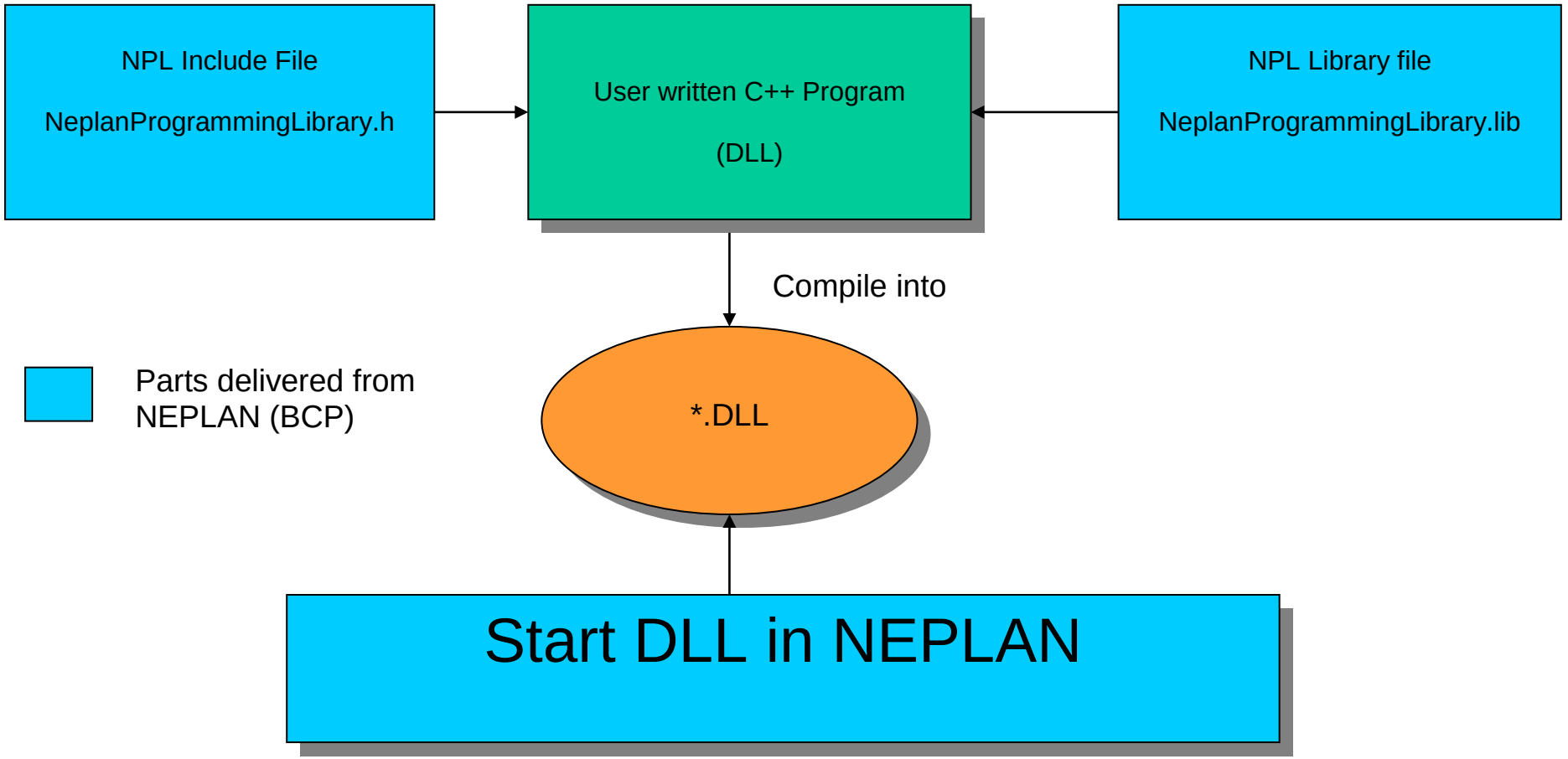
NPL - NEPLAN Programming Library

Contents

1) Introduction

The NPL contains a set of C/C++ library functions which allows to access directly NEPLAN project files (*.nepprj) through a user written C/C++ program. The functions allows to execute any analysis function like load flow analysis, transient stability analysis, short circuit analysis, etc. Furthermore it allows to access and modify any variable of all element types (e.g. length of a line, short-circuit voltage of a transformer. etc.). It is also possible to modify any data of the predefined regulators (e.g. exciters) and of all function blocks.
With this set of library function it is very easy to manipulate NEPLAN projects through a normal C/C++ program. The user has the possibility to add and remove elements from the network with library functions. The user does not need to know and learn any other batch programming language. Any user who knows how to write a C/C++ program with the Microsoft MCF compiler Visual .NET 2005 can easily define his own “batch program”. It even allows him to make his own new analysis modules (e.g. DACF module, voltage stability module etc.) or just to make a batch file which plots all diagrams in a project. The NPL allows you also to access the results. All the user needs is to include the NEPLAN programming header file (NeplanProgrammingLibrary.h) into his project and link it together with the NEPLAN programming library (NeplanProgrammingLibrary.lib). Then he compiles his C/C++ project into a dynamic link library (*.Dll file). The *dll file can then be started from NEPLAN with menu item “File->Run NPL…” or as an alternative NEPLAN may be started in batch mode with the /b option (e.g. “NEPLAN /npl test.dll”).

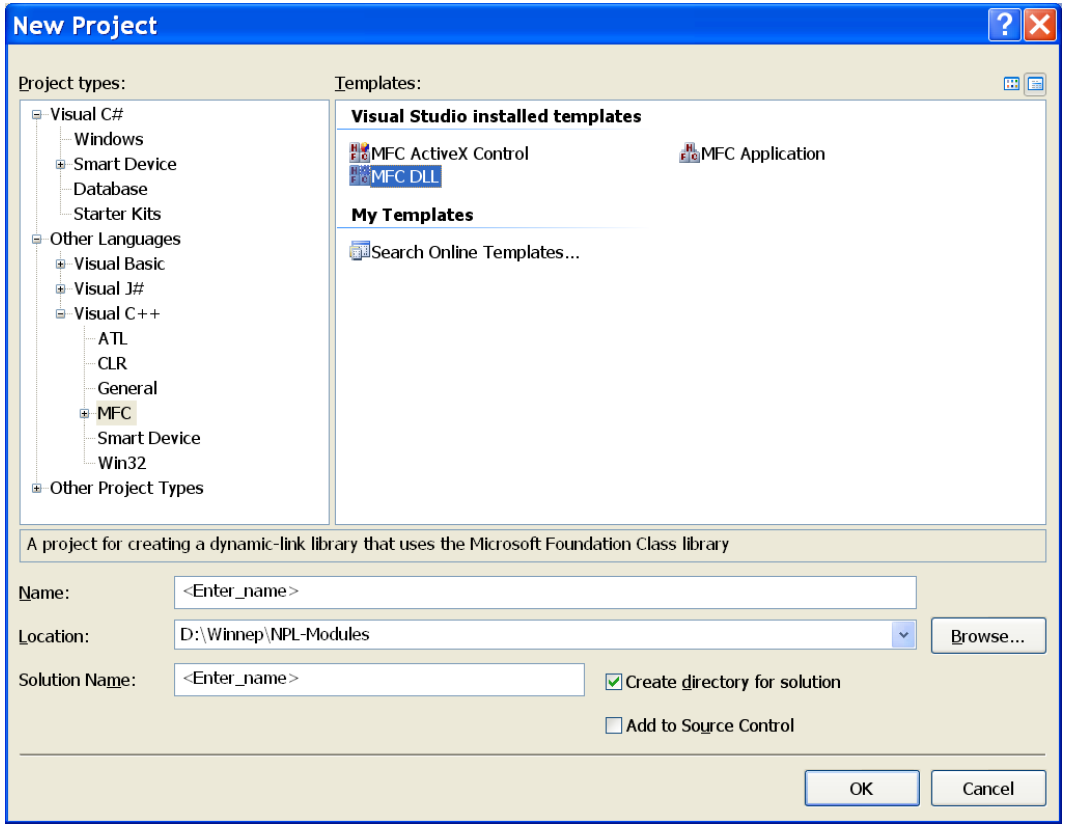
NPL Overview



2) Build the first C/C++ example

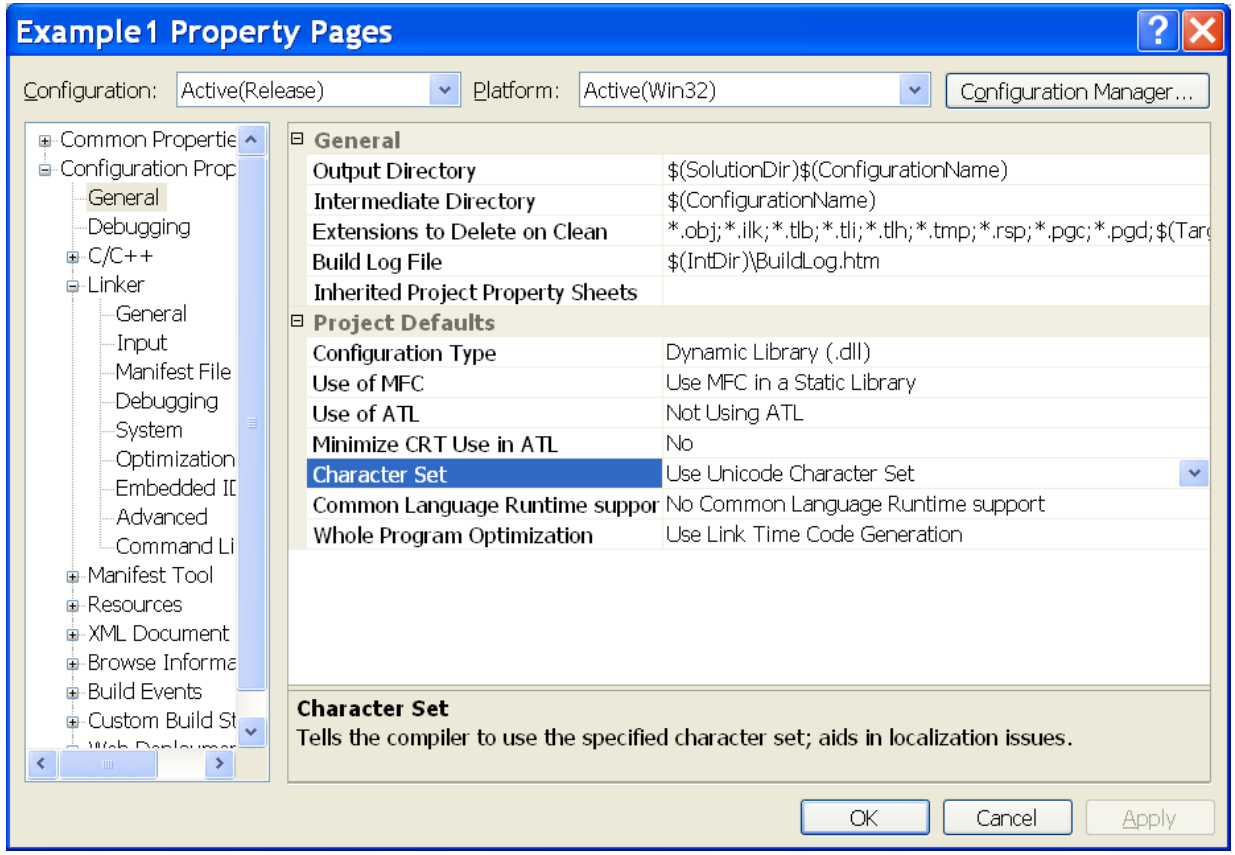
The following steps must be considered to build a user defined NEPLAN dynamic link library (*.dll):

1) Start MCF compiler Visual .NET 2005
2) Create a new project (choose MFC DLL)

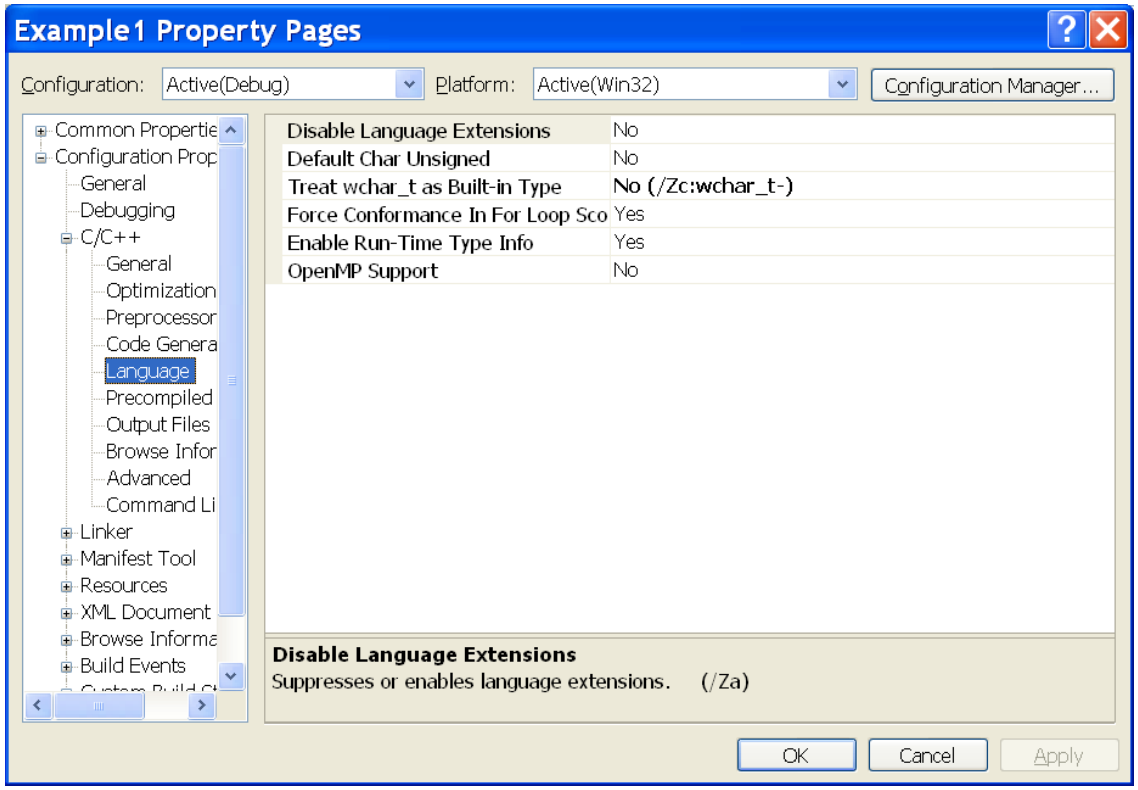


3) In the General properties choose:

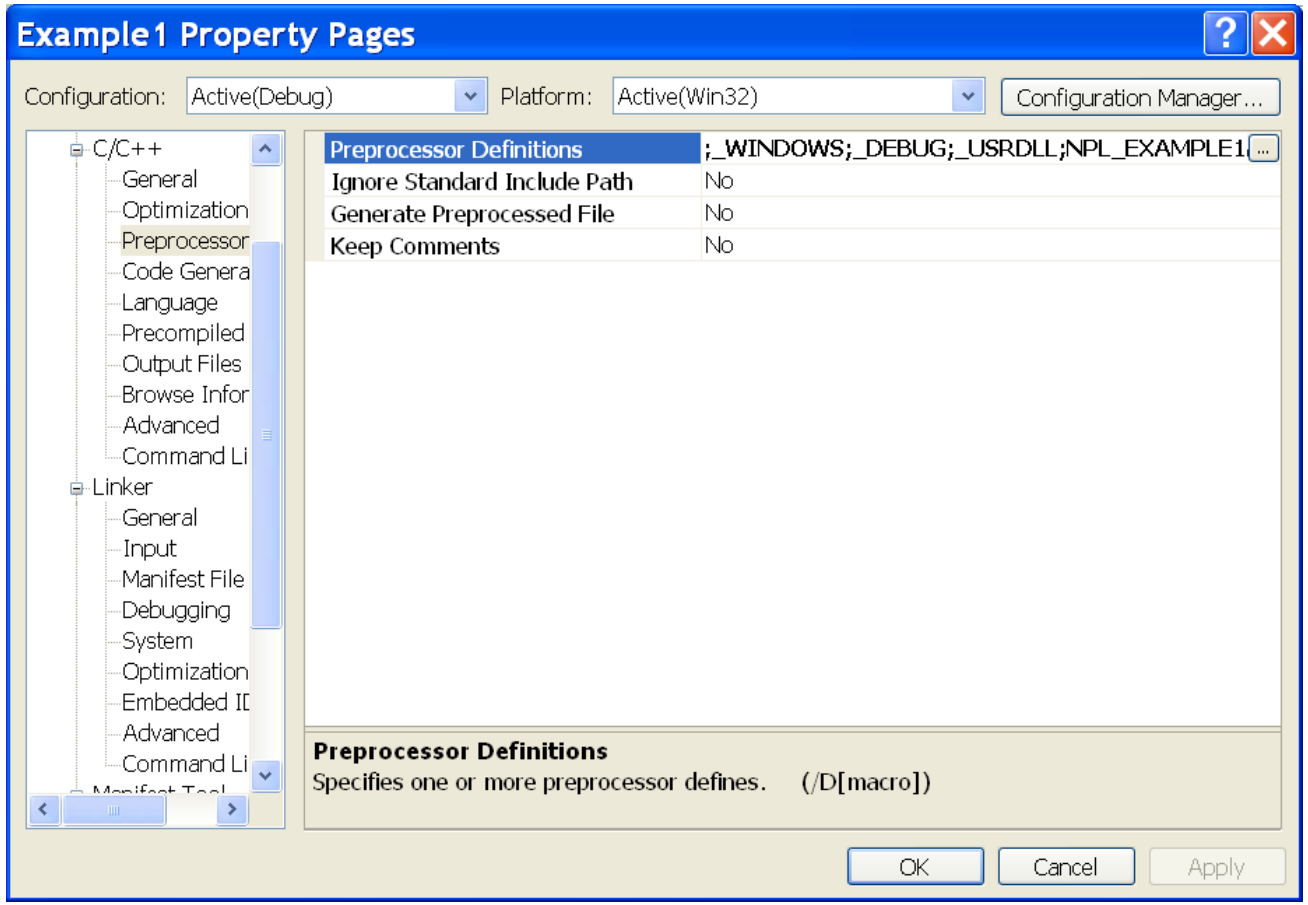
- Use Standard Windows Libraries
- **Use Unicode Character Set** (this is very important!)



4) Set the “wachr_t Type” in the “Language” tab “**No (/Zc:wchar_t-)**” (this is very important!)



5) The definition file (*.def, name depends on the application name) of the dll must be included in the Preprocessor property:



6) In the header file of the main *.cpp file (e.g. “Exampl1.h”) define the export function “RunNeplanScript” as follows:

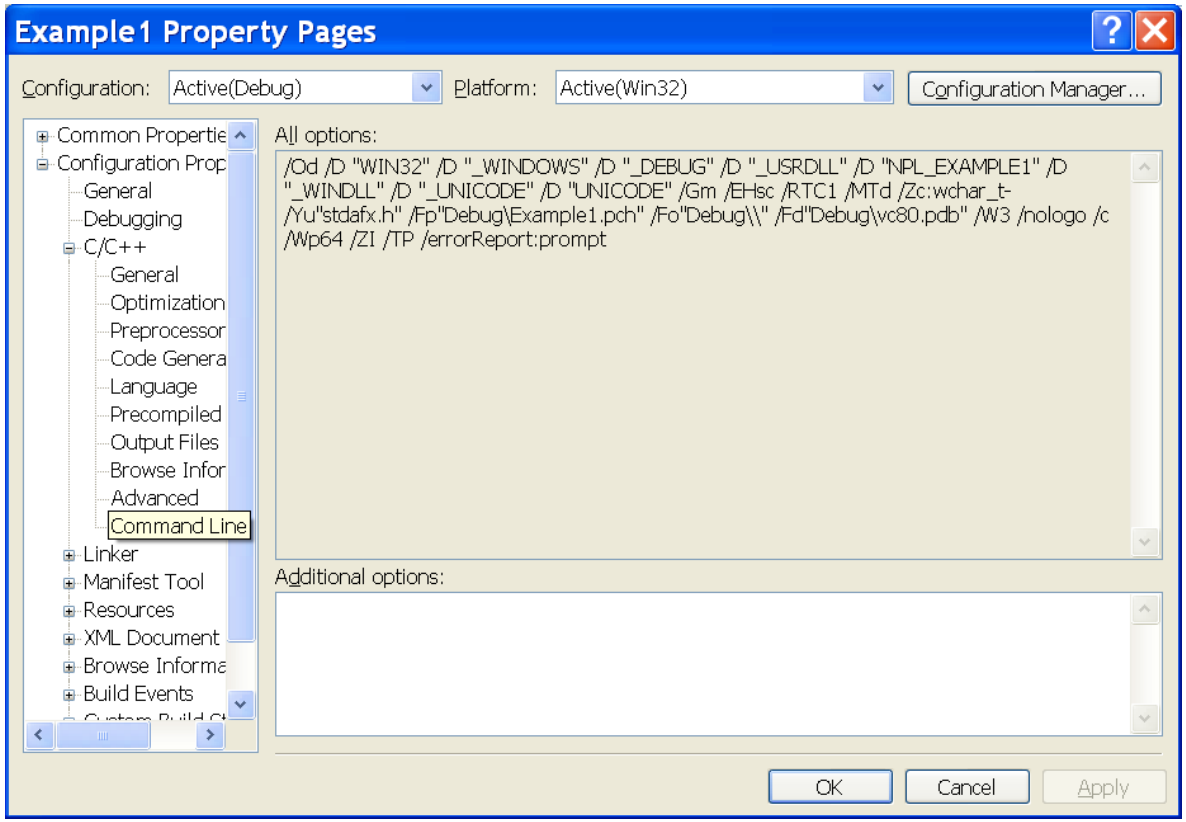
```
#ifdef NPL_EXAMPLE1
#define NPL_EXAMPL1_API __declspec(dllexport)
#else
#define NPL_EXAMPL1_API __declspec(dllimport)
#endif


NPL_EXAMPL1_API BOOL RunNeplanScript();
```

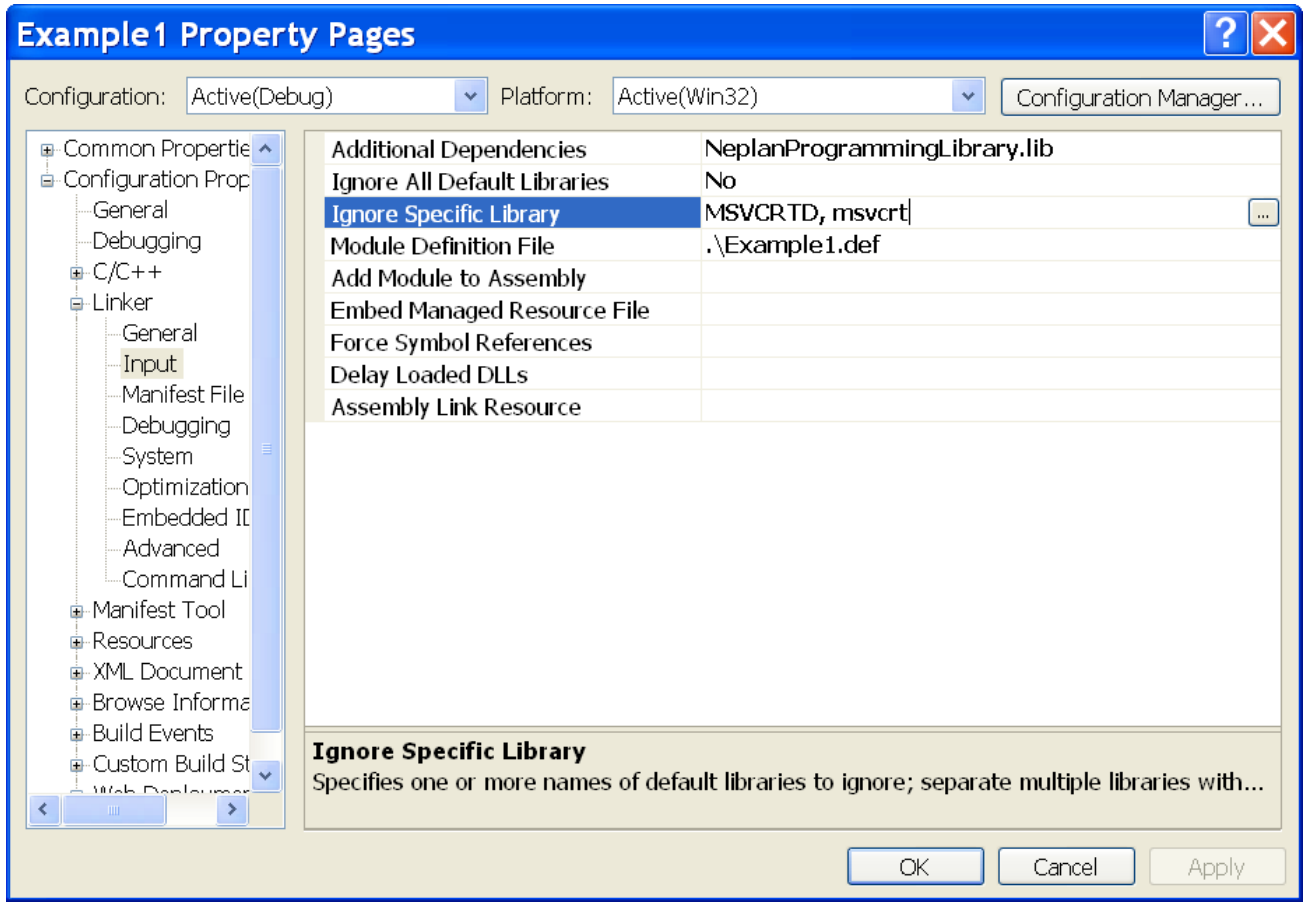
7) In the main *.cpp file (e.g. “Exampl1.cpp”) write the “RunNeplanScript” as follows:

```
////////////////////////////////
// RunNeplanScript is the
// main entry to run dll
////////////////////////////////
NPL_EXAMPL1_API BOOL RunNeplanScript()
{
  BOOL bRunOk = TRUE;
  //define here your NEPLAN application
 }
```

8) Below see the command line for the C++ compiler:



9) “Linker Input” property page : The “NeplanProgrammingLibrary.lib” file must be included and some specific libraries must be ignored



10) In the module definition File (e.g. Example1.def) the export function“RunNeplanScript” must be defined

```
; Example1.def : Declares the module parameters for the DLL.

LIBRARY      "Example1"

EXPORTS
    ; Explicit exports can go here
     RunNeplanScript  @1
```

11) Compile the dynamic link library (Example1.dll) and start it from NEPLAN with menu item “File->Run NPL…”

Below is a short C/C++ example which shows how to change the line length and then run a reliability analysis. First the NEPLAN project has to be opened, then the line has to be found in the project. After that the length can be changed and a reliability analysis can be started.

```
////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////
// Author: Giatgen Cott,   BCP Busarello + Cott + Partner AG
// Date:   25. September 2006
//
// Build the first C/C++ example with NPL
// The example shows how to change the length of a line and run a reliability analysis
// Use unicode declarations, e.g. TCHAR
////////////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////
#include "stdafx.h"
#include <stdio.h>
#include "Main.h"
#include "NepModuleDef.h"                //->include Module Definition file
#include "NeplanProgrammingLibrary.h"        //->include NEPLAN NPL file

NPL_EXAMPL1_API BOOL RunNeplanScript()
{
     TCHAR PathFileName[400];
     TCHAR *ProjectName = _T("NPL-Demo-Ele.nepprj");
     static TCHAR *DataExampleDirectory = _T("C:\\Neplan\\ExampleData\\");
     swprintf(PathFileName, _T("%s%s"), DataExampleDirectory, ProjectName);
     //
     // Open Project the project using the NEPLAN library function "OpenNeplanProject"
```

```
    BOOL bOpen = OpenNeplanProject(PathFileName);
    if (!bOpen)
       return FALSE;

    // Get the element ID of a line from the project
    // using the NEPLAN library function "GetElementByName"
    unsigned long ElementID=0;
    GetElementByName(_T("LINE"),_T("Line-xyz"), ElementID);

     if (ElementID > 0)
    {
        // change line length to 0.5 km using the NEPLAN library function
        //  "SetParameterDouble" and using the keyword "Length" to access the
        //  line length of the element with ID = ElementID
        SetParameterDouble(ElementID, _T("Length"), 0.5);
    }

    // run a relaibility analysis with the NPL function  "RunAnalysis "
    // using the keyword "BCP_SELECTED_MODULE_RELIABILITY " for
    // reliability analysis
     RunAnalysis(BCP_SELECTED_MODULE_RELIABILITY);

    //Close Project the project using the NPL library function "CloseCurrentProject"
    CloseCurrentProject();
}
```

3) Description of the NEPLAN library function

Please note that the UNICODE convention has to be used for character variables in the C/C++ program.

Following function are available in the NEPLAN Programming Library (NLP):

BOOL ShowReport()	Shows the report file of a NEPLAN NPL run. Returns TRUE if report can be shown
BOOL OpenNeplanProject(TCHAR* sFileName)	Opens a NEPLAN project. Returns TRUE if the project can be opened successfully
BOOL CloseCurrentProject()	Closes the currently opened project an returns TRUE on success.
BOOL SaveCurrentProject(TCHAR* wcFileName=NULL)	Saves the open project to a file. Returns TRUE on success.
BOOL GetParameterDouble(unsigned long lID,TCHAR* sParameter, double& dValue)	Gets a parameter of type “double” of an element. Input: lID = ID of the element type, sParameter = name of the parameter of the element, Output: dValue = value of the parameter. Returns TRUE on success. For a list of element of parameter types see appendix (List of element parameter types).
BOOL GetParameterLong(unsigned long lID,TCHAR* sParameter, long& lValue);	Same as “GetParameterDouble” but for “long” type value
BOOL GetParameterInt(unsigned long lID,TCHAR* wcParameter, int& nValue)	Same as “GetParameterDouble” but for “int” type value
BOOL GetParameterBool(unsigned long lID,TCHAR* wcParameter, bool& bValue)	Same as “GetParameterDouble” but for “bool” type value
BOOL GetParameterString(unsigned long lID,TCHAR* wcParameter, LPTSTR lpValue)	Same as “GetParameterDouble” but for “char” type value
BOOL SetParameterDouble(unsigned long lID,TCHAR*	Sets the value of an element parameter of type “double.

sParameter, double dValue)	Input: lID = ID of the element, sParameter = name of the parameter of the element, dValue = value of the parameter to be set. Returns TRUE on success. For a list of parameter types see appendix (List of element parameter types).
BOOL SetParameterLong(unsigned long lID,TCHAR* sParameter, long lValue);	Same as “SetParameterDouble” but for “long” type value
BOOL SetParameterInt(unsigned long lID,TCHAR* wcParameter, int nValue)	Same as “SetParameterDouble” but for “int” type value
BOOL SetParameterBool(unsigned long lID,TCHAR* wcParameter, bool bValue)	Same as “SetParameterDouble” but for “bool” type value
BOOL SetParameterString(unsigned long lID,TCHAR* wcParameter, TCHAR* wcValue)	Same as “SetParameterDouble” but for “char” type value
BOOL SetUserDefinedParameterDouble(unsigned long lID,TCHAR* wcParameter, int LF_DYN_SC, double dValue);	Sets the value of parameter a user defined of type “double. Input: lID = ID of the element, sParameter = name of the user defined parameter. LF_DYN_SC = model type (0=LF, 1=DYN, 2=SC) dValue = value of the parameter to be set. Returns TRUE on success.
BOOL GetUserDefinedParameterDouble(unsigned long lID,TCHAR* wcParameter, int LF_DYN_SC, double& dValue);	Gets a parameter of type “double” of a user defined element. Input: lID = ID of the element type, LF_DYN_SC = model type (0=LF, 1=DYN, 2=SC) sParameter = name of the user defined Output: dValue = value of the parameter. Returns TRUE on success.
BOOL GetCalcParameterDouble(TCHAR* sCalcParameter, TCHAR* sParameter, double& dValue);	Gets a “double” type calculation parameter. Input: sCalcParameter = name of the calculation parameter type (e.g. for load flow: _T("PARAM_ELEC")),

	sParameter = name of the calculation paramter Output: dValue = value of the parameter. Returns TRUE on success. For a list of parameter types see appendix (List of calculation parameter types).
BOOL GetCalcParameterInt(TCHAR* sCalcParameter, TCHAR* sParameter, int& nValue);	Same as “GetCalcParameterDouble” but for “int” type value.
BOOL GetCalcParameterBool(TCHAR* sCalcParameter, TCHAR* sParameter, bool& bValue);	Same as “GetCalcParameterDouble” but for “bool” type value.
BOOL GetCalcParameterString(TCHAR* sCalcParameter, TCHAR* sParameter, TCHAR** sValue);	Same as “GetCalcParameterDouble” but for “char” type value.
BOOL SetCalcParameterDouble(TCHAR* sCalcParameter, TCHAR* sParameter, double dValue);	Sets a “double” type calculation parameter. Input: sCalcParameter = name of the calculation parameter type (e.g. for load flow: _T("PARAM_ELEC")), sParameter = name of the calculation paramter dValue = value of the parameter. Returns TRUE on success. For a list of parameter types see appendix (List of calculation parameter types).
BOOL SetCalcParameterInt(TCHAR* sCalcParameter, TCHAR* sParameter, int nValue);	Same as “GetCalcParameterDouble” but for “int” type value.
BOOL SetCalcParameterBool(TCHAR* sCalcParameter, TCHAR* sParameter, bool bValue);	Same as “GetCalcParameterDouble” but for “bool” type value.
BOOL SetCalcParameterString(TCHAR* sCalcParameter, TCHAR* sParameter, TCHAR* sValue);	Same as “GetCalcParameterDouble” but for “char” type value.
BOOL SetFaultedNode(unsigned long lID);	Sets a faulted node for a short circuit calculation. Input: lID = ID of the node.

