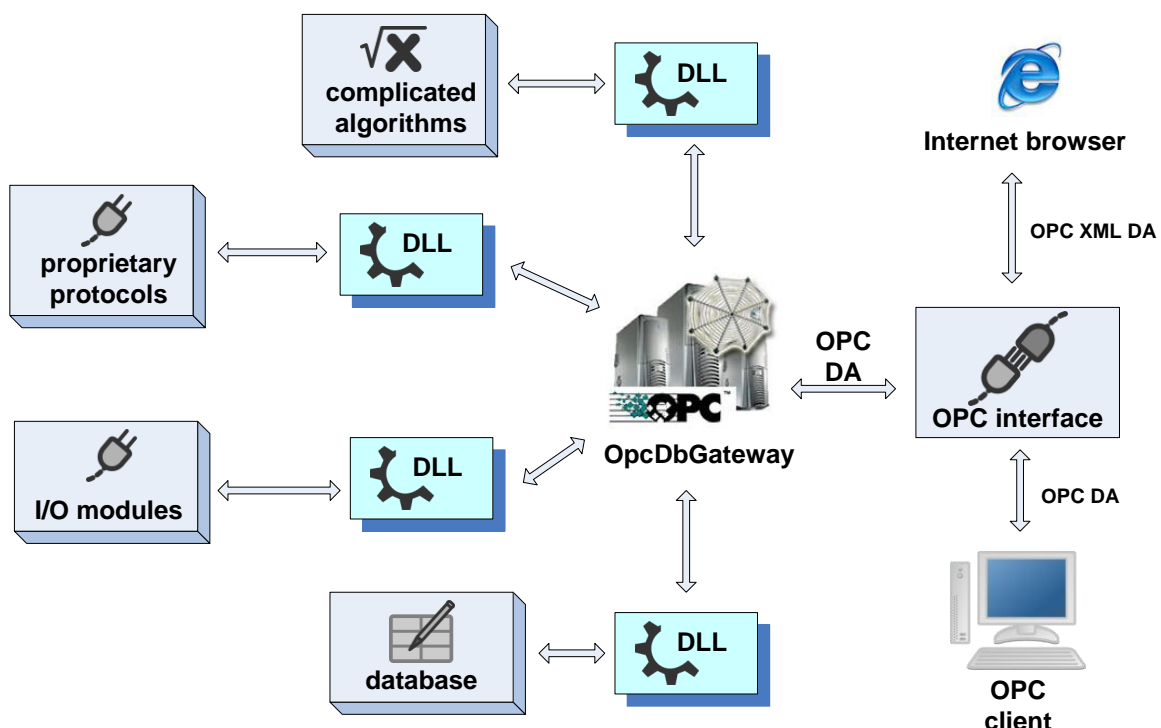


OpcDbGateway umožňuje využitie DLL a ActiveX iných dodávateľov pre rozšírenie funkcionality

OpcDbGateway posiela e-mail prostredníctvom externej DLL knižnice, z ktorej sú ďalej volané metódy ActiveX komponenty.

Efektívnosť OpcDbGateway pri integrácii aplikácii je zabezpečená využitím **konfigurácie/parametrizácie** namiesto prácnejšieho programovania. Jeho **flexibilita** je zabezpečená možnosťou vytvoriť parametrizovateľné komponenty DLL alebo ActiveX od rôznych dodávateľov.



Obrázok 1: Použitie externej DLL knižnice pre rozšírenie funkcionality OpcDbGateway.

Je viacero dodávateľov na trhu ktorí ponúkajú funkcionality posielania **e-mailov implementovanú napríklad vo forme ActiveX**. My si v našom príklade jednu takúto dostupnú knižnicu vyberieme a ukážeme si ako jednoducho pridať jej funkcionality do *OpcDbGateway*.

