

Especificação Sk_access_tcp.dll V1.1



Foto: SKO-44

Teclado de automação, Tecnologia Óptica
com display LCD e leitor magnético.

Descrição:

Sk_access_tcp.dll é uma biblioteca de funções desenvolvida pela Smak Teclados, com o objetivo de facilitar o trabalho daqueles que desenvolvem softwares para os produtos SMAK. Este documento destina-se apenas a desenvolvedores e contém as informações técnicas indispensáveis ao uso da DLL.



Especificação Sk_access_tcp.dll

out_bufferSumário

Histórico de alterações deste documento.....	3
Descrição SK_access_tcp.dll	4
Plataforma unificada.....	4
Quantidade de terminais.....	4
Demonstração e testes das funções da DLL Smak	5
Instalando a DLL	5
Utilizando a DLL da SMAK em seu ambiente de desenvolvimento	8
Tabela de funções e procedures disponibilizadas por sk_access_tcp.dll	9

Histórico de alterações deste documento:

Revisão 1.0 (1-03-2010) :

- Introdução de sk_access_tcp.dll versão 1.1.

Descrição SK_access_tcp.dll :

O teclado de automação SKO-44 Ethernet disponibiliza em sua documentação técnica, uma série de comandos necessários para que softwares possam ser desenvolvidos de modo a utilizar o teclado de forma apropriada. Adicionalmente a SMAK desenvolveu também uma DLL chamada “sk_access_tcp.dll” cujo objetivo é disponibilizar ao profissional de programação algumas funções e procedimentos essenciais para facilitar o trabalho de desenvolvimento de software.

Plataforma unificada:

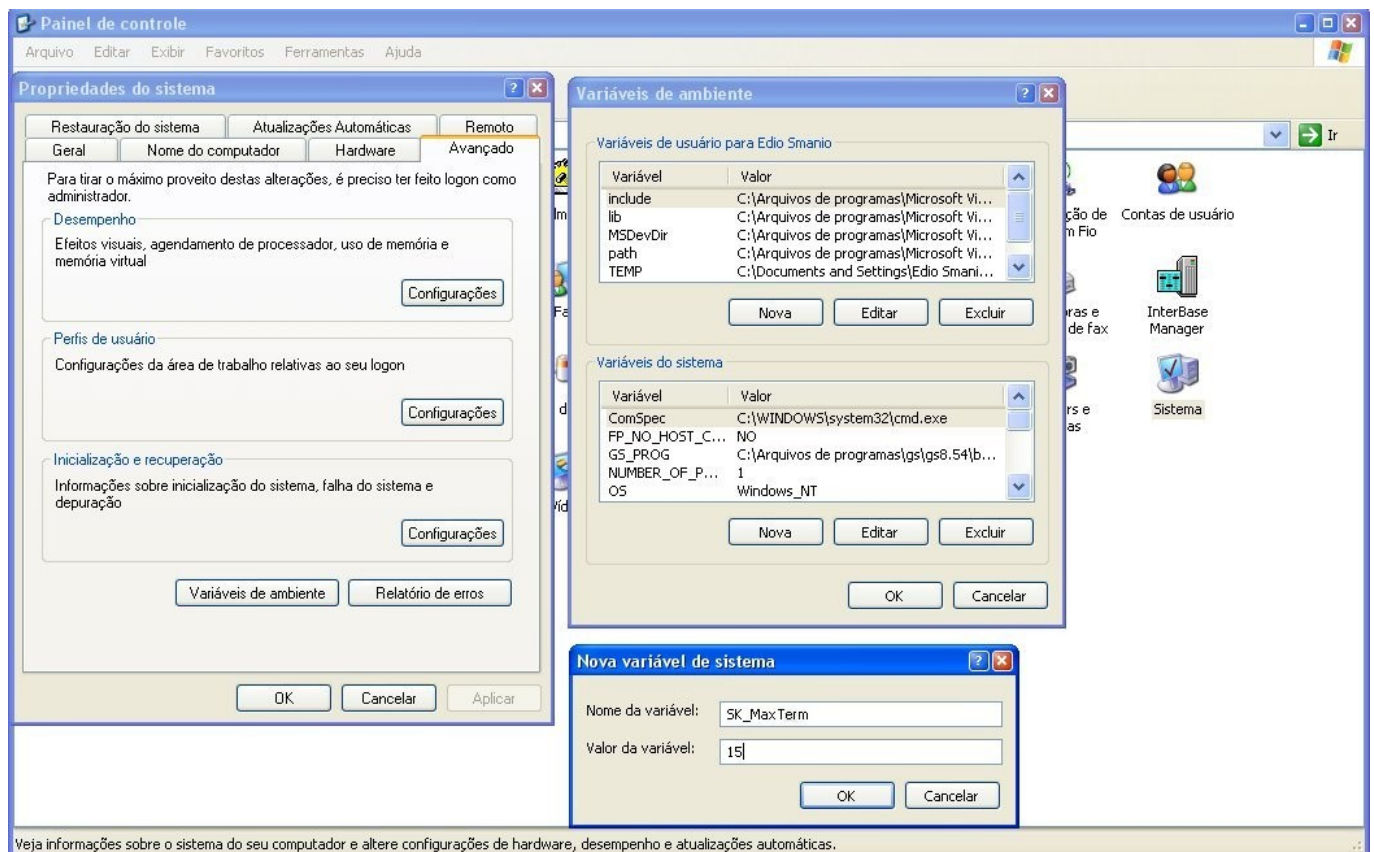
A DLL oferece uma plataforma unificada de desenvolvimento, de forma que um aplicativo feito utilizando suas funções, pode, Com pequenas modificações, acessar indistintamente os teclados nas interfaces PS2, serial RS-232 e USB,