	Returns TRUE on success.
BOOL RemoveFaultedNode(unsigned long lID);	Removes a faulted node for a short circuit calculation. Input: lID = ID of the node. Returns TRUE on success
BOOL SetFaultOnLine(unsigned long lID, double dDistance);	Sets a faulted line for a short circuit calculation. Input: lID = ID of the line, dDistance= distance form the “From Node” where the short circuit will be set. Returns TRUE on success.
BOOL RemoveFaultOnLine(unsigned long lID);	Removes a faulted line for a short circuit calculation. Input: lID = ID of the node. Returns TRUE on success.
void GetElementByName(TCHAR* wcType,TCHAR* wcName,unsigned long& lElementID);	Gets the internal element ID with the element name as input. This function will often be used before setting/getting element parameters if the ID is not known but the name of the element. Input: wcType = element type (e.g. “LINE” for a line, see appendix “List of element types”), wcName = Name of the element Output: lElementID = ID of the element
BOOL GetNodeFromElement(unsigned long& lNodeID, unsigned long lElementID, int nNodeNr)	Gets a node from an element Input: lElementID = ID of the element from which to get the node nNodeNr = (value 0-4, depends on which node ID to get) 0= 1st node, 2= 2nd node, 3 = 3rd node 4 = 4th node Output: lNodeID = ID of the node
BOOL GetProtectedElement(unsigned long& lElementID, unsigned long lProtectionID)	Gets the protected element of a protection device Input: lElementID = ID of the protection device from which to get the element Output: lElementID = ID of the protected element

void GetElements(TCHAR* sType,int& nCount,unsigned long*& pElementIDs);	Gets a list of all element ID's of a certain element type. This list can be used to iterate through the list of elements in a project with a "for" loop. Input: sType = element type (e.g. "LINE" for a line, see appendix "List of element types"), Output: nCount = number of elements in the list, pElementIDs = pointer to the list of element ID's
void FreeElements(unsigned long*& pElementIDs);	If the function "GetElements" has been used, then the memory of the list of ID's must be freed with this function. You should not use the C function "free" to free up the memory. Input: pElementIDs = pointer to the list of element ID's
BOOL DeleteElement(unsigned long lID);	Deletes an element from the project. Input: lID = ID of the element. Returns TRUE on success.
GetElementByUniqueName(TCHAR* wcName,unsigned long& lElementID)	Get an element by name. all names must be unique in the NEPLAN project! Input: wcName = name of element Output: lElementID = ID of the element Returns TRUE on success.
GetElementByUniqueName(TCHAR* wcName, TCHAR* wcType, unsigned long& lElementID)	Get an element by name. all names must be unique within their element type in the NEPLAN project! Input: wcName = name of element WcType = type of the elelement (e.g. „LINE", „LOAD", ect.) Output: lElementID = ID of the element Returns TRUE on success.
BOOL AddElement(unsigned long& lElementID, TCHAR* wcElementType, TCHAR* wcElementName, TCHAR* wcNodeName1=NULL, TCHAR* wcNodeName2=NULL, TCHAR* wcNodeName3=NULL,	Adds an element to an element to a NEPLAN project. Input: wcElementType = element type (e.g. "LINE" for a line, see appendix "List of element types"), wcNodeName1 = Name of the first node to which the

TCHAR* wcNodeName4=NULL);	element is connected, wcNodeName2 = Name of the 2nd node to which the element is connected, wcNodeName3 = Name of the 3rd node to which the element is connected (e.g. for 3-winding transformer), wcNodeName4 = Name of the 4th node to which the element is connected (e.g. for 4-winding transformer). Output: lElementID = internal ID of the added element. Returns TRUE on success.
BOOL AddRegulator(unsigned long& lElementID, TCHAR* wcElementType, TCHAR* wcElementName, int nRegulatorType, TCHAR* wcRegulatorName);	Adds a predefined regulator to the project. Input: wcElementType = element type (see appendix: List of Element types), wcElementName = name of the regulator, nRegulatorType = name of the predefined regulator type (see appendix: List of regulator types, e.g. EXCITER) wcRegulatorName: name of the predefined regulator type (e.g. _T("EXCITERS IEEET4"). Output: lElementID = internal ID of the added regulator. Returns TRUE on success.
BOOL SwitchElement(unsigned long lID, BOOL bSwitch);	Switches the logical switch of the element On/Off. Input: lID = ID of the element to switch On/Off, bSwitch = “FALSE” for switching off and “TRUE” for switching on the element. Returns TRUE on success.
BOOL SwitchElementAtNode(unsigned long lElemID, unsigned long lNodeID, BOOL bSwitch);	Switches the logical switch of the element at the specified node On/Off. Input: lElemID = ID of the element to switch On/Off, lNodeID = ID of the node bSwitch = “FALSE” for switching off and “TRUE” for switching on

	the element. Returns TRUE on success.
BOOL GetResultDouble(unsigned long lID,TCHAR* sResultType, double& dValue);	Gets a result value. Input: lID = ID of the element from which the result should be returned, sResultType = type of the result (see appendix: List of result types) Output: dValue = value of the result. Returns TRUE on success.
void RunSelectedAnalysis();	Runs the actual selected analysis of the analysis combo box in the toolbar of the NEPLAN project.
void RunAnalysis(int nCalculationModul);	Runs an analysis. Input: nCalculationModul = analysis to be executed (see appendix: List of analysis to be executed)
BOOL RunAnalysisGWTimeSim();	Runs a Gas/Water time simulation.
BOOL RunAnalysisGW();	Runs a Gas/Water calculation.
BOOL RunAnalysisLF(BOOL bStopIFNonConv=TRUE);	Runs a loadflow calculation. If bStopIFNonConv is set to FALSE then the error message indicating non-convergence will be supressed.
BOOL RunAnalysisFeederLF(unsigned long lNodeID)	Runs a loadflow calculation of a specified feeder. The feeder must be defined by a node or element identifier.
BOOL RunAnalysisPartnetLF(unsigned long lNodeID, BOOL bStopIFNonConv=TRUE)	Runs a loadflow calculation of a partial network. The partial network must be defined by a node or element identifier. If bStopIFNonConv is set to FALSE then the error message indicating non-convergence will be supressed.
BOOL RunAnalysisSC();	Runs a short circuit analysis.

void RunAnalysisTS(TCHAR* wcResFileName);	Runs a transient stability analysis. Input: Name of the result file name.
void RunAnalysisTSWithAllDisturbances(TCHAR* wcResFileName);	Runs a transient stability analysis (old stability module) with all defined disturbances. Input: Name of the result file name.
void RunAnalysisDynWithAllDisturbances(TCHAR* wcResFileName);	Runs the NEPLAN dynamic simulator with all defined disturbances. Input: Name of the result file name.
BOOL RunAnalysisTD(double dTEND=1.0, TCHAR* wcDisturbanceFilePathName=NULL,TCHAR* wcPlotFilePathName=NULL, TCHAR* wcResFileName=NULL, TCHAR* wcDynamicDataFileName=NULL);	Runs the NEPLAN Dynamic Simulator. Inputs: simulation time; full path of disturbance file; full path of screenplot file; full path of result file; full path of dynamic data file (See Appendix 6 for further details). Returns TRUE on success.
BOOL RunAnalysisLFWithProfile(TCHAR* wcTimeStamp)	Runs a load flow with load profile at a certain time. Input: wcTimeStamp = format for the time stamp is yyyy-mm-dd-hh-mm (year, month, day, hours, minutes) Returns TRUE on success
BOOL ExportToSQLDB(TCHAR* sConnect);	Exports the NEPLAN project into a SQL database. Input: sConnect = connect string to the database (e.g. _T("Provider=Microsoft.Jet.OLEDB.4.0; Data Source= TestDatabase.mdb; "). Returns TRUE on success.
BOOL ImportFromSQLDB(TCHAR* sConnect, TCHAR* sNetworkName);	Imports data from a SQL database. Input: sConnect = connect string to the database (e.g. _T("Provider=Microsoft.Jet.OLEDB.4.0; Data Source= TestDatabase.mdb; "), sNetworkName = name of the network which should be

	imported from the database. Returns TRUE on success.
BOOL PrintDirect();	Prints the actual selected diagram in the selected zoom view. Returns TRUE on success.
BOOL PrintZoomAll();	Prints the actual selected diagram in the “zoom all“ view. Returns TRUE on success.
BOOL PrintAllDiagrams();	Prints all diagrams of the project in the “zoom all“ view. Returns TRUE on success.
BOOL SelectViewSetting(TCHAR* wcViewName);	Selects a view, which is stored in the “Diagram View Settings”. Input: name of the stored view. Returns TRUE on success.
BOOL ActivateDiagramByName(TCHAR* wcViewName);	Opens a selected diagram. Input: Name of the diagram to be opened. Returns TRUE on success.
BOOL ImportFileElectrical(TCHAR* wcFileName, int nFileType, TCHAR* wcProtectionLibName=NULL, TCHAR* wcNeplanLibName=NULL);	Imports a power system network into a NEPLAN project. Input: wcFileName = file name of the import file, nFileType = file type of the import file (see appendix: List of import file types) wcProtectionLibName = library file name of the protection devices, wcNeplanLibName = name of the NEPLAN library file. Returns TRUE on success.
BOOL ExportFileElectrical(TCHAR* wcFileName, int nFileType);	Exports a power system network into a import/export file. Input: wcFileName = file name of the export file, nFileType = file type of the export file (see appendix: List of import/export file types) Returns TRUE on success.
BOOL ImportFileGasWater(TCHAR* wcFileName, int nFileType, TCHAR* wcNeplanLibName=NULL);	Imports a gas/water/district heating network into a NEPLAN project.

	Input: wcFileName = file name of the import file, nFileType = file type of the import file (see appendix: List of import file types) wcNeplanLibName = name of the NEPLAN library file. Returns TRUE on success.
BOOL ExportFileGasWater(TCHAR* wcFileName, int nFileType);	Exports a gas/water/district heating network into a import/export file. Input: wcFileName = file name of the export file, nFileType = file type of the export file (see appendix: List of import/export file types).
BOOL JoinProject(TCHAR* wcFileName);	Joines a second project to the actual open project. Input: wcFileName = File name to join.
BOOL WriteMessageToLogFile(TCHAR* wcMessage, int nNoteWarningError=0);	Writes a log message to a log file. Input: wcMessage = message to be written to the log file NNoteWarningError = defines the type of the message (0=note, 1= warning, 2=error) Returns TRUE on success.
BOOL RedrawDiagrams()	Updates data (all colors, texts, partial networks, etc.) and redraws all open diagrams.
BOOL RefreshScreen()	Refreshs the top most diagram. This redarw is faster than the redraw with the function “RedrawDiagrams()”, since it makes no data update for network coloring.
void ShowMessage(TCHAR* wcMessage, int nMessageWnd=0)	Shows a message to the NEPLAN message window. The nMessageWnd variable can be used to display to the different NEPLAN message windows. - nMessageWnd = 0 -> NEPLAN Messages window - nMessageWnd = 1 -> NEPLAN Error window - nMessageWnd = 2 -> NEPLAN Analyisis window

void ShowMessageNoNewLine(TCHAR* wcMessage, int nMessageWnd=0)	Same as 'ShowMessage' but no new line will be added to the message.
BOOL UpdateAllElementsWithLibraryData(TCHAR* wcElementLibFileName, TCHAR* wcProtectionLibFileName = NULL)	Updates all elements with the data from an element library (*.neplib) and a protection library (*.sd3) Input: wcElementLibFileName = element library file (*.neplib) wcProtectionLibFileName = protection file name (*.sd3)
BOOL UpdateElementWithLibraryData(unsigned long lElementID, TCHAR* wcElementLibFileName, TCHAR* wcLibName=NULL);	Updates the element data with the data from the actual element library Input: lElementID = ID of the element to be updated wcElementLibFileName = element library file (*.neplib) WcLibName= Name of the library (can be NULL for default library
BOOL RestorationShowObjectiveFunctionDlg()	Shows the objective function dialog of the load restoration module (re-supply module)
BOOL RestorationShowSelectPlanDlg ()	Shows the plan selection dialog (after analysis with different objective functions) of the load restoration module (re-supply module)
BOOL RestorationShowSelectStateDlg ()	Shows the state selection dialog of the load restoration module (re-supply module)
Close_NEPLAN(BOOL bAskForSave=FALSE)	Closes (exits) the NEPLAN application. Input: bAskForSave = if TRUE then NEPLAN ask if the changed projects should be saved before exit.
Hide_NEPLAN()	Hide the NEPLAN farme window. NEPLAN can therefore run in background mode.
Show_NEPLAN()	Shows the NEPLAn frame window after it has been hidden with the „Hide_Window()“ function.
BOOL ConnectToNetwork(TCHAR* wcIpAdress, int portNr,	NEPLAN my be connected to a TCP/IP bus. It may either run as

BOOL bServer)	server or as client. Input: wcIpAdress = IP Address portNr = port number bServer = if TRUE, the connect as server
BOOL SetShowMessageBox(BOOL bShow)	Enable or disable the pop-up message boxes in NEPLAN. If FALSE the warning and error messages will not pop-up. Input: bShow = If TRUE enable pop-up messages (default) Output: returns TRUE;
BOOL RunNPLDll(TCHAR* wcDllFileName)	Starts an external NPL Dll. Input: wcDllFileName = Filename of the NPL Dll. Output: TRUE if DLL can be started otherwise FALSE.
BOOL AllowToAddErrorMessages(BOOL bAllow)	Defines, if the error message shall be saved in for later retrival. Input: bAllow = if TRUE then messages are saved
BOOL ClearAllErrorMessages()	Clears all saved error messages, if "AllowToAddErrorMessages" was set to TRUE
BOOL GetNumOfSavedErrorMessages(int& nValue)	Gets the number of saved error messages. Output: returns TRUE on success Output: nValue = number of saved error messages
BOOL GetSavedErrorMessageAt(int nAt, LPTSTR lpValue, int& nErrorNum, int& nSeverity);	Gets the string of the saved error message at the position "nAt". The first message has position nAt=0. Input: Number of message to retrieve. Use first the function „GetNumOfSavedErrorMessages“ to find out the max. number of saved error messages. Output: TRUE on success. Output: lpValue = message string Output: nErrorNum = error number if available Output: nSeverity = 0 (will be returned in later versions)
BOOL GetAllSavedErrorMessages	Gets all saved error messages. This function is only available in

