sa;Password= ;"
```

Obs.: Ela sempre vírá após o método Open. As variáveis “Conn” e “cn” do exemplo foram definidas da forma:
set cn = Server.CreatObject(“ADODB.Connection”)

Esse casos, assim como string de consulta SQL, podem ou não serem analisados.Caso essas strings texto sejam extraídas por engano, a pessoa que for traduzí-las saberá que elas nao são textos que precisem de tradução, portanto incluí-las na análise ainda é apenas uma possibilidade.

2) Expressões booleanas

```
<% If a="some value" Then %>
// entre comando If  THEN ELSE , while, for, etc.
```

3) Importação de arquivos no código

```
<!--#include file="include1.asp"-->
```

4) Strings utilizadas em propriedades de objetos

Ex.: rs.fields("user_id").value

5) Variáveis nao definidas com Dim ou que instanciem objetos:

```
Set cmd = Server.CreateObject("ADODB.Command")
set rs = Server.CreateObject("ADODB.RecordSet")
```

5 - CRONOGRAMA

ATIVIDADES

- Estudo das especificações de como os códigos de outras linguagens são separados do resto do código do arquivo ASP e de como estes manipulam textos a serem exibidos na tela final;
- Estudo de geradores de parser como o Lex ou o Yacc;
- Implementação inicial da aplicação;
- Confecção do Relatório Final.

Baseados nessas atividades, foi criado o seguinte cronograma:

Atividade	Data
Proposta	23/03/2001
Estudo dos casos possíveis	24/03/2001

Geradores	16/05/2001
Aplicação	01/06/2001
Relatório Final	24/03/2001
Apresentação oral	29/06/2001

6 – TRALHOS FUTUROS

Nesta fase foram feitas pesquisas nos temas a serem trabalhados, como a sintaxe das linguagens script, HTML e de ASP.
A implementação de procura de textos entre tags HTML já foi feita. Esta parece ser trivial, mas ao analisarmos que os delimitadores de tag < e > podem aparecer dentro de códigos de scripts, percebe-se que esta não é tão simples.
A implementação final será feita em Projeto Orientado II.

7 - CONCLUSÃO

8 - REFERÊNCIAS

1. Izar Solutions Web site - http://www.izar.com

2. Internationalization - www.w3c.com

3. Intela Servicesis – http://www.intela.com

4. Introduction to Visual Basic Scription (VBScrpit) - http://tech.irt.org/articles/js117

5. http://www.advantage.co.nz/ur/aspquiz.htm

6. Tutoriais - http://www.astentech.com/tutorials