Quantidade de terminais:

Por padrão o servidor telnet iniciado pela DLL atende a somente 2 conexões.

A variável de ambiente SK_MaxTerm redefine o número de terminais que vão ter suas conexões aceitas, para criar/editar esta variável, no windows, deve-se acessar:

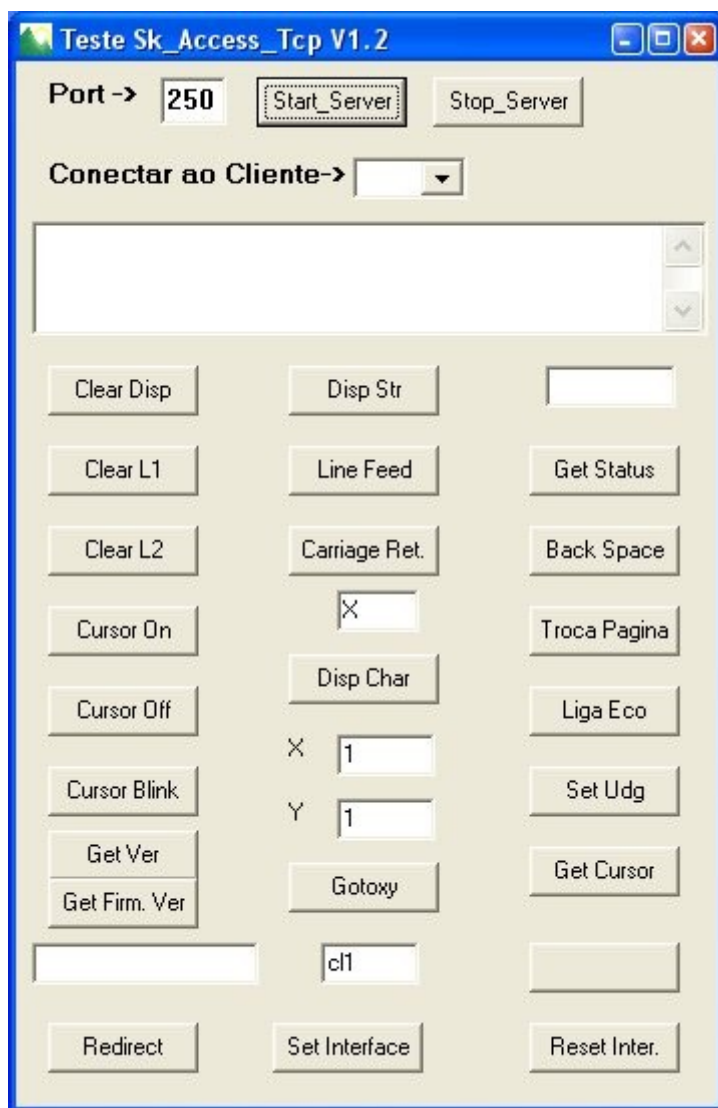
Painel de controle -> Sistema -> Avançado -> Variáveis de ambiente -> Nova/Editar