OpcDbGateway a externá DLL

OpcDbGateway ponúka možnosť zavádzať do svojho procesu jednu, alebo aj súčasne viaceré externé DLL. Externá DLL musí mať ale implementované definované rozhranie pozostávajúce z nasledovných funkcií (pozri na Obrázok 2):

- GetProductName,
- GetProductVersion,
- GetCompanyName,
- GetLegalCopyright,
- GetDescription,
- GetCountOfIO,
- DoProcessIO.

```
#define DllExport extern "C" __declspec (dllexport)
#define DllImport extern "C" __declspec (dllimport)

/////////////////////////////////////////////////////////////////
// EXTERNAL DLL INTERFACE
/////////////////////////////////////////////////////////////////
// The function retrieves a DLL product name as string.
DllExport DWORD WINAPI GetProductName (LPTSTR lpReturnedString, DWORD dwSize);
// The function retrieves a DLL product version as string.
DllExport DWORD WINAPI GetProductVersion (LPTSTR lpReturnedString, DWORD dwSize);
// The function retrieves a DLL company name as string.
DllExport DWORD WINAPI GetCompanyName (LPTSTR lpReturnedString, DWORD dwSize);
// The function retrieves a DLL copyright as string.
DllExport DWORD WINAPI GetLegalCopyright (LPTSTR lpReturnedString, DWORD dwSize);
// The function retrieves a DLL description as string.
DllExport DWORD WINAPI GetDescription (LPTSTR lpReturnedString, DWORD dwSize);
// The function retrieves a specified count of inputs/outputs.
DllExport void WINAPI GetCountOfIO (LPWORD lpInputs, LPWORD lpOutputs);
// The function processes all values from input buffer and retrieves the processed values
DllExport bool WINAPI DoProcessIO (const CComVariant* lpInputs, WORD wInCnt,
                                   CComVariant* lpOutputs, WORD wOutCnt);
```

Obrázok 2: Rozhranie ktoré musí byť implementované v externej DLL, aby bolo možné volať jej funkcie z aplikácie *OpcDbGateway*.

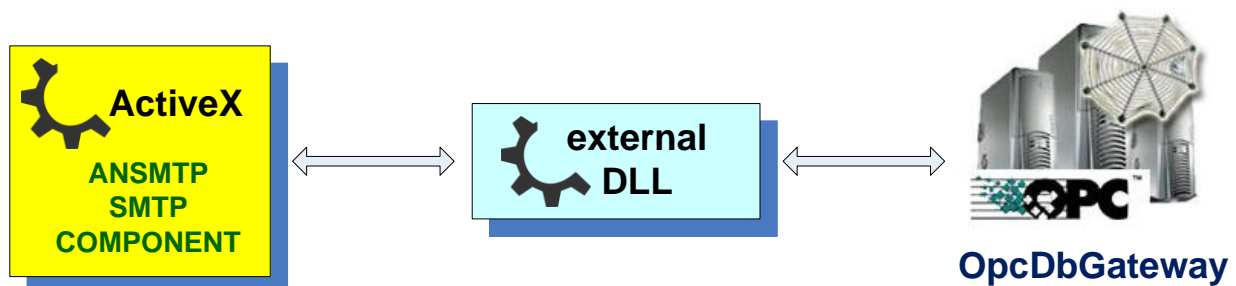
V externej **DLL** je možné implementovať prakticky ľubovoľnú funkcionality ktorá v spojení s aplikáciou **OpcDbGateway** získava aj **d'alší podstatný rozmer**. Údaje spracovávané v **OpcDbGateway** sú dostupné súčasne v **DLL** a naopak údaje spracovávané v **DLL** sú súčasne dostupné v **OpcDbGateway**.

Typické prípady použitia **externej DLL** v spolupráci s **OpcDbGateway** sú nasledovné (pozri na Obrázok 3):

- V externej DLL sú implementované náročné výpočty, alebo algoritmy.
- V externej DLL je implementovaný komunikačný protokol (napr.: SMS, e-mail, atď.).

- V externej DLL je implementovaný prístup na databázu (napr.: SQL Server, ORACLE, atď.).
- V externej DLL je implementované volanie funkcií z iných DLL, ActiveX alebo aplikácií, od iných dodávateľov. (napr. Systémové knižnice, ADO, etc.).
- Externá DLL plní úlohu sprostredkovateľa.

V našom príklade sme využili ActiveX komponentu pre posielanie e-mailov od iného dodávateľa. Pod názvom **ANSMTP SMTP COMPONENT**¹ ktorá môže byť jednoducho volaná aj z kódu programovacieho jazyka Visual C++.