	TCP/IP command mode.
BOOL AddGraphicToNode(unsigned long lID, double x, double y)	Add a graphic (point symbol) to the node with ID = lID. Input: lID: ID of the node x, y: Coordinates in mm. The user is reponsible to map world coordinates into drawing coordinates. The total size of the diagram shoulf not exceed 3m x 3m. The origin (0, 0) of the diagram is the upper-left corner. Output: returns TRUE on success
BOOL AddGraphicLine(unsigned long lID)	Adds a straight graphic line between two nodes. The nodes must not be symbols (no busbars). Input: lID: ID of the line Output: returns TRUE on success
BOOL AddGraphicsToAllLines(BOOL bOnlyLinesWithoutGraphic=TRUE);	Adds a straight graphic line between two nodes, for all lines in the network. . The nodes must not be symbols (no busbars) Input: bOnlyLinesWithoutGraphic: If TRUE only lines without graphic will be added by a graphic Output: returns TRUE on success
BOOL AddMultiPointGraphicToLine(unsigned long lID, int numPoints, double* pCoordinates);	Adds a graphic line with more than 2 points between two nodes. The nodes must not be symbols (no busbars). Input: lID: ID of the line numPoints: number of points pCoordinates: Arrayowith numPointsx2 double values Output: returns TRUE on success
BOOL CreateVariantFeeder(unsigned long lNodeID, unsigned long lElementID, TCHAR* wcFeederName, int colR, int colG, int colB)	Adds a new feeder definition to the variant. Input: - lNodeID node at which the feeder starts - lElementID element at node lNodeID from which the

	feeder starts - wcFeederName: Name of the feeder - colR, colG, colB: color of the feeder Output: returns TRUE on success
BOOL CreateVariantZone(TCHAR* wcFeederName, int colR, int colG, int colB, double dLoadactor)	Adds a new network zone to the network. Input: - wcName: Name of the zone - colR, colG, colB: color of the zone - dLoadFactor: zone load factor Output: returns TRUE on success
BOOL CreateVariantArea(TCHAR* wcFeederName, int colR, int colG, int colB)	Adds a new network area to the network. Input: - wcName: Name of the area - colR, colG, colB: color of the area Output: returns TRUE on success
BOOL DeleteVariantArea(TCHAR* wcName)	Deletes an area from the network. Input: - wcName: Name of the area Output: returns TRUE on success
BOOL DeleteVariantZone(TCHAR* wcName)	Deletes a zone from the network. Input: - wcName: Name of the area Output: returns TRUE on success
BOOL DeleteVariantFeeder(TCHAR* wcName)	Deletes a feeder from the network. Input: - wcName: Name of the area Output: returns TRUE on success
BOOL GetVariantParameterBool(TCHAR* wcParameter, bool& bValue);	Gets a variant parameter of type “bool” Input: wcParameter = name of the paramter Output: bValue = value of the parameter.

	Returns TRUE on success. For a list of variant parameters see appendix (List of variant parameter types).
BOOL SetVariantParameterString(TCHAR* wcParameter, TCHAR* wcValue);	Sets the value of a variant parameter of type “char”. Input: sParameter = name of the parameter of the element, wcValue = value of the parameter to be set. Returns TRUE on success. For a list of parameter types see appendix (List of element parameter types).
BOOL GetVariantParameterString(TCHAR* wcParameter, LPTSTR lpValue);	Gets a variant parameter of type “char” Input: wcParameter = name of the paramter Output: lpValue = value of the parameter. Returns TRUE on success. For a list of variant parameters see appendix (List of variant parameter types).
BOOL SetVariantParameterBool(TCHAR* wcParameter, bool bValue);	Sets the value of a variant parameter of type “bool”. Input: sParameter = name of the parameter of the element, bValue = value of the parameter to be set. Returns TRUE on success. For a list of parameter types see appendix (List of element parameter types).
BOOL AddLineSection(unsigned long lLineID, TCHAR* wcSectionName, TCHAR* wcSectionType, double length, TCHAR* wcElementLibFileName/*=NULL*/)	Adds a new line section (only for electrical networks) to a line. The electrical data of type ‚wcSectionType’ will automatically looked up in the library , wcElementLibFileName’ and assigned to the line section. Input: - LineID, the line at wich the section will be added. - length, the length of the line section - wcSectionType, the model tye of the line section. - wcElementLibFileName, the library file name

	Output: returns TRUE on success
BOOL GetNumLineSections(unsigned long lineID ,int &nCount)	Returns the number of line section of a line. Input: lineID = ID of the line Output : nCount = numer of line sections Returns TRUE on succces.
BOOL LoadTcpIpCommandDll(TCHAR* wcDllCmdFileName)	Loads a user written dll file, which contains commands which may be send to a NEPLAN server. Input: wcDllCmdFileName, DLL file name to load.
BOOL ShowNotesInLogFile(BOOL bShow)	Defines if note messages should be written to the log file Input: bShow= if TRUE note messages will be written Output: returns TRUE on success
BOOL ShowWarningsInLogFile(BOOL bShow)	Defines if warning messages should be written to the log file Input: bShow= if TRUE warning messages will be written Output: returns TRUE on success
BOOL ShowErrorsInLogFile(BOOL bShow)	Defines if error messages should be written to the log file Input: bShow= if TRUE error messages will be written Output: returns TRUE on success
void ClearWindowMessages(int nMsgWndNum=-1)	Clears the message windows in NEPLAN. Input: nMsgWndNum (-1= All windows 0=Message window 1=Error Window 2=Analysis Window)
void SetZoomTo(double dZoomValue)	Set the zoom level of the actual diagram. Input : dZoomValue = zoom level to set
BOOL GetUserDataDouble(unsigned long lID,TCHAR* sParameter, double& dValue)	Get user defined element parameter of type double
BOOL SetUserDataDouble(unsigned long lID,TCHAR* sParameter, double dValue)	Set user defined element parameter of type double
BOOL GetUserDataInt(unsigned long lID,TCHAR*	Get user defined element parameter of type integer

wcParameter, int& nValue)	
BOOL SetUserDataInt(unsigned long lID,TCHAR* wcParameter, int nValue)	Set user defined element parameter of type integer
BOOL GetUserDataBool(unsigned long lID,TCHAR* wcParameter, bool& bValue)	Get user defined element parameter of type BOOL
BOOL SetUserDataBool(unsigned long lID,TCHAR* wcParameter, bool bValue)	Set user defined element parameter of type BOOL
BOOL GetUserDataString(unsigned long lID,TCHAR* wcParameter, LPTSTR lpValue)	Get user defined element parameter of type string
BOOL SetUserDataString(unsigned long lID,TCHAR* wcParameter, TCHAR* wcValue)	Set user defined element parameter of type string
BOOL CheckDatabaseConsistency(BOOL bTryToResolveProblems, int& nReturnCode)	Performs database consistency check
BOOL GetConnectedElementsAtNode(unsigned long lNodeID, int nAllOnOff, int& nCount, unsigned long*& pElementIDs);	Int nAllOnOff : /0= get all elements, 1...Switched On elements, 2...Switched Off element !! Importan t!! the pElementIDs must be freed withe 'FreeElements' function after use to avoid memery leaks
BOOL RunAnalysisLFAdvanced(int nStep, BOOL bCalcSelectedFeeders);	The load flow runs directly on the internal data structure without building up the network completely new for load flow nStep=0: Setup load flow nStep=1: Read network data nStep=2: Run first load flow nStep=3: Run subsequent load flow nStep=4: Delete load flow
BOOL GetNodeResultLFAdvanced(unsigned long lID, double& UL1, double& UL2, double& UL3, double& UangL1, double& UangL2, double& UangL3);	Get the node result directly from the internal load flow calculation structure. Can be used after using with “RunAnalysisLFAdvanced”.
BOOL GetElementResultLFAdvanced(unsigned long lID, int	Get the element result directly from the internal load flow

nSide, double P[3], double Q[3], double Imag[3], double& dLoading);	calculation structure. Can be used after using with “RunAnalysisLFAdvanced”.
BOOL SetLoadLFAdvanced(unsigned long lID, double dP, double dQ);	Sets the load data directly in the internal load flow structure, without building up the network completely new.
BOOL SwitchElementLFAdvanced(unsigned long lID, BOOL bSwitch);	Sets the logical switch directly in the internal load flow structure, without building up the network completely new.
BOOL UpdatePartialNetworkID();	Updates the NEPLAN internal partial network numbers.
BOOL SearchAndShowElementOnActiveDiagram(unsigned long lID);	BOOL Search and display (zoom to) the element on the active diagram
BOOL AddMultiPointBusbar(unsigned long lID, int numPoints, double* pCoordinates);	adds graphic coordinates to a busbar with more that 2 points numPoints=number of points-> in pCoordinates = 2*numPoints values. The coordinates (x,y) must be given im mm.

4) Appendix

4a) Appendix : List of analysis to be executed

Description of the run command for running various power system analysis functions
BOOL RunAnalysis(int nCalculationModule)
Output: returns TRUE if the analysis has started
Input: nCalculationModule, the calculation modiule to start,
this can be one of the following:
Example:

```
RunAnalysis(BCP_SELECTED_MODULE_RELIABILITY);

BCP_SELECTED_MODULE_LF                          // Loadflow
BCP_SELECTED_MODULE_LF_WITH_PROFILE             // Loadflow  with loadprofile
BCP_SELECTED_MODULE_OPTIMAL_SEPERATION_POINTS   // optimal separation points
BCP_SELECTED_MODULE_OPTIMAL_DISTRIBUTION_NETWORK // optimal distribution network
BCP_SELECTED_MODULE_CONTINGENCY_ANALYSIS        // contingency analysis
BCP_SELECTED_MODULE_OPTIMAL_POWER_FLOW          // optimal power flow
BCP_SELECTED_MODULE_SC                          // short circuit analysis
BCP_SELECTED_MODULE_HS                          // harmonic analysis
BCP_SELECTED_MODULE_MS                          // motor starting analysis
BCP_SELECTED_MODULE_VS                          // voltage stability
BCP_SELECTED_MODULE_SS                          // small signal stability
BCP_SELECTED_MODULE_TS                          // transient stability
BCP_SELECTED_MODULE_RELIABILITY                 // reliability analysis
BCP_SELECTED_MODULE_CP                          // optimal capacitor placement
BCP_SELECTED_MODULE_INVA                        // investment analysis
```

```
BCP_SELECTED_MODULE_SIMP          // Simpow transient stability
BCP_SELECTED_MODULE_FF            // Fault finding procedure analysis
BCP_SELECTED_MODULE_SIMP_LA       // Simpow linear analysis
BCP_SELECTED_MODULE_CABLEDIM      // cable dimensioning
BCP_SELECTED_MODULE_ATC           // Transfer capapility calculation
BCP_SELECTED_MODULE_FEEDERREIN    // Feeder reinforcement analysis
BCP_SELECTED_MODULE_RESUPPLY      // Load restoration analysis
```

4b) Appendix : List of the of element types

```
////////////////////////////////////////////////////////////////////////////
//Description of the parameter access for element names
//Following element types for accessing elements are defined
// Access of lElementId through: GetElementByName(TYPE, Name, lElementId);
// Example : GetElementByName(_T("GENERATOR"),_T("GX"),lElementId);
//          lElementId will be returned
////////////////////////////////////////////////////////////////////////////
//Following element types (TYPES) are defined
//Water and Heating elements for water networks

// Water element types
_T("WATER-NODE")
_T("WATER-LINE")
_T("WATER-VALVE")
_T("WATER-RESERVOIR")
_T("WATER-PUMP")
```

```
_T("WATER-CIRC-PUMP")
_T("WATER-SLIDER")
_T("WATER-HYDRANT")
_T("WATER-STATION")
_T("WATER-SPECIAL-LOAD")
_T("WATER-LINE-LOAD")
_T("WATER-FITTING")
_T("WATER-HEATPLANT")
_T("WATER-HEATLOAD")

//Gas element types
_T("GAS-NODE")
_T("GAS-LINE")
_T("GAS-VALVE")
_T("GAS-RESERVOIR")
_T("GAS-PUMP")
_T("GAS-CIRC-PUMP")
_T("GAS-SLIDER")
_T("GAS-HYDRANT")
_T("GAS-STATION")
_T("GAS-SPECIAL-LOAD")
_T("GAS-LINE-LOAD")
_T("GAS-FITTING")

//Power system element types
_T("STATION")
_T("BUSBAR-NODE")
_T("LINE")
_T("ASY_LINE")
```

_T("COUPLING")
_T("REACTOR")
_T("TRANSFORMER")
_T("ASY-TRANSFORMER")
_T("SERIE-RLC")
_T("PARALLEL-RLC")
_T("3W-TRANSFORMER")
_T("4W-TRANSFORMER")
_T("EQUIVALENT_SERIE_LF")
_T("EQUIVALENT_SERIE_SC")
_T("DISCSWITCH")
_T("LOADSWITCH")
_T("CIRC_BREAKER")
_T("FEEDER")
_T("GENERATOR")
_T("MACHINE")
_T("LOAD")
_T("SHUNT")
_T("SVS")
_T("DC_NODE")
_T("DC_LINE")
_T("DC_LOAD")
_T("EARTHCOND")
_T("PYLON")
_T("LINECOUPL")
_T("PSBLK")
_T("FILTER")
_T("SERIE_RLC_E")
_T("EQUIVALENT_SHUNT_LF")

```
_T("EQUIVALENT_SHUNT_SC")
_T("CURSOURCE")
_T("VOLSOURCE")
_T("DC_VOLSOURCE")
_T("DC_REACTOR")
_T("DC_SHUNT")
_T("DC_MOTOR")
_T("DC_GROUND")
_T("CONVERTER")
_T("CONVERTER_3POLE")
_T("CUSTOMER_CONN")
_T("LINE-LOAD")
_T("TCSC")
_T("UPFC")
_T("STATCOM")
_T("PWM")
_T("SERIE_TR2")
_T("GROUND")
_T("AC_GENERIC_COMP")

//protection devices
_T("DIST_RELAIS")
_T("OVERCUR_RELAIS")
_T("FUSE")
_T("CIRC_BREAKER_2")
_T("CUR_TRANS")
_T("VOL_TRANS")
_T("DISCSWITCH_2")
_T("LOADSWITCH_2")
```

```
_T("EARTHSWITCH")
_T("SURGEARRESTER")
_T("FREQ_RELAIS")
_T("VOL_RELAIS")
_T("POW_RELAIS")
_T("MINMAX_RELAIS_ON_NODE")
_T("MINMAX_RELAIS_ON_LINK")
_T("PS_RELAIS")
_T("MEASURE")
_T("CCT")

//Regulator/Specials
_T("REGULATOR")
_T("TURBINE")
_T("TABLE")
_T("INERTIA")
_T("MECHLOAD")
_T("FAULT")

//function blocks types
_T("BLOCK-INPUT")
_T("BLOCK-OUTPUT")
_T("BLOCK-SOURCE")
_T("BLOCK-SUMMATION")
_T("BLOCK-PRODUCT")
_T("BLOCK-INVERTER")
_T("BLOCK-RATIO")
_T("BLOCK-EXPONENTIAL")
_T("BLOCK-RECTANGULAR")
```