Demonstração e testes das funções da DLL Smak :

O utilitário de teste disponibilizado em Delphi3 (contendo inclusive o código fonte) foi desenvolvido especificamente como um exemplo para verificação da operação das funções exportadas pela DLL. A figura a seguir ilustra sua aparência no Windows.

Cada botão tem o nome da função sendo testada. É importante lembrar que Este documento se refere à última versão da DLL sendo que versões anteriores podem não suportar todas as funções descritas aqui.



Instalando a DLL :

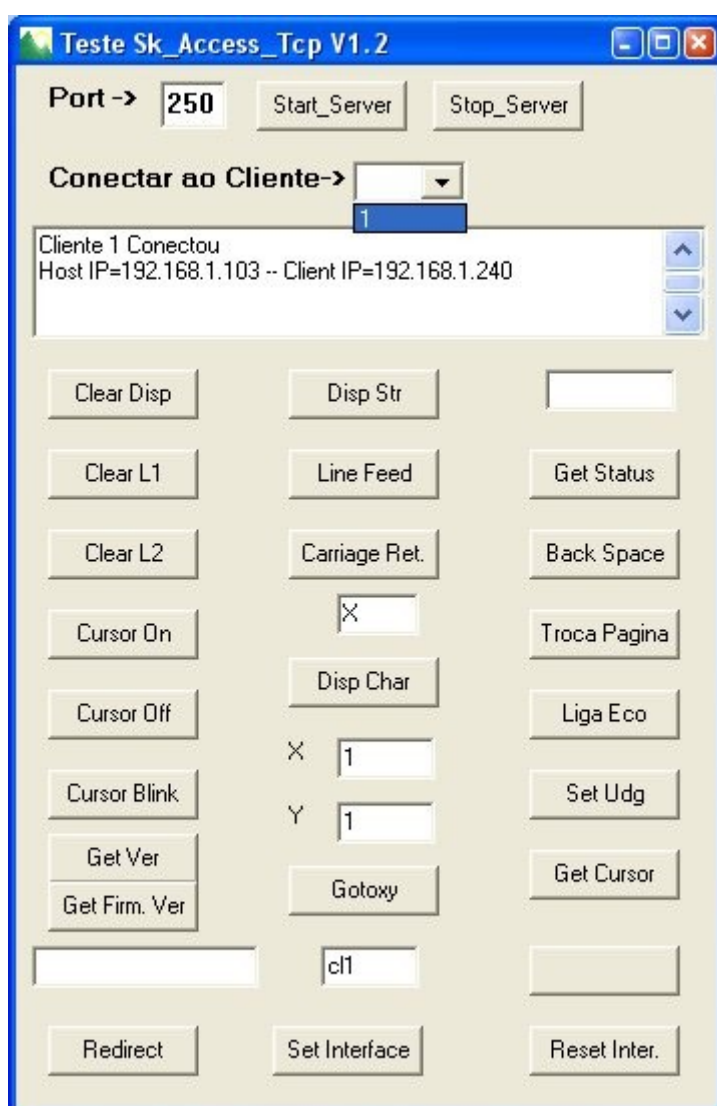
A instalação de consiste em obter o pacote de software, disponível no site www.smak.com.br, descompactar e copiar manualmente os arquivos nos diretórios a que se destinam.