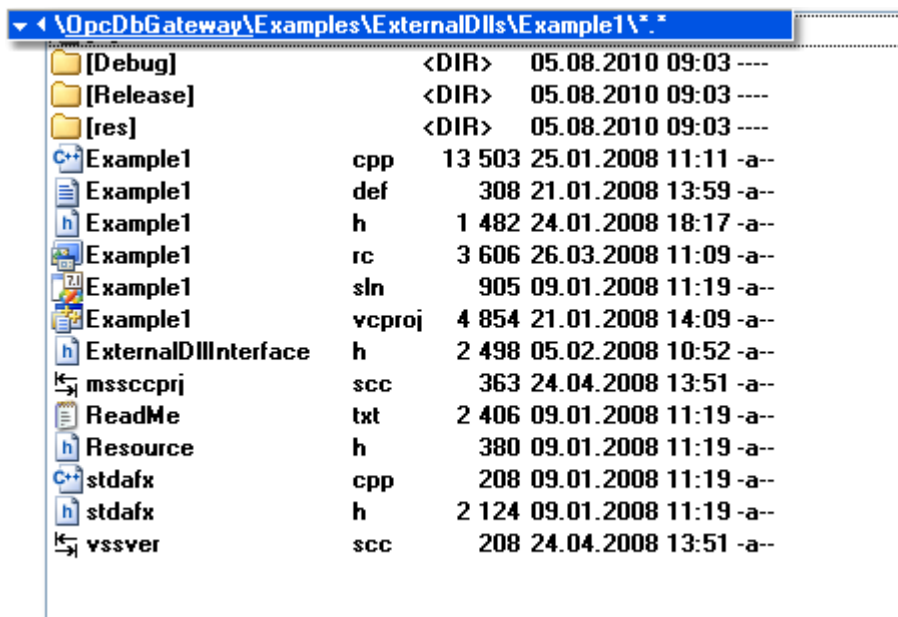


Obrázok 3: Príklad, keď externá DLL pôsobí ako sprostredkovateľ medzi ActiveX komponentou (ANSMTP SMTP COMPONENT) a aplikáciou OpcDbGateway.

Vytvorenie externej DLL ktorá sprostredkuje metódy komponenty ANSMTP SMTP COMPONENT

Vytvorenie vlastnej externej DLL nie je zložité. Inštalčný balíček OpcDbGateway obsahuje aj zdrojové kódy ukážkovej extejnej DLL nazvanej **Example1** (Obrázok 5), ktorá zahŕňa všetky povinné funkcie, tak ako boli uvedené v dokumente hore.

¹ Viac príkladov a podrobnosti o použití a práci s komponentou ANSMTP SMTP sa dozviete na stránke <http://www.emailarchitect.net/webapp/smtpcom/>



Obrázok 4: Zdrojové kódy príkladovej externej DLL nazvanej Example1.

Po nainštalovaní komponenty **ANSMTMP SMTP Component** sa **AOSMTMP.dll** uloží na požadovanej ceste. Potom stačí komponentu len v zdrojovom kóde externej DLL sprístupniť prostredníctvom príkazu **#import**.

```
#import "C:\Program Files\AdminSystem.NET\ANSMTMP\AOSMTMP.dll"
```

Obrázok 5: Importovanie funkcií ActiveX komponenty (ASNMTMP SMTP COMPONENT) do externej DLL vytváranej pre aplikáciu OpCdbGateway.

To, že sme importovali komponentu do externej DLL umožní ďalej v externej DLL používať objekt **AOSMTMP.Mail**. Po vytvorení inštancie objektu **oSmtmp** je možné používať metódy jeho rozhrania, tak ako je to uvedené na Obrázku 6:

```

BOOL CExample1App::Send(BSTR From,BSTR To,BSTR Subject,BSTR Body)
{
    ::CoInitialize( NULL );

    IMailPtr oSmtmp = NULL;
    oSmtmp.CreateInstance("ASMTMP.Mail");

    oSmtmp->ServerAddr = _bstr_t( "" );
    oSmtmp->FromAddr = _bstr_t(From);
    oSmtmp->AddRecipient( _bstr_t(""),_bstr_t(To), 0 );
    oSmtmp->Subject = _bstr_t(Subject);
    oSmtmp->BodyText = _bstr_t(Body);

    if (oSmtmp->SendMail() == 0)
        //cout << "Message delivered!" << endl;
        return true;
    else
        //cout << (const char*)(oSmtmp->GetLastErrDescription()) << endl;
        return false;
}

```

Obrázok 6: Odoslanie e-mail z externej DLL prostredníctvom komponenty ActiveX (ASNMTMP SMTP COMPONENT).