_T("BLOCK-POLAR")
_T("BLOCK-CONSTANT")
_T("BLOCK-NLF1")
_T("BLOCK-NLF2")
_T("BLOCK-NLF3")
_T("BLOCK-NLF4")
_T("BLOCK-LIMIT1")
_T("BLOCK-LIMIT2")
_T("BLOCK-LV-GATE")
_T("BLOCK-HV-GATE")
_T("BLOCK-PERUNIT")
_T("BLOCK-SATURATION")
_T("BLOCK-STATEXC")
_T("BLOCK-INTEGRATOR")
_T("BLOCK-LAG")
_T("BLOCK-DERLAG")
_T("BLOCK-DERIVATIVE")
_T("BLOCK-LEADLAG")
_T("BLOCK-PID")
_T("BLOCK-R2")
_T("BLOCK-R3")
_T("BLOCK-R4")
_T("BLOCK-STEP")
_T("BLOCK-FIRINGANGLE")
_T("BLOCK-DEADBAND")
_T("BLOCK-NOT")
_T("BLOCK-AND")
_T("BLOCK-OR")
_T("BLOCK-SWITCH")

_T("BLOCK-POWER")
_T("BLOCK-ABSOLUTE")
_T("BLOCK-R5")
_T("BLOCK-ITESWITCH")
_T("BLOCK-SIGNAL GENERATOR")
_T("BLOCK-DISTRIBUTER")
_T("BLOCK-COSINE")
_T("BLOCK-SINE")
_T("BLOCK-TANGENTS")
_T("BLOCK-ARCCOSINE")
_T("BLOCK-ARCSINE")
_T("BLOCK-ARCTANGENTS")
_T("BLOCK-NETWORK_SOURCE")
_T("BLOCK-HYSTERESIS")
_T("BLOCK-RUNTIME_DELAY")
_T("BLOCK-BISTABLE_SWITCH")
_T("BLOCK-ACTIVE_DIS")
_T("BLOCK-PICKUP_DELAY")
_T("BLOCK-RESET_DELAY")
_T("BLOCK-IMPULS")
_T("BLOCK-MESSAGE")
_T("BLOCK-TRIPLE_AND")
_T("BLOCK-TRIPLE_OR")
_T("BLOCK-COUNTER")
_T("BLOCK-MINMAX_RELAY")
_T("BLOCK-LINE")
_T("BLOCK-CIRCLE")
_T("BLOCK-POLYGON")
_T("BLOCK-LENS")

_T("BLOCK-AD_TIME_DELAY")
_T("BLOCK-TD_TIME_DELAY")
_T("BLOCK-NETWORK_PU")

4c) Appendix : List of the element variables

```
////////////////////////////////////////////////////////////////////////////
//Description of the paramter access for electrical elements
//example:
//    1) GetElementByName(_T("GENERATOR"),_T("GX"),lElementId); // gets the internal Element-ID
//    2) SetParameterDouble(lElementId,_T("Td01"),1.23);
//
////////////////////////////////////////////////////////////////////////////
// the parameter can be accesde with the following functions:

//to get the NEPLAN internal ID of an element use the follwoing function
BOOL GetElementByName(TCHAR* wcType,TCHAR* wcName,unsigned long& lElementID)
      Output: returns TRUE if the element has been found
      Output: lElementID, the ID of the element
      Input: wcType, the type of the element (see above)
      Input: wcName, the name of the element which id has to be returned

// to set the parameters use the following functions
BOOL  SetParameterInt(unsigned long lID,TCHAR* wcParameter, int nValue)
BOOL  SetParameterBool(unsigned long lID,TCHAR* wcParameter, bool bValue)
BOOL  SetParameterString(unsigned long lID, TCHAR* wcParameter, TCHAR* wcValue)
BOOL  SetParameterDouble(unsigned long lID,TCHAR* wcParameter, double dValue)

      Output: returns TRUE if the data has changed succesfully
      Input: lID, ID of the element
```

```
      Input: wcParameter, name of the parameter to change (see below)
      Input: dValue, value of the parameter.


//description of variables which can be get/set for all elements
      ElementType               | String   | Type of the element, e.g. „LINE“, (only get is possible!)
      Name                      | String   | Name of the element
      AliasName                 | String   | AliasName of the element
      Description               | String   | Description of the element
      ModelType                 | String   | Type (library type) of the element
      Projected                 | BOOL     | Projected flag of the element
      InMaintenance             | BOOL     | In maintenance flag of the element
      FeederName                | String   | Name of the network feeder to which the element/node belongs to
      ZoneName                  | String   | Name of the network zone to which the element/node belongs to
      AreaName                  | String   | Name of the network area to which the element/node belongs to
      PartialNetworkNr          | int      | Partial network id of the element/node
      Phase                     | int      | Phase information of the element/node


//Description of the Busbar
      Un                        | double   |
      Fn                        | double   |
      Umin                      | double   | Min. allowable node voltage in %.
      Umax                      | double   | Max. allowable node voltage in %.
      Ipmax                     | double   | Max. allowable peak short circuit current of bus-bar in kA
      Uset                      | double   | Setting value in % of nominal voltage
      Trelais                   | double   | Tripping time in seconds of a primary protection, e.g. fuse in a
distribution feeder
      AccessTime                | double   | Reliabilty: Access time for busbar
      ManualSwitchingTime       | double   |
      GapBetweenConductors      | double   |
      NodeType                  | integer  | Node type. 0=Busbar; 1=Sleeve; 2= Special Node; 3= Main busbar
      ProtectionType            | integer  | For reliability: 0:None 1:Diff.Prot. 2:Backup 3:Diff.+Backup 4:Standard
      InStationType             | integer  |
      FailureGroup              | integer  |
```

```
      SCIndicator               | integer  |
      SwitchbayConfig           | integer  |
      Equipment                 | integer  |
      UseDefaultAccessTime      | BOOL     | use default access time
      DistProtNode              | BOOL     | If TRUE, node will be handled like a distance protection relay node
      EMTNode                   | BOOL     | If TRUE, node will be treated as EMT node during mixed mode dyn. sim.
      SwitchbayRemoteCtrl       | BOOL     |
      SwitchbayRemoteOpen       | BOOL     |
      SwitchbayUseDefault       | BOOL     |
      UseDefaultManualSwitchingTime | BOOL     |
      ReliabilityIdeal          | BOOL     |
      ReliabilityTypeBusbar     | string   |
      ReliabilityTypeCB         | string   |
      ReliabilityTypeDisc       | string   |

//Description of the 2-pole Circuit Breaker
      Ir                        | double   | Rated current
      Ipmax                     | double   | Admissible peak current
      Iamax                     | double   | Max. break current
      Ur                        | double   | Rated voltage
      Ik2max                    | double   | Max. short circuit current Ik"
      R1                        | double   |
      X1                        | double   |
      R0                        | double   |
      X0                        | double   |
      Kfactor                   | double   | Only for ANSI: Ratio of rated maximum voltage Urmax to the lower limit
of the range of operation voltage
      CosPhiTest                | double   | Cos(phi) at which the breaker was tested (only for ANSI breakers)
      Cycles                    | integer  | Interrupting time of the ANSI breakers in cycles. Possible values are:
0="2", 1="3", 2="5", 3="8
      NormIecAnsi               | integer  | 0=IEC, 1=ANSI
      VoltageLevelLvHv          | integer  | Voltage level: 0=LV, 1=HV
      RemoteControlled          | BOOL     | Indicates, if the switch is remote controlled
      MiniatureCB               | BOOL     | indicates if this is a miniatur circuit breaker
```

```
      BusbarProtection          | BOOL     | Reliability: Indicate if the circuit breaker will be used for busbar
protection

//Description of the Circuit Breaker which are on an element
      Ir                        | double   | Rated current
      Ipmax                     | double   | Admissible peak current
      Iamax                     | double   | Max. break current
      Ur                        | double   | Rated voltage
      Ik2max                    | double   | Max. short circuit current Ik"
      R1                        | double   |
      X1                        | double   |
      R0                        | double   |
      X0                        | double   |
      Kfactor                   | double   | Only for ANSI: Ratio of rated maximum voltage Urmax to the lower limit
of the range of operation voltage
      CosPhiTest                | double   | Cos(phi) at which the breaker was tested (only for ANSI breakers)
      Cycles                    | integer  | Interrupting time of the ANSI breakers in cycles. Possible values are:
0="2", 1="3", 2="5", 3="8
      NormIecAnsi               | integer  | 0=IEC, 1=ANSI
      VoltageLevelLvHv          | integer  | Voltage level: 0=LV, 1=HV
      RelFailureGroup           | integer  | Reliability: Failure group
      BayConfig                 | integer  | Reliability: Switch bay configuration
      ReliabilityType           | String   | Reliability: Reliability type
      RemoteControlled          | BOOL     | Indicates, if the switch is remote controlled
      MiniatureCB               | BOOL     | indicates if this is a miniatur circuit breaker
      BusbarProtection          | BOOL     | Reliability: Indicate if the circuit breaker will be used for busbar
protection
      ReliabilityIdeal          | BOOL     | Reliability: If TRUE, then the CN will be treated as ideal

//Description of the HVDC Converter
      Iset                      | double   |
      Pset                      | double   |
      Uset                      | double   |
      Xc                        | double   | Commutating reactance in Ohm
      Im                        | double   | Current margin in % of the current set value Iset
```

```
      REquivLosses              | double   | Equivalent total active power losses in the valves and auxiliaries in
Ohm
      VDrop                     | double   | Voltage drop across the valves in V
      Umode                     | double   | only in P control: voltage below which the Pd-mode changes to Id-mode
      Udr                       | double   |
      Ir                        | double   |
      DR                        | double   |
      DX                        | double   |
      TetaIni                   | double   |
      TetaMin                   | double   |
      TetaMax                   | double   |
      GammaMin                  | double   |
      GammaSet                  | double   |
      TapNom                    | double   | Nominal tap ratio of the converter transformer in pu from DC to AC side
      TapAct                    | double   |
      DeltaTap                  | double   | Converter transformer tap-step in pu
      TapMin                    | double   | Minimum value of converter transformer tap ratio in pu
      TapMax                    | double   | Maximumvalue of converter transformer tap ratio in pu.
      Rg                        | double   |
      ImDistribution            | double   | This is the converter participation factor in % in case of a multi-
terminal system
      L                         | double   | Bulid in smoothing reactor for stabilizing mode (positive slope)
      NumBridges                | integer  | Number of bridges
      PIUControl                | integer  | 0=P: Power: 1=I:Current, 2=U: Voltage
      RectiverInverter          | integer  | Mode: 0=Rectivier; 1=Inverter
      NegativePositivePole      | integer  |
      TapLock                   | BOOL     | If TRUE, then the tap will be fixed on TapAct
      SerieConverter            | BOOL     | If TRUE, converter belonsg to a serie converter
      TrafoIncluded             | BOOL     | if TRIU, the trafo is included in the converter model
      ThetaFixToSetValue        | BOOL     | If TRUE theta is fixed
      GammaFixToSetValue        | BOOL     | If TRUE gamma is fixed
      GammaFixToMin             | BOOL     | If TRUE gamma is fixed to gamma min
      UseEnhancedModel          | BOOL     | Use enhanced converter model

//Description of the Busbar Coupler
```

```
      R                         | double   | Positive sequence resistance in mOhm
      X                         | double   |
      Y                         | double   |
      R0                        | double   | Zero sequence resistance in mOhm
      X0                        | double   |
      Y0                        | double   |
      Ir                        | double   | Rated current in kA
      Ipmax                     | double   | Max. allowable peak short circuit current in kA
      BayConfig                 | integer  | Reliability: Switch bay configuration  (0=CB-Disc-CB; 1= Disc)
      RemoteControlled          | BOOL     | Reliability: Indicates, if the coupler is remote controlled
      BusbarProtection          | BOOL     | Reliability: Indicate if the circuit breaker will be used for busbar
protection
      IdealCoupler              | BOOL     | If TRUE, the coupler will be treatde as idela in load flow (R=X=0)

  //Description of the Mechanical Load
      Mr                        | double   | Numinal torque in Nm
      K                         | double   | For dynamic analysis: Loading factor
      N                         | double   | Exponential factor
      M0                        | double   | Factor for parabel
      M1                        | double   | Factor for parabel
      M2                        | double   | Factor for parabel
      ParabolaTableExponential  | integer  | Load Torque given as  0=Parabola 1=Table 2=Exponential
      TableInNm                 | BOOL     | if TRUE, values in the table ar in Nm
      M012InNm                  | BOOL     | if TRUE values in the of m=,M1 and M2 ar in Nm
      Active                    | BOOL     | If TRUE, mechanical load is active

//Description of the Simpow Regulators and Turbines
      UserDefinedDSLArgument    | String   | DLS argument for user defined regulators
      Active                    | BOOL     | if TRUE, the regulator is active

      //{{DESCRIPTION_PART(CNPTechCurrentTrafoElec)
      Ir1             | double   | Rated current on primary side of CT in A
      Ir2             | double   | Rated current on secondary side of CT in A
      Ithks           | double   | Thermal short circuit current in A.
      Ithlf           | double   | Thermal steady state current in A.
```

```
Vr              | double   | Secondary terminal voltage rating for accuracy class
Knee            | double   | Secondary terminal voltage rating for knee-point in %
Sb              | double   | Rated power output in VA
ALF             | double   | Accuracy limit factor
Rct             | double   | Secondary winding resistance in Ohm
Lam_rem         | double   | Remanence
Off             | double   | offset magnitude
L_cable         | double   | Cable length in units
Rl_LU           | double   | Resistance Ohm/units Cable CT-Relay
Xl_LU           | double   | Reactance Ohm/units Cable CT-Relay
VA_relay        | double   | Relay Burdon MVA
PF_relay        | double   | Relay burdon cos(phi)
Standard        | integer  | Standard  (0: IEEE C37.110-1996, 1: IEC 60044-1 2003)
L               | integer  | accuracy class (0: C, 1: K)
CE              | integer  | Composite error
NP              | integer  | CT connection (0: 1phase, 1: 3phase)
CON             | integer  | CT connection (0: Y, 1: D)
CON_AT          | integer  | CT connection (0: at switchhouse, 1: at CT)
Units           | integer  | Units for line length and impedances
CableType       | String   | Cable type
ExitCurveAvaila| BOOL     | Saturation curve available

//{{DESCRIPTION_PART(CNPTechDcBusbarElec)
Un              | double   | Nominal voltage
Umin            | double   | Min. voltage
Umax            | double   | Max. voltage
Usoll           | double   | Set value for voltage (in %)
Ir              | double   | Rated busbar current (in A)
ReliabilityIdeal | BOOL     | Flag if busbar is ideal
MastaNode       | BOOL     | Indicates if it is a Masta node

//{{DESCRIPTION_PART(CNPTechVoltageSourceElec)
Ur              | double   | Rated voltage

//{{DESCRIPTION_PART(CNPTechUPFCElec)
```

```
      VTMax           | double   | Maximum series voltage magnitude in % of the bus nominal voltage
      VTMin           | double   | Minimum series voltage magnitude in % of the bus nominal voltage
      IQMax           | double   | Maximum shunt current in A
      IQMin           | double   | Minimum shunt current in A
      V1set           | double   | Setpoint for voltage control of the sending end
      V2Min           | double   | Minimum voltage magnitude at the receiving end
      V2Max           | double   | Maximum voltage magnitude at the receiving end
      RL              | double   | leakage resistance transformer (series)
      XL              | double   | leakage reactance transformer (series)
      RQ              | double   | leakage resistance transformer (shunt)
      XQ              | double   | leakage reactance transformer (shunt)
      PexchMax        | double   | Maximum MW through the UPFC device
      Pset            | double   | Set value for line flow active power
      Qset            | double   | Set value for line flow reactive power
      CurrSourceModel| integer  | Current Source model for dynamics (0: Source with real and imaginary part, 1:
source with magnitude and angle

      //{{DESCRIPTION_PART(CNPTechTrafo4Elec)
      Ur1             | double   | Rated voltage primary side
      Ur2             | double   | Rated voltage secondary side
      Ur3             | double   | Rated voltage tertiary side
      Ur4             | double   | Rated voltage 2. tertiary side
      Uk12            | double   | Short circuit voltage from primary to secondary (positive sequence)
      Uk012           | double   | Short circuit voltage from primary to secondary (zero sequence)
      Vcu12           | double   | Copper losses from primary to secondary (positive sequence)
      Uk13            | double   | Short circuit voltage from primary to tertiary (positive sequence)
      Uk013           | double   | Short circuit voltage from primary to tertiary (zero sequence)
      Vcu13           | double   | Copper losses from primary to tertiary (positive sequence)
      Uk23            | double   | Short circuit voltage from secondary to tertiary (positive sequence)
      Uk023           | double   | Short circuit voltage from primary to tertiary (zero sequence)
      Vcu23           | double   | Copper losses from secondary to tertiary (positive sequence)
      Uk14            | double   | Short circuit voltage from primary to 2. tertiary (positive sequence)
      Uk014           | double   | Short circuit voltage from primary to 2. tertiary (zero sequence)
      Vcu14           | double   | Copper losses from primary to 2. tertiary (positive sequence)
      Uk24            | double   | Short circuit voltage from secondary to 2. tertiary (positive sequence)
```