Passos para a instalação Windows 9x ou XP:

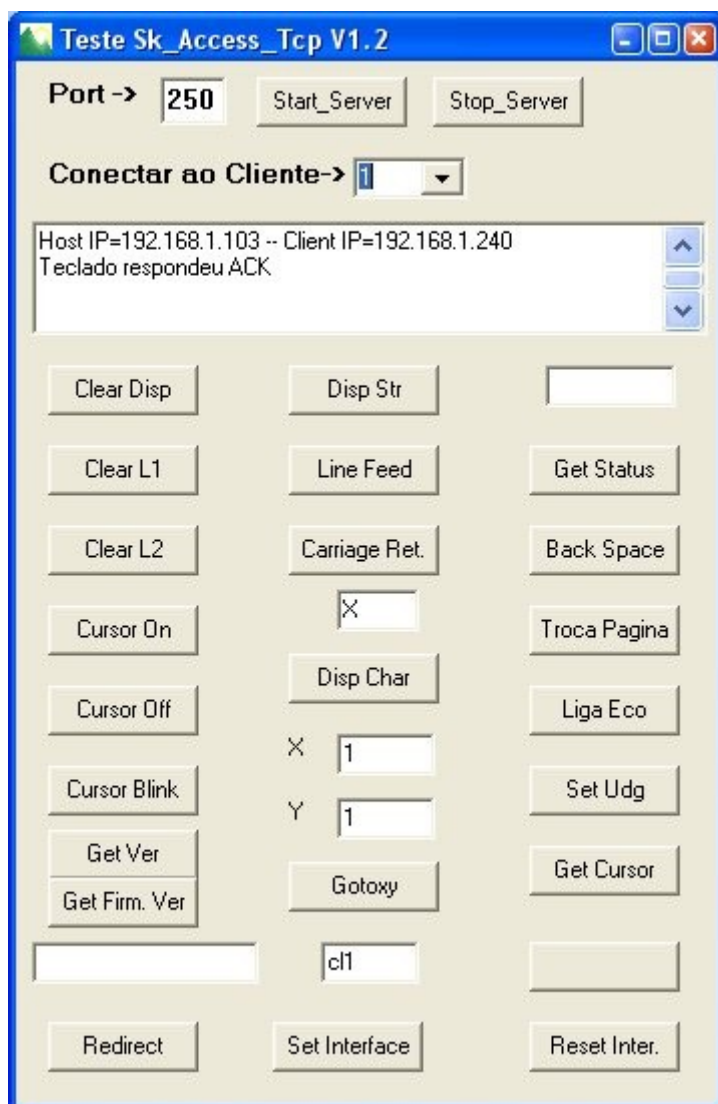
1. Descompacte em uma pasta temporária o software que pode ser obtido via download no site www.smak.com.br.
2. Copie a DLLs para o diretório "C:\WINDOWS\SYSTEM32".
3. Crie uma pasta de trabalho por exemplo "C:\SMAK" destinada a receber o utilitário de teste e demais arquivos do SK044, tais como documentos, exemplos, etc.
4. Testes: Conecte o computador e um teclado SKO-44 ethernet à mesma rede , execute o utilitário de testes e em seguida teste alguns comandos.

Exemplos de testes:

Com o utilitário na tela click em "Start_Server", isso deve iniciar o servidor telnet de forma que os terminais poderão ter suas solicitações de conexão aceitas.



Após observar na janela de status que um cliente se conectou, selecionar o cliente para torná-lo o cliente ativo (colocá-lo em foco) de forma que dados serão recebidos e enviados exclusivamente para ele.