Do *Example1* príkladovej externej DLL je možné pridávať nové funkcie, ktoré rozšíria už existujúcu funkcionálnosť. Takouto pridanou funkciou môže byť napríklad funkcia **CExample1App::Send**, pre odoslanie e-mail prostredníctvom komponenty **ASNMTMP SMTP** (pozri na Obrázku 6 hore). Novo pridaná funkcia **Send** má 4 parametre (*From*, *To*, *Subject*, *Body*). V tejto funkcii sa hodnoty jednotlivých parametrov priradujú do metód logicky korešpondujúcich s názvami metód z komponenty **ASNMTMP SMTP**.

Výsledkom priradenia hodnôt metódam komponenty sa zostaví požadovaný e-mail a zavolaním `oSmtmp->SendMail()` sa daný e-mail s požadovaným obsahom odošle na zadanú adresu.

Pri vytváraní externej DLL doporučujeme vychádzať z toho, čo je už spravené. Inštalčný balíček **OpcDbGateway** obsahuje aj zdrojové kódy ukážkovej **externej DLL** nazvanej **Example1**.

Tak ako sa môžu do preddefinovaných funkcií vzorovej *Example1* externej DLL pridávať nové funkcie, rovnako sa môžu aj povodne preddefinované funkcie modifikovať a rozširovať. Toto môžeme spraviť aj v prípade preddefinovanej funkcie **DoProcessIO**. V našom príklade do tela tejto funkcie pridáme počítadlo cyklov. Ak potom počítadlo dosiahne určitý počet cyklov, o ktorom chcem byť informovaný napr. 10, tak je zavolaná funkcia **theApp.Send** a následne odoslaný e-mail.

Vo vzorovom príklade sa nachádza funkcia **GetCountOfIO** definujúca počet vstupov=4 a rovnako aj počet výstupov=4. Aby sme však mohli zadávať ďalšie vstupné hodnoty z aplikácie **OpcDbGateway** (napr. e-mail adresy *From* a *To*) je potrebné modifikovať počet vstupov na 6. Potom je možné odovzdať parametre do funkcie **theApp.Send** prostredníctvom hodnôt definovaných ako memory operandy z **OpcDbGateway** nasledovným spôsobom:

```
theApp.Send(_bstr_t(lpInputs[4]),
_bstr_t(lpInputs[5]),
```

V položke poľa *lpInputs[4]* sa nachádza adresa *From* a v položke poľa *lpInputs[5]* sa nachádza cieľová adresa *To*.

Vstupné parametre, ktoré prichádzajú do externej DLL z aplikácie OpcDbGateway musia byť tiež správne zadefinované v konfiguračnom projekte pre OpcDbGateway. Pre vstupné premenné, ktoré do externej DLL prenášajú adresy *From* (memory operand I5_sender) a *To* (memory I6_receiver) je potrebné do konfigurácie pridať takzvané **memory operandy**. Požadovaná hodnota pre jednotlivé memory operandy je nastavovaná pomocou **funkčných blokov** vid' Obrázok 7 a Obrázok 8.

Name	Sim...	DataType	VarAddr
I1	No	LONG	201
I2	No	LONG	202
I3	No	LONG	203
I4	No	LONG	204
I5_sender	No	STRING	205
I6_receiver	No	STRING	206
O1	No	LONG	207
O2	No	LONG	208
O3	No	LONG	209
O4	No	LONG	210
Result	No	BOOL	200

Obrázok 7: Memory operandy pre definovanie e-mailových adries boli pridané do zoznamu.

Name: 60 I6=e-mail_receiver

Description:

Line Number: 60 Operation: SET

Output type: Mem DB Output: DllExampleData/I6_receiver

Input1 type: Mem DB Const Input 1: e-mail_receiver_address

Obrázok 8: Priradenie e-mailovej adresy prostredníctvom funkčného bloku.