Uk024	double	Short circuit voltage from secondary to 2. tertiary (zero sequence)
Vcu24	double	Copper losses from secondary to 2. tertiary (positive sequence)
Uk34	double	Short circuit voltage from tertiary to 2. tertiary (positive sequence)
Uk034	double	Short circuit voltage from tertiary to 2. tertiary (zero sequence)
Vcu34	double	Copper losses from tertiary to 2. tertiary (positive sequence)
Re1	double	Earthing resistance primary side
Xe1	double	Earthing reactance primary side
Ze1_activ	double	Portion of active earting impedance primary side
Re2	double	Earthing resistance secondary side
Xe2	double	Earthing reactance secondary side
Ze2_activ	double	Portion of active earting impedance secondary side
Re3	double	Earthing resistance tertiary side
Xe3	double	Earthing reactance tertiary side
Ze3_activ	double	Portion of active earting impedance tertiary side
Re4	double	Earthing resistance 2. tertiary side
Xe4	double	Earthing reactance 2. tertiary side
Ze4_activ	double	Portion of active earting impedance 2. tertiary side
Sr1	double	Rated power at primary side
Sr2	double	Rated power at secondary side
Sr3	double	Rated power at tertiary side
Sr4	double	Rated power at 2.tertiary side
Delta_u	double	Angle in ° of the additional voltage per tap step
Erdtype1	integer	Earthing type primary (0:direct, 1:impedance, 2:isolated)
Erdtype2	integer	Earthing type secondary (direct, 1:impedance, 2:isolated)
Erdtype3	integer	Earthing type tertiary (0:direct, 1:impedance, 2:isolated)
Erdtype4	integer	Earthing type 2. tertiary (0:direct, 1:impedance, 2:isolated)
Seiten	integer	Tap chnager side (0:primary, 1:secondary, 2:tertiary, 3:2. tertiary
Tapmin	integer	Min. Tap position
Tapmit	integer	Rated Tap position
Tapmax	integer	Max. Tap position
Tapakt	integer	Actual tap position
Schagru12	String	Vector group primary to secondary
Schagru13	String	Vector group primary to tertiary
Schagru14	String	Vector group primary to 2. tertiary

```
//{{DESCRIPTION_PART(CNPTechTrafo3Elec)
    Ur1             | double   | Rated voltage primary side
    Ur2             | double   | Rated voltage secondary side
    Ur3             | double   | Rated voltage tertiary side
    Uk12            | double   | Short circuit voltage from primary to secondary (positive sequence)
    Uk012           | double   | Short circuit voltage from primary to secondary (zero sequence)
    Vcu12           | double   | Copper losses from primary to secondary (positive sequence)
    Uk23            | double   | Short circuit voltage from secondary to tertiary (positive sequence)
    Uk023           | double   | Short circuit voltage from secondary to tertiary (zero sequence)
    Vcu23           | double   | Copper losses from secondary to tertiary (positive sequence)
    Uk31            | double   | Short circuit voltage from tertiary to primary (positive sequence)
    Uk031           | double   | Short circuit voltage from tertiary to primary (zero sequence)
    Vcu31           | double   | Copper losses from tertiary to primary (positive sequence)
    Re1             | double   | Earthing resistance primary side
    Xe1             | double   | Earthing reactance primary side
    Ze1_activ       | double   | Portion of active earting impedance primary side
    Re2             | double   | Earthing resistance secondary side
    Xe2             | double   | Earthing reactance secondary side
    Ze2_activ       | double   | Portion of active earting impedance secondary side
    Re3             | double   | Earthing resistance tertiary side
    Xe3             | double   | Earthing reactance tertiary side
    Ze3_activ       | double   | Portion of active earting impedance tertiary side
    Sr12            | double   | Rated power from primary to secondary
    Sr23            | double   | Rated power from secondary to tertiary
    Sr31            | double   | Rated power from tertiary to primary
    Beta_u          | double   | Angle in ° of the additional voltage per tap step
    Delta_u         | double   | Voltage in % of the additional voltage
    Beta_u_2        | double   | Angle in ° of the additional voltage per tap step for 2nd tapchnager
    Delta_u_2       | double   | Voltage in % of the additional voltage for 2nd tapchanger
    Uger            | double   | Set value for regulated voltage
    Pger            | double   | Set value for regulated active power
    Tn_uk12         | double   | SC voltage at tap min. primary to secondary (positive sequence)
    Tn_uk012        | double   | SC voltage at tap min. primary to secondary (zero sequence)
    Tx_uk12         | double   | SC voltage at tap max. primary to secondary (positive sequence)
    Tx_uk012        | double   | SC voltage at tap max. primary to secondary (zero sequence)
```

Tn_uk23	double	SC voltage at tap min. secondary to tertiary (positive sequence)
Tn_uk023	double	SC voltage at tap min. secondary to tertiary (zero sequence)
Tx_uk23	double	SC voltage at tap max. secondary to tertiary (positive sequence)
Tx_uk023	double	SC voltage at tap max. secondary to tertiary (zero sequence)
Tn_uk31	double	SC voltage at tap min. tertiary to primary (positive sequence)
Tn_uk031	double	SC voltage at tap min. tertiary to primary (zero sequence)
Tx_uk31	double	SC voltage at tap max. tertiary to primary (positive sequence)
Tx_uk031	double	SC voltage at tap max. primary to secondary (zero sequence)
Ir_gr_mn1	double	Minimum current limit primary side
Ir_gr_mx1	double	Maximum current limit primary side
Ir_gr_mn2	double	Minimum current limit secondary side
Ir_gr_mx2	double	Maximum current limit secondary side
Ir_gr_mn3	double	Minimum current limit tertiary side
Ir_gr_mx3	double	Maximum current limit tertiary side
Sr_gr_mn	double	Minimum power limit primary side
Sr_gr_mx	double	Maximum power limit primary side
Sr2_gr_mn	double	Minimum power limit secondary side
Sr2_gr_mx	double	Maximum power limit secondary side
Sr3_gr_mn	double	Minimum power limit tertiary side
Sr3_gr_mx	double	Maximum power limit tertiary side
I0	double	No-load current
Vfe	double	Iron losses
Erdtype1	integer	Earthing type primary (0:direct, 1:impedance, 2:isolated)
Erdtype2	integer	Earthing type secondary (direct, 1:impedance, 2:isolated)
Erdtype3	integer	Earthing type tertiary (0:direct, 1:impedance, 2:isolated)
Seiten	integer	Tap chnanger at (0:primary, 1:secondary, 2:tertiary)
Tapmin	integer	Min. Tap position
Tapmit	integer	Rated Tap position
Tapmax	integer	Max. Tap position
Tapakt	integer	Actual tap position
Seiten_2	integer	2nd Tap chnanger at (0:primary, 1:secondary, 2:tertiary)
Tapmin_2	integer	Min. Tap position for 2nd tapchanger
Tapmit_2	integer	Rated Tap position for 2nd tapchanger
Tapmax_2	integer	Max. Tap position for 2nd tapchanger
Tapakt_2	integer	Actual tap position for 2nd tapchanger

```
Auswertung      | integer  | Evaluation according to (0:Current, 1:Power)
EarthingComponentSide | integer  | Earthing side
Schagru         | String   | Vector group
Regeln          | BOOL     | On-load tapchanger ?
Ausgleichs_wicklung | BOOL     | Compensation winding ?
Blocktrafo      | BOOL     | Power system unit transformer?
UseSaturationModel | BOOL     | Use saturation model?
CommonEarthing | BOOL     | common earthing ?
AutoTransformer | BOOL     Auto transformer ?
UseEarthingComponent | BOOL     | use earthing component?

//{{DESCRIPTION_PART(CNPTechTrafo2Elec)
Ur1             | double   | Rated voltage primary side
Ur2             | double   | Rated voltage secondary side
Uk              | double   | Short circuit voltage (positive sequence)
Uk0             | double   | Short circuit voltage (zero sequence)
Vcu             | double   | Copper losses (positive sequence)
Vcu0            | double   | Copper losses (zero sequence)
Re1             | double   | Earthing resistance primary side
Xe1             | double   | Earthing reactance primary side
Ze1_activ       | double   | Portion of active earting impedance primary side
Re2             | double   | Earthing resistance secondary side
Xe2             | double   | Earthing reactance secondary side
Ze2_activ       | double   | Portion of active earting impedance secondary side
I0              | double   | No load current
Vfe             | double   | Iron losses
Sr              | double   | Rated power
Beta_u          | double   | Angle in ° of the additional voltage per tap step
Delta_u         | double   | Voltage in % of the additional voltage
Uger            | double   | Set value for regulated voltage
Pger            | double   | Set value for regulated active power
Tn_uk           | double   | SC voltage at tap min. (positive sequence)
Tn_uk0          | double   | SC voltage at tap min. (zero sequence)
Tx_uk           | double   | SC voltage at tap max. (positive sequence)
Tx_uk0          | double   | SC voltage at tap max. (zero sequence)
```