- Teste da função DISP: Click o botão “Disp Str”. A primeira linha da janela de status será enviada para o terminal selecionado.

- Teste da função CLEAR_L1 e CLEAR_L2: Click os botões “Clear_L1” e “Clear_L2” para limpar as linhas do display.

Para saber sobre todas as funções exportadas pela DLL, consulte a informação na tabela no final deste documento.

Utilizando a DLL da SMAK em seu ambiente de desenvolvimento :

Uma vez que você conseguiu verificar o correto funcionamento do utilitário de testes, isto significa que *"sk_access_tcp.dll"* está corretamente disponível para uso no sistema. É interessante lembrar que em caso de dúvidas sobre o uso das funções da DLL, o profissional desenvolvedor poderá consultar o código fonte do aplicativo de testes disponibilizado em Delphi3.

Notar que quando uma função é chamada, essa direciona seus dados para o cliente em foco, para colocar um cliente em foco a função utiliza da é a *Set_Interface*.

Set_Interface recebe uma string do tipo "cl1" .. "cln" para selecionar cliente 1 até cliente n, retorna -1 se cliente não esta conectado ou 1..n se estiver.

A string "cl0" coloca em foco cliente nenhum.

Além de enviar os dados para o cliente em foco, a o servidor também gerencia a recepção de dados dos clientes.

A troca de informações no sentido DLL(servidor telnet) -> aplicativo é feita fundamentalmente através de mensagens de sistema, que são geradas pelo servidor telnet e devem ser capturadas pelo aplicativo.

O servidor gera dois tipos de mensagens: conexão e comunicação.

As mensagens de *conexão* informam o tipo de evento (conexão/desconexão) , o ID e o IP do cliente.

As mensagens de *comunicação* informam o ID do cliente que gerou a comunicação e o valor da tecla pressionada no cliente.

A função que ativa o servidor é:

Procedure Start_Server(hWnd: HWND;M_CONNECT,M_COMMUNIC:Dword); stdcall;

Esse procedimento recebe 3 parâmetros importantes que são:

hWnd: HWND = Handler do aplicativo que vai receber as mensagens do servido telnet.

M_CONNECT:Dword = Número da mensagem de conexão que o servidor deve usar que está sendo esperada pelo aplicativo.

M_COMMUNIC:Dword = Número da mensagem de comunicação que o servidor deve usar que está sendo esperada pelo aplicativo.

Se hWnd é passado como 0(NULL) o servidor telnet não gera mensagens.

Veja à seguir a tabela com as funções e procedimentos disponíveis na DLL Smak.

Tabela de funções e procedures disponibilizadas por sk_access_tcp.dll :