Downloads

V tejto časti je zoznam liniek na dôležité dokumenty, ktoré súvisia s aplikáciou OpcDbGateway.

OpcDbGateway documentation (User's Guide)

http://www.saeautom.sk/download/help/opcdbgateway_en.pdf

Použitie našich produktov

http://www.saeautom.sk/download/products_solutions_sk.pdf

OpcDbGateway - Powerfull programm system for PC

http://www.saeautom.sk/download/opcdbgateway_en.pdf

Historical Trend Wizard for OpcDbGateway

http://www.saeautom.sk/download/gwtrends_en.pdf

Configuring OPC and DCOM for OPC server and OPC client applications from SAE – Automation, Ltd.

http://www.saeautom.sk/download/dcom_config.pdf

Configuring DCOM for using OPC UA COM Wrapper with OPC servers from SAE – Automation, Ltd.

<http://www.saeautom.sk/download/opcuaforsaeproducts.pdf>

Internet browser based OPC client

http://www.saeautom.sk/download/opc_explorer.pdf

SAEAUT SNMP OPC Server documentation (User's Guide)

http://www.saeautom.sk/download/help/saeaut_snmp_opc_server_en.pdf

Elegant solution for the management of computer network

http://www.saeautom.sk/download/snmpopcserver_en.pdf

Monitoring of network infrastructure

http://www.saeautom.sk/download/monitoring_en.pdf

Internet browser based OPC client

http://www.saeautom.sk/download/opc_explorer.pdf

Disclaimer

The information contained in these pages is based on our testing and practices experience. SAE – Automation, Ltd. and the authors of this document assume no responsibility for direct, indirect, or consequential liability for its accuracy or suitability for a user's particular application. The reader is responsible for proper application to their particular situation.

Príloha: Zdrojový kód externej DLL, ktorá pôsobí ako sprostredkovateľ medzi inou ActiveX komponentou a aplikáciou OpcDbGateway

```
/*
**
** Copyright: SAE-Automation,s.r.o.Nova Dubnica, Slovakia,
**
**
** Project: OpcDbGateway.exe          Project name:  OpcDbGateway
**
**
** @doc EXTERNAL
** @module Example1.h | Example external DLL.
**
**
** @info
** Version          : 1.0<nl>
** Compiler         : MSVC 7.0<nl>
** Author           : SAE-Automation,s.r.o.<nl>
**
** @history
** 24.01.2008      Initial version<nl>
**
**
#pragma once

#ifndef __AFXWIN_H__
    #error include 'stdafx.h' before including this file for PCH
#endif

#include "resource.h"          // main symbols

// CExample1App
// See Example1.cpp for the implementation of this class
//

class CExample1App : public CWinApp
{
public:
    CExample1App();
    BOOL Send(BSTR From,BSTR To,BSTR Subject,BSTR Body);
}
```



```

// Overrides
public:
    virtual BOOL InitInstance();

    DECLARE_MESSAGE_MAP()

};

```

Dodatok 1: Example1.h

```

/*****
**
** Copyright:          SAE-Automation,s.r.o.Nova Dubnica, Slovakia,
**
**
**
** Project: OpcDbGateway.exe          Project name: OpcDbGateway
**
**
**
** @doc EXTERNAL
** @module Example1.cpp | Example external DLL.
**
**
** @info
** Version            : 1.0<nl>
** Compiler           : MSVC 7.0<nl>
** Author             : SAE-Automation,s.r.o.<nl>
**
**
** @history
** 24.01.2008 Initial version<nl>
**
**
*****/

#include "stdafx.h"
#include "Example1.h"
#include "ExternalDllInterface.h"

#include <comdef.h>
#include <iostream>
#import "C:\Program Files\AdminSystem.NET\ANSMTTP\AOSMTTP.dll"
//rename_namespace("AOSMTPLib")
using namespace AOSMTPLib;
using namespace std;

#ifdef _DEBUG
#define new DEBUG_NEW
#endif

//
// Note!
//
// If this DLL is dynamically linked against the MFC
// DLLs, any functions exported from this DLL which
// call into MFC must have the AFX_MANAGE_STATE macro
// added at the very beginning of the function.
//
// For example:
//
// extern "C" BOOL PASCAL EXPORT ExportedFunction()