Ir_gr_mn1	double	Minimum current limit primary side
Ir_gr_mx1	double	Maximum current limit primary side
Ir_gr_mn2	double	Minimum current limit secondary side
Ir_gr_mx2	double	Maximum current limit secondary side
Sr_gr_mn	double	Minimum power limit
Sr_gr_mx	double	Maximum power limit
Uk010	double	open circuit voltage in zero sequence at primary side
Uk020	double	open circuit voltage in zero sequence at secondary side
Comp_imin	double	Minimum current value for compounding
Comp_imax	double	Maximum current value for compounding
Comp_umin	double	Minimum voltage value for compounding
Comp_umax	double	Maximum voltage value for compounding
Ubmax	double	Max. Operating voltage
Ibmax	double	Max. Operating current
Cosb	double	Operating cos(phi)
Ubmin	double	Min. Operating voltage
pTap	double	Off-load Tapchanger
UgerDC	double	Voltage set value for regulated DC node
DCAngleGer	double	Angle set value
DCGammaGer	double	Converter angle set value
Erdtype1	integer	Earthing type primary (0:direct, 1:impedance, 2:isolated)
Erdtype2	integer	Earthing type secondary (0:direct, 1:impedance, 2:isolated)
Tapside	integer	Tap on side (0:primary, 1:secondary)
busContrside	integer	Controlled bus on side (0:primary, 1:secondary, 2:remote)
Tapmin	integer	Min. Tap position
Tapmit	integer	Rated Tap position
Tapmax	integer	Max. Tap position
Tapakt	integer	Actual tap position
Auswertung	integer	Evaluation according to (0:Current, 1:Power)
DCAngleVoltage	integer	Voltage Angle control (0:Control angle, 1:Control Voltage)
GammaOrTheta	integer	Converetr control (0:Theta control, 1:Gamma Control)
Schagru	String	Vector group (old obsolete style)
VectorGroup	String	Vector group
Blocktrafo	BOOL	power system unit transformer?
Regeln	BOOL	On-load tapchanger?

```
Compound        | BOOL     | Compounding active?
Ausgleichs_wicklung | BOOL     | Compensation winding?
Schaltbar       | BOOL     | Switchable?
OperValueActiv | BOOL     | Operating values for SC active?
DCControl       | BOOL     | DC node controlled?
UseSaturationModel    | BOOL     | Use saturation model?
CommonEarthing        | BOOL     | common earthing ?
AutoTransformer       | BOOL     Auto transformer ?
UseEarthingComponent  | BOOL     | use earthing component?
SymmetricalRegulation | BOOL     | Symmetrical Regulation for UCTE?
ReliabilityType       | String   | Reliability: Reliability type
ReliabilityIdeal      | BOOL     | Reliability: Reliability ideal


//{{DESCRIPTION_PART(CNPTechTrafo2AsyElec)
Ur1             | double   | Rated voltage primary side
Ur2             | double   | Rated voltage secondary side
Uk              | double   | Short circuit voltage (positive sequence)
Vcu             | double   | Copper losses (positive sequence)
I0              | double   | No load current
Vfe             | double   | Iron losses
Sr              | double   | Rated power
Delta_u         | double   | Additional voltage per tap step
Uger            | double   | Set value for voltage regulation
Phase2          | integer  | Phase connection secondary side
Tapside         | integer  | Tapside (0: primary, 1: secondary)
busContrside    | integer  | Controlled node: (0: primary, 1: secondary)
Tapmin          | integer  | Min. Tap position
Tapmit          | integer  | Rated Tap position
Tapmax          | integer  | Max. Tap position
Tapakt          | integer  | Actual tap position
Schagru         | String   | Vector group
Negativ_Polarit| BOOL     | Is there negative polarity?
Regeln          | BOOL     | Regulated?
Schaltbar       | BOOL     | Is transformer swichable?
```

```
//{{DESCRIPTION_PART(CNPTechTCSCElec)
    Xc              | double   | Reactance of the capacitor in Ohm
    Xl              | double   | Reactance of the inductor in Ohm
    Tetamin         | double   | Minimum value of thyristor firing angle.
    Tetamax         | double   | Maximum value of thyristor firing angle.
    Xmin            | double   | Minimum value of module reactance
    Xmax            | double   | Maximum value of module reactance
    Pset            | double   | Set value for active power
    Iset            | double   | Set value for current
    TrAngleSet      | double   | Set value for transmission angle
    XtotSet         | double   | Set value for Xtot
    MaxVoltDrop     | double   | Maximum voltage drop along the TCSC
    Control         | integer  | Control mode (0:P, 1:I, 2:Xtot, 3:Transm. angle
    ModuleOper      | integer  | 0: One module operation, 1: multi module operation
    NumModules      | integer  | Number of modules
    TetaXLimits     | integer  | Limits: (0:Teta Limits, 1:X Limits

//{{DESCRIPTION_PART(CNPTechSwitchElec)
    Ir              | double   | Rated current
    Ipmax           | double   | max. peak current
    Iamax           | double   | max. breaking current
    Ur              | double   | Rated voltage
    Ik2max          | double   | max. Ik"
    R1              | double   | Resistance positive system
    X1              | double   | Reactance positive system
    R0              | double   | Resistance zero system
    X0              | double   | Reactance zero system
    Remote_controlled | BOOL     | remote controlled

    //{{DESCRIPTION_PART(CNPTechSVSElec)
    Uref            | double   | Reference voltage
    Xsl             | double   | Slope admittance:
    Qc              | double   | Capacitive power
    Ql              | double   | Inductive power
```

```
      Transformer    | BOOL     | Is there a transformer?

  //{{DESCRIPTION_PART(CNPTechSurgeArrestorElec)
      Ur             | double   | Rated voltage in kV.
      Uc             | double   | Permanent operating voltage in kV
      Ures_10ka      | double   | Residual Voltage in kV for 10 kA (8/20(s)
      Ures_1ka       | double   | Residual Voltage in kV for 1 kA (30/60(s)
      Energie        | double   | Energy absorption capacity in kJ/kVuc.
      Festigkeit     | double   | Long-wave capacity 2000(s in A.

  //{{DESCRIPTION_PART(CNPTechSTATCOMElec)
      Uref           | double   | Reference voltage
      Xsl            | double   | Slope admittance:
      ImaxL          | double   | Maximal current for inductive operation.
      ImaxC          | double   | Maximal current for capacitive operation.
      P0             | double   | Zero sequence active power
      Q0             | double   | Zero sequence reactive power
      P              | double   | MW injection or MW comsumption, losses (whatever)
      CurrSourceModel| integer  | Current Source model for dynamics (0: Source with real and imaginary part, 1:
source with magnitude and angle
      Transformer    | BOOL     | Is there a transformer?

  //{{DESCRIPTION_PART(CNPTechShuntElec)
      P1             | double   | Active power (positive sequence)
      Q1             | double   | Reative power (positive sequence)
      P0             | double   | Active power (zero sequence)
      Q0             | double   | Reactive power (zero sequence)
      Ur             | double   | Rated Voltage
      Uset           | double   | Set value for voltage control
      UdPosDev       | double   | Voltage deviation (positive sequence)
      UdNegDev       | double   | Voltage deviation (negative sequence)
      UdMin          | double   | Min. voltage
      UdMax          | double   | Max. voltage
      Opt1           | double   |
      Opt2           | double   |
```

```
    Sw1             | double   |
    Sw2             | double   |
    Sr              | double   | Earthing transformer rated power
    uRr0            | double   | Earthing transformer copper losses (zero sequence)
    ukr0            | double   | Earthing transformer short circuit voltage (zero sequence)
    RE              | double   | Earthing transformer earthing resistance
    XE              | double   | Earthing transformer earthing reactance
    CosPhiSet       | double   | cos(phi) set value
    CosphiMin       | double   | cos(phi) min. value
    CosphiMax       | double   | cos(phi) max. value
    Regulation      | integer  | Control mode (0: none, 1: discrete, 2: continous)
    RemoteControlled | BOOL     | Remote controlled ?
    EarthingTrafo   | BOOL     | Shunt is an earthing transformer?
    CntrlCosPhi     | BOOL     | Is shunt cos(phi) regulated?
    SetIsCap        | BOOL     | Ist cos(phi) set a capacitive value?
    HasCosphiLimits | BOOL     | Has cos(phi) limits?
    MinIsCap        | BOOL     | Is cos(phi) min capacitive?
    MaxIsCap        | BOOL     | Is cos(phi) max capacitive?

//{{DESCRIPTION_PART(CNPTechSerTr2Elec)
    SN              | double   | Rated power
    UN1             | double   | Rated voltage primary side
    UN2             | double   | Rated voltage secondary side
    Uk              | double   | Positive sequence short circuit voltage in %
    Vcu             | double   | positive sequence copper losses in %
    USoll           | double   | Voltage set value
    Delta_u         | double   | Magnitude of the additional voltage per tap step on the tap location side.
    VectorGroup     | String   | Vector Group
    Tapmit          | integer  | Rated tap position
    Tapmax          | integer  | Max. Tap position
    Tapakt          | integer  | Actual tap position
    busContrside    | integer  | Tapchanger on side (0: primary, 1: secondary)
    CntrlActive     | BOOL     | Tapchanger active?

//{{DESCRIPTION_PART(CNPTechSeriesRLCElec)
```

```
    Ur              | double    | Rated voltage
    R               | double    | Resistance
    L               | double    | Inductance
    C               | double    | Capacitance
    Ipr             | double    | Capacitor protective level
    C0              | double    | Capacitance of zero sequence
    Ir              | double    | Rated current
    MOV             | BOOL      | MOV protective series capacitor?

//{{DESCRIPTION_PART(CNPTechSeriesERLCElec)
    Ur              | double    | Rated voltage
    R               | double    | Resistance
    L               | double    | Inductance
    C               | double    | Capacitance
    Isolated        | BOOL      | Is isolated in zero sequence (no ground connection)?

//{{DESCRIPTION_PART(CNPTechReactorElec)
    Ur              | double    | Rated voltage
    Uk              | double    | Positive sequence short circuit voltage in %
    Ir              | double    | Rated current
    Vcu             | double    | Positive sequence Copper losses in %
    Uk0             | double    | Zero sequence short circuit voltage in %
    Vcu0            | double    | Zero sequence Copper losses in %

//{{DESCRIPTION_PART(CNPTechPWMElec)
    SN              | double    | Rated power
    UN              | double    | Rated voltage
    XQ              | double    |
    Fr              | double    |
    CTeta           | double    |
    R               | double    |
    X               | double    |
    RP              | double    |
    Cmin            | double    |
    Cmax            | double    |
```

```
    XC              | double   |
    XT              | double   |
    SetValue1       | double   |
    SetValue2       | double   |
    CregIni         | double   |
    TetaIni         | double   |
    K               | double   |
    Ti              | double   |
    TF              | double   |
    UB1             | double   |
    UB2             | double   |
    TFB             | double   |
    KLT             | double   |
    TILT            | double   |
    TFLT            | double   |
    KLM             | double   |
    TILM            | double   |
    Z               | double   |
    ILIM            | double   |
    TFLM            | double   |
    Cntr1           | integer  |
    Cntr2           | integer  |
    LinePort        | integer  | for P,Q control = 0,1,2 (actually line can be also transformer)
    DSLType         | integer  | 0=VSCREG, 1=VSCSUM
    SWB             | integer  |
    InitCteta       | BOOL     |

//{{DESCRIPTION_PART(CNPTechPowStatUnitElec)
    Ur1             | double   | Transformer rated voltage primary side
    Ur2             | double   | Transformer rated voltage secondary side
    Uk              | double   | Transformer short circuit voltage positive sequence
    Uk0             | double   | Transformer short circuit voltage zero sequence
    Vcu             | double   | Transformer copper losses positive sequence
    Vcu0            | double   | Transformer copper losses zero sequence
    Re1             | double   | Transformer earthing resistance primary side
```

```
    Xe1             | double   | Transformer earthing reactance primary side
    Srt             | double   | Transformer rated power
    Xd2str          | double   | Saturated subtransient reactance (d_axis) in %
    Xdsat           | double   | Saturated synchronous reactance
    X2              | double   | Reactance of negative sequence
    Cosphi          | double   | cos(Phi)
    Srg             | double   | Generator rated power
    Pbetr           | double   | Operating active power
    Qbetr           | double   | Operating reactive power
    pUrG            | double   | Generator Ur deviation
    pTap            | double   | Off-load Tapchanger
    Uger            | double   | set voltage value for PV
    Erdtype         | integer  | Transformer Earthing type (0:direct, 1:impedance, 2:isolated)
    Ufmzuufr        | integer  | Ratio maximum to rated exciter voltage (0:"1.3", 1:"1.6", 2:"2.0")
    RelPriority     | integer  | Priority for reliability calculation
    RelLoadCurve    | integer  | Load curve for reliability calculation
    StUpPrio        | integer  | startup priority
    Lftype          | integer  | Node type (0=PQ, 1=PV)
    Schagru         | String   | Transformer vector group
    Turbo           | BOOL     | Generator is a turbo type?
    Regeln          | BOOL     | On-load Tapchanger?

    //{{DESCRIPTION_PART(CNPTechParallelRLCElec)
    Ur              | double   | Rated voltage
    R               | double   | Resistance
    L               | double   | Inductance
    C               | double   | Capacitance

//{{DESCRIPTION_PART(CNPTechMeasurDeviceElec)
    Mess_PL1        | double   | measured active power in Phase R
    Mess_QL1        | double   | measured reactive power in Phase R
    Mess_IL1        | double   | measured current in Phase R
    Mess_UL1        | double   | measured voltage in Phase R
    Mess_PL2        | double   | measured active power in Phase S
    Mess_QL2        | double   | measured reactive power in Phase S
```

```
    Mess_IL2        | double   | measured current in Phase S
    Mess_UL2        | double   | measured voltage in Phase S
    Mess_PL3        | double   | measured active power in Phase T
    Mess_QL3        | double   | measured reactive power in Phase T
    Mess_IL3        | double   | measured current in Phase T
    Mess_UL3        | double   | measured voltage in Phase T
    MessMin_PL1     | double   | Min. measured active power in Phase R
    MessMin_QL1     | double   | Min. measured reactive power in Phase R
    MessMin_IL1     | double   | Min. measured current in Phase R
    MessMin_UL1     | double   | Min. measured voltage in Phase R
    MessMin_PL2     | double   | Min. measured active power in Phase S
    MessMin_QL2     | double   | Min. measured reactive power in Phase S
    MessMin_IL2     | double   | Min. measured current in Phase S
    MessMin_UL2     | double   | Min. measured voltage in Phase S
    MessMin_PL3     | double   | Min. measured active power in Phase T
    MessMin_QL3     | double   | Min. measured reactive power in Phase T
    MessMin_IL3     | double   | Min. measured current in Phase T
    MessMin_UL3     | double   | Min. measured voltage in Phase T
    Mess_per_phase  | BOOL     | are measured values per phase?
    Use_meas_data   | BOOL     | user measurement data?

//{{DESCRIPTION_PART(CNPTechMachine)
    Ur              | double   | Rated voltage
    Pr              | double   | Rated active power
    Cosphi          | double   | cos(Phi)
    Eta             | double   | Efficiency factor
    IazuIr          | double   | Ratio starting to rated current
    M0              | double   | constant term of load torque parable
    M1              | double   | linear term of load torque parable
    M2              | double   | quadratic term of load torque parable
    Ir              | double   | Rated current
    J               | double   | Moment of Inertia
    sr              | double   | Rated slip
    MazuMr          | double   | Ratio Starting to rated torque
    Cosanl          | double   | cos(Phi) at startup
```

```
MkzuMr           | double   | Ratio Peak to rated torque
Pbetr            | double   | Operating active power
Qbetr            | double   | Operating reactive power
Plim             | double   | Cos(phi) regulation, limit for active power
Cosphi_oper      | double   | Operating cos(phi)
Cosphi_min       | double   | Minimum cos(phi)
Cosphi_max       | double   | Maximum cos(phi)
Pmin             | double   | Active power lower limit
Pmax             | double   | Active power higher limit
Rm               | double   | Stator resistance
Ansi_fak         | double   | ANSI factor
Pmech            | double   | mechanical power
R1_zu_Zr         | double   | Stator resistance in per unit of machine
X1_zu_Zr         | double   | Stator reactance in per unit of machine
Xh_zu_Zr         | double   | Magnetizing reactance in per unit of machine
T_hochf          | double   | Time delay for startup in s
Tr_ur1           | double   | Rated voltage of startup Transformer at primary side
Tr_ur2           | double   | Rated voltage of startup Transformer at secondary side
Tr_sr            | double   | Rated apparent power of startup Transformer
Tr_uRr           | double   | Copper losses of startup Transformer in %
Tr_ukr           | double   | Short circuit voltage of startup Transformer in %
Zanf_re          | double   | Startup Resistance in Ohm
Zanf_im          | double   | Startup Reactance in Ohm
R_rotor          | double   | Startup Rotor resistance in Ohm
Qc               | double   | Startup reactive power for C startup
T_umschalt       | double   | By-pass the startup device at this time
S_umschalt       | double   | By-pass the startup device at this slip
DASMTorqueInp    | double   | Torque for DFIG
DASMSlipInp      | double   | Slip for DFIG
DASMPtotInp      | double   | Total power for DFIG
DASMQtotInp      | double   | Total reactive power for DFIG
SIMPOWC1         | double   | SIMPOW factor C1
SIMPOWC2         | double   | SIMPOW factor C2
SIMPOWDX         | double   | SIMPOW factor DX
SIMPOWTorqueFactor| double   | SIMPOW Torque factor
```

```
Ini_slip         | double   | Initial value for slip
Ini_Tm           | double   | Initial value for mechanical torque
Ini_freq_s       | double   | Initial value for frequency stator
Ini_freq_r       | double   | Initial value for frequency rotor
Ini_uq_s         | double   | Initial value for q-axis stator voltage
Ini_ud_s         | double   | Initial value for d-axis stator voltage
Ini_uq_r         | double   | Initial value for q-axis rotor voltage
Ini_ud_r         | double   | Initial value for d-axis rotor voltage
Ini_iq_s         | double   | Initial value for q-axis stator current
Ini_id_s         | double   | Initial value for d-axis stator current
Ini_iq_r         | double   | Initial value for q-axis rotor current
Ini_id_r         | double   | Initial value for d-axis rotor current

Sim_fak_LoadBalance  | double   | Load balance factor P
Sim_fak_LoadBalanceQ | double   | Load balance factor Q
SoftStartUs          | double   | Soft Start Us
SoftStartIs          | double   | Soft Start Is
SoftStartRampTime    | double   | Soft Start Ramp Time
SoftStartIsLimit  | double   | Soft Start Is Limit
SoftStartUe       | double   | Soft Start Ue
SoftStartIe       | double   | Soft Start Ie
SoftStartUsKick   | double   | Soft Start Us Kick
SoftStartIsKick   | double   | Soft Start Is Kick
SoftStartRampTimeKick | double   | Soft Start Ramp Time Kick
Anzahl           | integer  | Number of parallel (identical) machines
Polpaar          | integer  | Pol paires
ControlType      | integer  | Cos(phi) regulation (0:no, 1:cos(phi) charact., 2:Q-reg, 3:QP-reg
Lftype           | integer  | Type for LF: (1: ML, 2: PQbetr. 2: PQnom)
Starting_device | integer  | Startup device (0:Direct, 1:YD 2:Zstator, 3:Rrotor, 4:Transformer, 5:Capacitance)
N_decrem         | integer  | number of cascade for startup
Slip_umschalt    | integer  | By-pass the startup device depending on slip (value 1) or time (value 0)
LoadTorqueGiven | integer  | Load Torque given as 0=Parabola 1=Table
Model            | integer  | Dynamic Model: 0:1.order, 1:3.order, 2:5.order
RelLoadCurve     | integer  | Reliability load characteristics
RelPriority      | integer  | Reliability priority for re-connection
```

```
    StUpPrio          | integer  | Startup priority
    DASMInput_P_T_s   | integer  | 0: Power-Torque as input, 1: Active-Reactive power as input
    TransientSlipIni  | integer  | Slip init. (0: Mload, 1: Loadflow/Mload, 2:Loadflow
    SIMPOWType        | integer  | Simpow type
    SoftStartMode     | integer  | Soft Start Mode
    LoadCharForStar   | String   | Type Load Characteristic
    LoadCharForOper   | String   | Type of Load Characteristic
    MotorChar         | String   | Type of Motor Characteristic
    HarmSource        | String   | Harmonic source type
    Srspsg            | BOOL     | Converter drive?
    RueckSpeisung     | BOOL     | Reversible power flow possible?
    Capacitive_oper   | BOOL     | Operating cos(phi) is capacitive?
    Capacitive_min    | BOOL     | Minimum cos(phi) is capacitive?
    Capacitive_max    | BOOL     | Maximum cos(phi) is capacitive?
    Hochf             | BOOL     | Motor will startup?
    TableInNm         | BOOL     | Table given in Nm
    M012InNm          | BOOL     | Load torque parabola (M0,M1,M2 given in Nm
    DoubleFed         | BOOL     | double fed ASM (DFIG)?
    DASMTorqueFromC   | BOOL     | Torque from load torque DFIG
    DASMMinCos        | BOOL     | Min cosphi for DFIG
    ANSIUnits         | BOOL     | SNSI units?
    MotorAsHarmSource | BOOL     | Is Motor a harmonic current source
    ReadMeasurementData | BOOL     | use measurement data instead of day characteristic
    Sim_fak_var     | BOOL     | Active for for load balancing
    SoftStartKick | BOOL     | Soft Start Kick yes/no
    SoftStartIsLim | BOOL     | Soft Start Is Lim yes/no
    CalculateEquivalentCircuitParameter | BOOL     | CalculateEquivalentCircuitParameter (yes/no)

//{{DESCRIPTION_PART(CNPTechLoadSwitch2Elec)
    Ir              | double   | Rated current
    Ipmax           | double   | max. peak current
    Iamax           | double   | max. breaking current
    Ur              | double   | Rated voltage
    Ik2max          | double   | max. Ik"
    R1              | double   | Resistance positive system
```

```
    X1              | double   | Reactance positive system
    R0              | double   | Resistance zero system
    X0              | double   | Reactance zero system
    RelFailureGroup| integer  | Failure group
    ReliabilityType| String   | Reliability type
    ReliabilityIdeal| BOOL    | Element ideal
    Remote_controlled| BOOL    | remote controlled

//{{DESCRIPTION_PART(CNPTechLoad)
    Slast           | double   | Apparent power
    Pload           | double   | Active power
    Qload           | double   | Reactive power
    Iload           | double   | Load current
    Cosphi          | double   | Cos(phi)
    Elast           | double   | Yearly energy consumption
    K1_vel          | double   | Velander coefficient 1
    K2_vel          | double   | Velander coefficient 2
    Sim_fak_LoadBalance| double   | P Scaling factor
    Sim_fak_LoadBalanceQ| double   | Q Scaling factor
    XP              | double   | LF exponential model: voltage exp. Factor for P -> obsolete in 5.4.3
    XQ              | double   | LF exponential model: voltage exp. Factor for Q -> obsolete in 5.4.3
    XP_DYN          | double   | Dynamic exponential model: voltage exp. Factor for P -> obsolete in 5.4.3
    XQ_DYN          | double   | Dynamic exponential model: voltage exp. Factor for Q -> obsolete in 5.4.3
    BLIP            | double   | constant current fraction of active load for static part -> obsolete in 5.4.3
    BLIQ            | double   | constant current fraction of reactive load for static part -> obsolete in 5.4.3
    BLPP            | double   | constant power fraction of active load for static part -> obsolete in 5.4.3
    BLPQ            | double   | constant power fraction of reactive load for static part -> obsolete in 5.4.3.
    BLIP_DYN        | double   | constant current fraction of active load for dynamic part -> obsolete in 5.4.3
    BLIQ_DYN        | double   | constant current fraction of reactive load for dynamic part -> obsolete in 5.4.3
    BLPP_DYN        | double   | constant power fraction of active load for dynamic part -> obsolete in 5.4.3.
    BLPQ_DYN        | double   | constant power fraction of reactive load for dynamic part -> obsolete in 5.4.3.
    BLFP            | double   | frequency-dependence factor of active load for static part -> obsolete in 5.4.3.
    BLFQ            | double   | frequency-dependence factor of reactive load for static part -> obsolete in 5.4.3.
    BLFP_DYN        | double   | frequency-dependence factor of active load for dynamic part -> obsolete in 5.4.3.
    BLFQ_DYN        | double   | frequency-dependence factor of active load for dynamic part -> obsolete in 5.4.3.
```