Index	PROTÓTIPO	DESCRIÇÃO
1	Function Send_Data(data:byte):boolean; stdcall;	Envia um byte para o terminal
2	Function Send_Disp_Ctrl(data:byte): boolean; stdcall;	Envia comando direto para o display
3	Function Clear_Disp:boolean; stdcall;	Apagar o display
4	Function Clear_L1:boolean ; stdcall;	Apagar a linha 1 do display
5	Function Clear_L2:boolean ; stdcall;	Apagar a linha 2 do display
6	Function Cursor_Off:boolean ; stdcall;	Desligar o cursor no display
7	Function Cursor_On:boolean ; stdcall;	Ligar o cursor
8	Function Cursor_Blink :boolean ; stdcall;	Piscar o cursor
9	Function Back_Space:boolean; stdcall;	Back space do cursor
10	Function Line_Feed :boolean ; stdcall;	Line-feed no display
11	Function Carriage_Return : boolean; stdcall;	Carriage return
12	Function Gotoxy(x,y : byte): boolean ; stdcall;	Posiciona cursor na Coluna X, linha Y. (sendo: y: 1ou 2 e x:de 1 a 40)
13	Function Disp_Char(data:char) : boolean; stdcall;	Exibe um caracter no display
14	Function Disp(data:PChar) : boolean; stdcall;	Exibe uma string de até 80 caracteres.
15	Function get_cursor(x_pos:pointer; y_pos:pointer):byte; stdcall;	Lê em x_pos e y_pos as coordenadas atuais do cursor (home = 1,1), Adicionalmente retorna a posição absoluta do cursor (0 a 79). Caso o teclado não responda retorna 255.
16	Function get_status:byte; stdcall;	Lê status da última operação, Retorna: 00h se não houve erros; 01h se houve erro e FFh se saiu por time-out.
17	Function Set_page1 : boolean; stdcall	Ativa pagina 1 de scancode.
18	Function Set_page0 : boolean; stdcall	Ativa pagina 0 de scancode.
19	Procedure Free_sk_access; stdcall;	Compatibilidade com sk_access.dll
20	Function GetOperatingSystem: Integer; stdcall;	Identifica sistema operacional retorna um valor numérico de -1 a 6 sendo: -1=Desconhecido; 0=Win95; 1=Win98; 2=Win98se; 3=WinME; 4=WinNT; 5=Win2000 e 6=WinXP
21	Procedure Set_Eco_On; stdcall;	Liga eco
22	Procedure Set_Eco_Off; stdcall;	Desliga eco
23	Function Set_Interface(interf:Pchar):integer; stdcall;	Recebe string "cl1" a "cln" para selecionar cliente 1..n. Retorna 1..n se cliente conectado e -1 se não conectado
24	Function Select_Interface:integer; stdcall;	Retorna:-1, Somente para compatibilidade com sk_access.dll
25	Procedure Reset_Interface; stdcall;	Desliga eco e redirecionamento de todos os terminais, tira foco dos terminais
26	Procedure Get_DLL_Version(ver:PChar); stdcall;	Retorna string com versão da DLL , max 20 caracteres.

Index	PROTÓTIPO	DESCRIÇÃO
27	Procedure Set_Udg(caracter,def0,def1,def2,def3,def4,def5,def6,def7:byte); stdcall	<p>UDG (User Defined Graphics) Permite o desenho de até oito caracteres especiais que ficam armazenados nas posições 0 a 7 da tabela interna do display. Na procedure, o parâmetro "caracter" representa o código a ser atribuído ao desenho. "def0" a "def7" definem os elementos ativos da matriz 5x7 do caracter no display.</p> <p>Veja abaixo um exemplo dos valores de "caracter" e de "def0 a 7" para formar a letra 'R' na posição 01h da tabela interna</p> <pre> caracter=01h def1 = x x x 1 1 1 1 0 = 1Eh def2 = x x x 1 0 0 0 1 = 11h def3 = x x x 1 0 0 0 1 = 11h def4 = x x x 1 1 1 1 0 = 1Eh def5 = x x x 1 0 1 0 0 = 14h def6 = x x x 1 0 0 1 0 = 12h def7 = x x x 1 0 0 0 1 = 11h </pre>
28	Procedure Redirect; stdcall;	Captura dados do terminal e redireciona para o buffer do teclado.
29	Function Get_Data_Rs (buffer:pointer;size:integer): integer; stdcall;	Transfere para "buffer" um máximo de "size" caracteres lidos da porta serial. Retorna o número de caracteres efetivamente lidos.
30	Procedure Keyb_Reset; stdcall	Reseta o teclado.
31	Procedure Get_Firmware_Version(ver:PChar); stdcall;	Retorna string no formato (TCP)=v.vvv
32	Function Get_Com_Type:boolean; stdcall;	Retorna TRUE = Virtual , compatibilidade com sk_access.dll
33	Function Set_Client(cl:byte):integer; stdcall;	Seleciona cliente pelo número, Retorna -1 se falha ou nro do cliente se OK.
34	Procedure Set_Port(port:byte); stdcall;	Define número da porta do servidor TELNET
35	Procedure Get_Host_IP(ip:Pchar); stdcall;	Devolve em ip uma string com IP do servidor.
36	Procedure Get_Client_IP(ip:Pchar); stdcall;	Devolve em ip uma string com IP do cliente em foco.
37	Function IP_ascii2HEX(ip_ascii:Pchar): Dword; stdcall	Converte endereço IP em ASCII para HEX.
38	Function IP_HEX2ascii (ip:Dword): Pchar; stdcall	Converte endereço IP em HEX para ASCII.
39	Function Inv_IP_HEX(ip:Dword): Dword; stdcall	Inverte endereço IP HEXA MSB = LSB
40	Function IP_HEX2ascii (ip:Dword): Pchar; stdcall	Converte endereço IP em HEX para ASCII.
41	Procedure Kill_Server; stdcall;	Desativa e remove o servidor Telnet.