```

```

//      {
//          AFX_MANAGE_STATE(AfxGetStaticModuleState());
//          // normal function body here
//      }
//
//      It is very important that this macro appear in each
//      function, prior to any calls into MFC. This means that
//      it must appear as the first statement within the
//      function, even before any object variable declarations
//      as their constructors may generate calls into the MFC
//      DLL.
//
//      Please see MFC Technical Notes 33 and 58 for additional
//      details.
//
// CExample1App

BEGIN_MESSAGE_MAP(CExample1App, CWinApp)
END_MESSAGE_MAP()

// CExample1App construction
CExample1App::CExample1App()
{
    // TODO: add construction code here,
    // Place all significant initialization in InitInstance
}

// The one and only CExample1App object
CExample1App theApp;

// CExample1App initialization
BOOL CExample1App::InitInstance()
{
    CWinApp::InitInstance();

    return TRUE;
}

/*****
*****
// @mfunc    DWORD | GetDescription | The function retrieves a DLL product name as
string.
//
// @param    LPCTSTR          | lpReturnedString | [out] Pointer to the buffer
that receives the retrieved string.
// @param    DWORD           | dwSize          | [in] Size of the buffer
pointed to by the lpReturnedString parameter, in TCHARs
//
// @rdesc    The return value is the number of characters copied to the buffer, not
including the terminating null character.
//
//          If the supplied destination buffer is too small to hold the requested
string,
//          the string is truncated and followed by a null character, and the
return value is equal to nSize minus one.
//
// @comm     The function retrieves a DLL product name as string.
//
//          The function will be called from OpcDbGateway configurator module.
//
// @history  24.01.2008 13:11, created by SAE-Automation,s.r.o.
/*****
*****

```

```

DllExport DWORD WINAPI GetProductName(LPTSTR lpReturnedString, DWORD dwSize)
{
    // TODO: Add your product name

    // EXAMPLE: "1.0.0.0"
    // BEGIN {
    const TCHAR* ReturnedString = _T("External DLL sample");

    if( _tcslen(ReturnedString) < dwSize ) {
        _tcscpy(lpReturnedString, ReturnedString);
    }
    else {

        if( !dwSize )
            return 0;

        _tcsncpy(lpReturnedString, ReturnedString, dwSize - 1);
    }
    // } END

    return (DWORD)_tcslen(lpReturnedString);
}

/*****
*****
// @mfunc    DWORD | GetDescription | The function retrieves a DLL product version
as string.
//
// @param    LPTSTR          | lpReturnedString | [out] Pointer to the buffer
that receives the retrieved string.
// @param    DWORD          | dwSize          | [in] Size of the buffer
pointed to by the lpReturnedString parameter, in TCHARs
//
// @rdesc    The return value is the number of characters copied to the buffer, not
including the terminating null character.
//
//          If the supplied destination buffer is too small to hold the requested
string,
//          the string is truncated and followed by a null character, and the
return value is equal to nSize minus one.
//
// @comm     The function retrieves a DLL product version as string.
//
//          The function will be called from OpcDbGateway configurator module.
//
// @history  24.01.2008 13:11, created by SAE-Automation,s.r.o.
*****/
DllExport DWORD WINAPI GetProductVersion(LPTSTR lpReturnedString, DWORD dwSize)
{
    // TODO: Add your product version

    // EXAMPLE: "1.0.0.0"
    // BEGIN {
    const TCHAR* ReturnedString = _T("1.0.0.0");

    if( _tcslen(ReturnedString) < dwSize ) {
        _tcscpy(lpReturnedString, ReturnedString);
    }
    else {

        if( !dwSize )
            return 0;

        _tcsncpy(lpReturnedString, ReturnedString, dwSize - 1);
    }
}

```

```

        // } END

        return (DWORD)_tcslen(lpReturnedString);
}

/*****
*****
// @mfunc  DWORD | GetDescription | The function retrieves a DLL company name as
string.
//
// @param  LPTSTR          | lpReturnedString | [out] Pointer to the buffer
that receives the retrieved string.
// @param  DWORD          | dwSize          | [in] Size of the buffer
pointed to by the lpReturnedString parameter, in TCHARs
//
// @rdesc  The return value is the number of characters copied to the buffer, not
including the terminating null character.
//
//          If the supplied destination buffer is too small to hold the requested
string,
//          the string is truncated and followed by a null character, and the
return value is equal to nSize minus one.
//
// @comm   The function retrieves a DLL company name as string.
//
//          The function will be called from OpcDbGateway configurator module.
//
// @history 24.01.2008 13:11, created by SAE-Automation,s.r.o.
/*****
*****
DllExport DWORD WINAPI GetCompanyName(LPTSTR lpReturnedString, DWORD dwSize)
{

    // TODO: Add your company name

    // EXAMPLE: "SAE - Automation, s.r.o."
    // BEGIN {
    const TCHAR* ReturnedString = _T("SAE - Automation, s.r.o.");

    if( _tcslen(ReturnedString) < dwSize ) {
        _tcsncpy(lpReturnedString, ReturnedString);
    }
    else {

        if( !dwSize )
            return 0;

        _tcsncpy(lpReturnedString, ReturnedString, dwSize - 1);
    }
    // } END

    return (DWORD)_tcslen(lpReturnedString);
}

/*****
*****
// @mfunc  DWORD | GetDescription | The function retrieves a DLL copyright as
string.
//
// @param  LPTSTR          | lpReturnedString | [out] Pointer to the buffer
that receives the retrieved string.
// @param  DWORD          | dwSize          | [in] Size of the buffer
pointed to by the lpReturnedString parameter, in TCHARs
//
// @rdesc  The return value is the number of characters copied to the buffer, not
including the terminating null character.

```

```

//
//      If the supplied destination buffer is too small to hold the requested
string,
//      the string is truncated and followed by a null character, and the
return value is equal to nSize minus one.
//
// @comm   The function retrieves a DLL copyright as string.
//
//      The function will be called from OpcDbGateway configurator module.
//
// @history 24.01.2008 13:11, created by SAE-Automation,s.r.o.
//*****
//*****
DllExport DWORD WINAPI GetLegalCopyright(LPTSTR lpReturnedString, DWORD dwSize)
{
    // TODO: Add your Copyright

    // EXAMPLE: "Copyright © 2002-2007 SAE - Automation, s.r.o. All rights
reserved."
    // BEGIN {
    const TCHAR* ReturnedString = _T("Copyright © 2002-2010 SAE - Automation,
s.r.o. All rights reserved.");

    if( _tcslen(ReturnedString) < dwSize ) {
        _tcsncpy(lpReturnedString, ReturnedString);
    }
    else {
        if( !dwSize )
            return 0;

        _tcsncpy(lpReturnedString, ReturnedString, dwSize - 1);
    }
    // } END

    return (DWORD)_tcslen(lpReturnedString);
}

//*****
//*****
// @mfunc   DWORD | GetDescription | The function retrieves a DLL description as
string.
//
// @parm    LPTSTR          | lpReturnedString | [out]Pointer to the buffer
that receives the retrieved string.
// @parm    DWORD          | dwSize          | [in] Size of the buffer
pointed to by the lpReturnedString parameter, in TCHARs
//
// @rdesc   The return value is the number of characters copied to the buffer, not
including the terminating null character.
//
//      If the supplied destination buffer is too small to hold the requested
string,
//      the string is truncated and followed by a null character, and the
return value is equal to nSize minus one.
//
// @comm   The function retrieves a DLL description as string.
//
//      The function will be called from OpcDbGateway configurator module.
//
// @history 24.01.2008 13:11, created by SAE-Automation,s.r.o.
//*****
//*****
DllExport DWORD WINAPI GetDescription(LPTSTR lpReturnedString, DWORD dwSize)
{