Index	PROTÓTIPO	DESCRIÇÃO
42	<pre> Procedure List_Clients(lista:PTsockets); stdcall; lista[1] = IP do terminal 1; lista[2] = IP do terminal 2; . . . Se IP=0000, não há cliente conectado. type Tsockets=record ip:array[0..255] of uint end PTsockets= ^Tsockets </pre>	Transfere para “lista” os dados dos terminais conectados.

Index	PROTÓTIPO
43	<pre> Procedure Start_Server(hWnd: HWND; M_CONNECT,M_COMMUNIC: Dword); stdcall; </pre> <p>-Esse procedimento ativa o Servidor telnet para que possa receber conexões dos terminais. -O cada comunicação que o servidor troca com um terminal ele envia uma mensagem para o programa do usuário para que as comunicações sejam tratadas.</p> <p>hWnd : HWND = Handler do programa do usuário que vai receber as mensagens</p> <p>M_CONNECT = Número da mensagem de conexão/desconexão M_COMMUNIC = Número da mensagem de comunicação</p> <p>Parâmetros das mensagens enviadas:</p> <p>M_CONNECT -> Wparam = IP do Cliente Lparam HI = ID do cliente Lparam LO = 1 se cliente conectou e 0 se desconectou</p> <p>M_COMMUNIC -> Lparam HI = Codigo da tecla precionada no teclado Lparam LO = ID do cliente</p> <p>Exemplo:</p> <pre> const WM_CONNECT = WM_USER + 0 //Define número das mensagens WM_COMMUNIC = WM_USER + 1 type TForm1 = class(TForm) private {-----Define Procedimentos para receber mensagens da DLL-----} Procedure deal_CONNECT(var Msg: TMessage) message WM_CONNECT Procedure deal_COMMUNIC(var Msg: TMessage) message WM_COMMUNIC public { Public declarations } end </pre>

```

{-----
    Ativa Servidor de teclados SMAK
-----}
procedure TForm1.BStart_ServerClick(Sender: TObject)
begin
    Set_Port(250)
    Start_Server(self.Handle,      //Handler para esse aplicativo
                 WM_CONNECT,      //Passa número da mensagem de conexão
                 WM_COMMUNIC);    //Passa número da mensagem de comunicação
end

{-----
    Procedimento que recebe evento de conexão
    Wparam = IP do cliente em hexadecimal
    LParamHi = ID do cliente
    LParamLo = 1 para conexão 0 para desconexão
-----}
Procedure TForm1.deal_CONNECT(var Msg: TMessage)
var
    conec_msg:string[40]
begin
    if Msg.LParamLo=1 then begin
        {Trata conexão do cliente Msg.LParamHi}
    end else begin
        {Trata desconexão do cliente Msg.LParamHi}
    end
end
end

{-----
    Procedimento que recebe evento de comunicação
    Wparam = Tipo de comunicação
    LParamHi = ID do cliente
    LParamLo = Caracter recebido
-----}
procedure TForm1.deal_COMMUNIC(var Msg: TMessage)
begin
    {Trata caracter recebido=Msg.LParamLo, do cliente ID=Msg.LParamHi}
end

```