```

```

// TODO: Add your external DLL description

// EXAMPLE: "Output[i] = Inputs[i] + 1"
// BEGIN {
const TCHAR* ReturnedString = _T("Output[i] = Inputs[i] + 1");

if( _tcslen(ReturnedString) < dwSize ) {
    _tcscpy(lpReturnedString, ReturnedString);
}
else {

    if( !dwSize )
        return 0;

    _tcsncpy(lpReturnedString, ReturnedString, dwSize - 1);
}
// } END

return (DWORD)_tcslen(lpReturnedString);
}

/*****
*****
// @mfunc void | GetCountOfIO | The function retrieves a specified count of
inputs/outputs.
//
// @param LPWORD | lpInputs | [in, out] Pointer to count of input
values.
// @param LPWORD | lpOutputs | [in, out] Pointer to count of output
values.
//
// @rdesc Returns always void.
//
// @comm The function retrieves a specified count of inputs/outputs.
//
// The function will be called from OpcDbGateway configurator module and
runtime module.
//
// @history 24.01.2008 13:11, created by SAE-Automation,s.r.o.
/*****
*****
DllExport void WINAPI GetCountOfIO(LPWORD lpInputs, LPWORD lpOutputs)
{

    // TODO: Set your count of inputs and outputs

    // EXAMPLE: Inputs=4, Outputs=4
    // BEGIN {
    *lpInputs = 6; // TODO: Set your count of inputs
    *lpOutputs = 4; // TODO: Set your count of outputs
    // } END

    return;
}

/*****
*****
// @mfunc bool | DoProcessIO | The function processes all values from input
buffer and retrieves the processed values in output buffer.
//
// @param const CComVariant* | lpInputs | [in] Pointer to the buffer that
includes input values.
// @param WORD | wInCnt | [in] Size of the buffer pointed to by
the lpInputs parameter, in CComVariant
// @param CComVariant* | lpOutputs | [out] Pointer to the buffer that
recieves output values.

```

```

// @param WORD | wOutCnt | [in] Size of the buffer pointed to by
the lpOutputs parameter, in CComVariant
//
// @rdesc Returns one of the following values:
//
// @flag true | Successfully
// @flag false | Error value otherwise
//
// @comm The function processes all values from input buffer and retrieves the
processed values in output buffer.
//
// The function will be called from OpcDbGateway runtime module as
Function Blocks/Command.
//
// @history 24.01.2008 13:11, created by SAE-Automation,s.r.o.
//*****
DllExport bool WINAPI DoProcessIO(const CComVariant* lpInputs, WORD wInCnt,
                                  CComVariant*
lpOutputs, WORD wOutCnt)
{
    // TODO: Add your functionality

    // EXAMPLE: Output[i] = Inputs[i] + 1
    // BEGIN {
    long counter = 0;

    for(DWORD i=0; i < wOutCnt; i++)
    {
        CComVariant value = lpInputs[i];
        value.ChangeType(VT_I4);

        ATLTRACE("\nDLL: i:%d VARTYPE=%d Value=%d", i, value.vt, value.lVal);
        lpOutputs[i] = value.lVal + 1;
        counter = value.lVal;
    }

    if (counter % 10 == 0)
    {
        theApp.Send(_bstr_t(lpInputs[4]),
                    _bstr_t(lpInputs[5]),
                    _bstr_t("Test mail"),
                    _bstr_t("Hello OPCDBGATEWAY of SAE Automation want to
remain you that time of processing next 10 cycles is yet fullfilled ...."));
    }
    // } END

    return 1; // OK
}

BOOL CExample1App::Send(BSTR From,BSTR To,BSTR Subject,BSTR Body)
{
    ::CoInitialize( NULL );
    IMailPtr oSmtP = NULL;
    oSmtP.CreateInstance("AOSMTP.Mail");

```

```
oSmtp->ServerAddr = _bstr_t( "" );  
//If you don't have SMTP server, use the following code  
//send email via DNS lookup, ANSMTP lookups SMTP server automatically.  
//oSmtp->ServerAddr = _bstr_t("");  
  
oSmtp->FromAddr = _bstr_t(From);  
oSmtp->AddRecipient( _bstr_t(""),_bstr_t(To), 0 );  
  
oSmtp->Subject = _bstr_t(Subject);  
oSmtp->BodyText = _bstr_t(Body);  
  
if( oSmtp->SendMail() == 0 )  
    //cout << "Message delivered!" << endl;  
return true;  
else  
    //cout << (const char*)(oSmtp->GetLastErrDescription()) << endl;  
    return false;  
}
```