P0	double	Active Power for zero system
Q0	double	Reactive Power for zero system
A0P	double	Facror for load model.
A1P	double	Facror for load model.
A2P	double	Facror for load model.
A0Q	double	Facror for load model.
A1Q	double	Facror for load model.
A2Q	double	Facror for load model.
N0P	double	Exponential facror for load model.
N1P	double	Exponential facror for load model.
N2P	double	Exponential facror for load model.
N0Q	double	Exponential facror for load model.
N1Q	double	Exponential facror for load model.
N2Q	double	Exponential facror for load model.
CFP	double	frequency-dependence factor of active load for static part
CFQ	double	frequency-dependence factor of reactive load for static part
TP	double	Time constant for load
TP	double	Time constant for load
TQ	double	Time constant for load
TV	double	Time constant for load
TF	double	Time constant for load
StaticPortionP	double	Static Portion of P
StaticPortionQ	double	Static Portion of Q
Ua1	double	Upper voltage limit for a reduction factor R(u)=1.
Ub1	double	Upper voltage limit for a reduction factor R(u)=0.
Ua2	double	Lower voltage limit for a reduction factor R(u)=1.
Ub2	double	Lower voltage limit for a reduction factor R(u)=0.
DBUmin	double	Umin for dead band
DBUmax	double	Umax for dead band
TLagP	double	Time delay for active power for internal lag model
TLagQ	double	Time delay for reactive power for internal lag model
PPortion_DYN	double	Portion of dynamic load for active power P
QPortion_DYN	double	Portion of dynamic load for active power Q
Lftype	integer	Node type: (0=PQ,1=PC,2=IC,3=PI,4=SC,5=EC)
Units	integer	Units (0=High voltage/1=low voltage)

```
      Anz_we          | integer   | Number of domestic units
      EquivalentCircu| integer   | Modelled as parallel circuit in Harmonic Analysis
      VoltageDepModel| integer   | Type of voltage dependency( 0: none, 1: composite, 2: exponential
      RelPriority     | integer   | Priority for power station re-connection
      RelLoadCurve    | integer   | Load curve index (RAMSES system states)
      DynamicModel    | integer   | Dynamic model (0 : None, 1 : external Control circ., 2 : Internal Lag function, 3
: controlled admittance with Control circ.
      HarmSource      | String    | Harmonic source type
      Sim_fak_var     | BOOL      | For Load balance fix (not considered)?
      ReadMeasurement| BOOL      | Read measurement data from file
      UseMeasurementD| BOOL      | LF time simulations: use measurement data instead of day characteristic
      ConstantPowerFo| BOOL      | Voltage dependence: do not use the volt dep model for LF but use always constant
power model for LF
      DeathBand       | BOOL      | Death band active for SIMPOW
      LoadAsHarmSourc| BOOL      | Is load a harmonic current source

  //{{DESCRIPTION_PART(CNPTechLineLoadElec)
      Slast           | double    | Apparent power
      Plast           | double    | Active power
      Qlast           | double    | Reactive power
      Ilast           | double    | Load current
      Cosphi          | double    | Cos(phi)
      Elast           | double    | Yearly energy consumption
      K1_vel          | double    | Velander coefficient 1
      K2_vel          | double    | Velander coefficient 2
      Distance        | double    | Distance in % from starting node
      Sim_fak_LoadBal| double    | P Scaling factor
      Sim_fak_LoadBal| double    | Q Scaling factor
      StreetNr        | integer   | street number
      Lftype          | integer   | Node type: (0=PQ,1=PC,2=IC,3=PI,4=SC,5=EC)
      Units           | integer   | Units (0=High voltage/1=low voltage)
      Anz_we          | integer   | Number of domestic units
      eetAb           | String    | Address string

  //{{DESCRIPTION_PART(CNPTechLineElec)
```

```
  R_pos           | double   | Positive sequence resistance in Ohm/km or see Units.
  X_pos           | double   | Positive sequence reactance in Ohm/km or see Units
  G_pos           | double   | Positive sequence conductance in µS/km or see Units.
  C_pos           | double   | Positive sequence capacitance in µF/km or see Units
  Ir_min          | double   | Minimal rated current in A.
  Ir_max          | double   | Maximuml rated current in A.
  ReductionFac    | double   | Reduction factor.
  R_zero          | double   | Zero sequence resistance in Ohm/km or see Units.
  X_zero          | double   | Zero sequence reactance in Ohm/km or see Units
  C_zero          | double   | Zero sequence capacitance in µF/km or see Units
  G1_pos          | double   | Line compensation active power at starting node (positive sequence)
  B1_pos          | double   | Line compensation reactive power at starting node (positive sequence)
  G2_pos          | double   | Line compensation active power at ending node (positive sequence)
  B2_pos          | double   | Line compensation reactive power at ending node (zero sequence)
  G1_zero         | double   | Line compensation active power at starting node (zero sequence)
  B1_zero         | double   | Line compensation reactive power at starting node (zero sequence)
  G2_zero         | double   | Line compensation active power at ending node (zero sequence)
  B2_zero         | double   | Line compensation reactive power at ending node (zero sequence)
  Shunt1_active   | double   | portion in % of the compensation which is active.
  Shunt2_active   | double   | portion in % of the compensation which is active.
  Q               | double   | Cross section of the line in mm2
  PermTemp        | double   | Max. permitted temperature in °C for SC
  OperTemp        | double   | Operating Temperature in °C for Load flow calculation
  MaxOperTemp     | double   | Max Operating Temperature
Length          | double   | Line length
  Units           | integer  | Units for the input values below.
  ParallelLines   | integer  | Number of parallel lines between starting and ending node.
  Material        | integer  | Material (0:CU, 1:AL, 2:St, 3:None)
  InstallCost     | integer  | Installation costs(0:None, 1:Type 1, 2:Type 2, 3:Type3, 4:Type4, 5:Type5
  FreqAbh         | String   | Type of frequency dependence
  CableType       | String   | Cable Type
  SimpowDSLArgument| String   | SIMPOW DSL Argument
  Switchable      | BOOL     | Indicates, if the line is switchable
  Cable           | BOOL     | Indicates, whether the line is a cable or not
  Overhead        | BOOL     | Indicates, whether the line is a overhead or not
```

```
    UseSectionRelData | BOOL     | Flag if reliability data of line sections are used

//{{DESCRIPTION_PART(CNPTechLineAsyElec)
    R_rr            | double   | Resistance in Ohm/units (Phase R-R)
    R_rs            | double   | Resistance in Ohm/units (Phase R-S)
    R_rt            | double   | Resistance in Ohm/units (Phase R-T)
    R_rn            | double   | Resistance in Ohm/units (Phase R-N)
    R_ss            | double   | Resistance in Ohm/units (Phase S-S)
    R_st            | double   | Resistance in Ohm/units (Phase S-T)
    R_sn            | double   | Resistance in Ohm/units (Phase S-N)
    R_tt            | double   | Resistance in Ohm/units (Phase T-T)
    R_tn            | double   | Resistance in Ohm/units (Phase T-N)
    R_nn            | double   | Resistance in Ohm/units (Phase N-N)
    X_rr            | double   | Reactance in Ohm/units (Phase R-R)
    X_rs            | double   | Reactance in Ohm/units (Phase R-S)
    X_rt            | double   | Reactance in Ohm/units (Phase R-T)
    X_rn            | double   | Reactance in Ohm/units (Phase R-N)
    X_ss            | double   | Reactance in Ohm/units (Phase S-S)
    X_st            | double   | Reactance in Ohm/units (Phase S-T)
    X_sn            | double   | Reactance in Ohm/units (Phase S-N)
    X_tt            | double   | Reactance in Ohm/units (Phase T-T)
    X_tn            | double   | Reactance in Ohm/units (Phase T-N)
    X_nn            | double   | Reactance in Ohm/units (Phase N-N)
    C_rr            | double   | Capacitance in uF/units (Phase R-R)
    C_rs            | double   | Capacitance in uF/units (Phase R-S)
    C_rt            | double   | Capacitance in uF/units (Phase R-T)
    C_rn            | double   | Capacitance in uF/units (Phase R-N)
    C_ss            | double   | Capacitance in uF/units (Phase S-S)
    C_st            | double   | Capacitance in uF/units (Phase S-T)
    C_sn            | double   | Capacitance in uF/units (Phase S-N)
    C_tt            | double   | Capacitance in uF/units (Phase T-T)
    C_tn            | double   | Capacitance in uF/units (Phase T-N)
    C_nn            | double   | Capacitance in uF/units (Phase N-N)
    Ir_min          | double   | Minimum rated current in A.
    Ir_max          | double   | Maximum rated current in A.
```

```
    ReductionFac    | double   | Reduction factor.
    Q               | double   | Cross section of the line in mm2
    PermTemp        | double   | Max. permitted temperature in °C for the SC
    MaxOperTemp     | double   | Max Operating Temperature
    Units           | integer  | Units for the input values below.
    ParallelLines   | integer  | Number of parallel lines between starting and ending node.
    Material        | integer  | Material (0:CU, 1:AL, 2:St, 3:None)
    InstallCost     | integer  | Installation costs (0:None, 1:Type 1, 2:Type 2, 3:Type3, 4:Type4, 5:Type5
    FreqAbh         | String   | Type of frequency dependence
    Switchable      | BOOL     | Indicates, if the line is switchable
    Cable           | BOOL     | Indicates, whether the line is a cable or not
    Overhead        | BOOL     | Indicates, whether the line is a overhead or not

//{{DESCRIPTION_PART(CNPTechInertiaElec)
    Active          | BOOL     | Inertia active?

//{{DESCRIPTION_PART(CNPTechGroundElec)
    R               | double   | Resistance of ground
    L               | double   | Reactance of ground
    C               | double   | Capacitance of ground

//{{DESCRIPTION_PART(CNPTechGenerator)
    Ur              | double   | Rated voltage
    Sr              | double   | Rated power
    Xd2sat          | double   | d-axis subtransient reactance (saturated) in %
    Xd2             | double   | d-axis subtransient reactance (unsaturated) in %
    Xd1sat          | double   | d-axis transient reactance (saturated) in %
    Xd1             | double   | d-axis transient reactance (unsaturated) in %
    Xdsat           | double   | d-axis synchronous reactance (saturated) in %
    Xd              | double   | d-axis synchronous reactance (un-saturated) in %
    X2              | double   | Stator reactance X of negative system
    R2              | double   | Stator resistance of negative system
    X0              | double   | Reactance of zero system
    Cosphi          | double   | cos(Phi)
    Ikk             | double   | Steady state Sc current
```

Re	double	Earting impedance
Xe	double	Earthing reactance
Ze_activ	double	Factor for earthing impedance Re, Xe
Mue	double	Factor Mue
Pbetr	double	Operating active power
Qbetr	double	Operating reactive power
Uger	double	voltage set value for PV node
UWger	double	voltage angle (for slack node)
Qmin	double	Lower limit for reactive power
Qmax	double	Highest limit for reactive power
Pmin	double	Lower limit for active power
Pmax	double	Highest limit for reactive power
Sl_anteil	double	slack portion
Statik	double	Static
Pv_q_anteil	double	Portion of reactive power for PV-node
c0	double	Costs factor: constant term
c1	double	Costs factor: linear term
c2	double	Costs factor: quadratic term
CostMultipl	double	Multiplication factor for costs
Plim	double	Cos(phi) regulation, limit for active power
Cosphi_oper	double	Operating cos(phi)
Cosphi_min	double	Minimum cos(phi)
Cosphi_max	double	Maximum cos(phi)
H	double	Inertia
R	double	stator resistance for stability
RG	double	Stator resistance for SC
D	double	Mechanical Damping
Xq	double	q-axis synchronous reactance in %
Xq1	double	q-axis transient reactance (unsaturated) in %
Xq2	double	q-axis subtransient reactance (unsaturated) in %
Xp	double	Stator leakage reactance or Potier reactance in %
Xc	double	Characteristic reactance in %
Ia_d	double	Saturation parameter ia for d-axis
Ib_d	double	Saturation parameter ib for d-axis
Ic_d	double	Saturation parameter ic for d-axis

```
A_d             | double   | Saturation parameter A for d-axis
B_d             | double   | Saturation parameter B for d-axis
Ia_q            | double   | Saturation parameter ia for q-axis
Ib_q            | double   | Saturation parameter ib for q-axis
Ic_q            | double   | Saturation parameter ic for q-axis
A_q             | double   | Saturation parameter A for q-axis
B_q             | double   | Saturation parameter B for q-axis
Td1             | double   | d-axis transient short-circuit time constant in s.
Tq1             | double   | q-axis transient short-circuit time constant in s
Td2             | double   | d-axis subtransient short-circuit time constant in s
Tq2             | double   | q-axis subtransient short-circuit time constant in s
Td01            | double   | d-axis transient open-circuit time constant in s
Tq01            | double   | q-axis transient open-circuit time constant in s
Td02            | double   | d-axis subtransient open-circuit time constant in s
Tq02            | double   | q-axis subtransient open-circuit time constant in s
Xl              | double   | EQ circuit parameters in %
Xrc             | double   | EQ circuit parameters in %
Xad             | double   | EQ circuit parameters in %
Xfd             | double   | EQ circuit parameters in %
Rfd             | double   | EQ circuit parameters in %
Xaq             | double   | EQ circuit parameters in %
X1d             | double   | EQ circuit parameters in %
R1d             | double   | EQ circuit parameters in %
X1q             | double   | EQ circuit parameters in %
R1q             | double   | EQ circuit parameters in %
X2q             | double   | EQ circuit parameters in %
R2q             | double   | EQ circuit parameters in %
InitTeta0       | double   | Initial teta value
InitU0          | double   | Initial U0
InitF0          | double   | Initial F0
Sim_fak_LoadBalance| double    | Load balancing factor P
Sim_fak_LoadBalanceQ| double    | Load balancing factor Q
pUrG            | double   | Deviation from Urg
Ufmzuufr        | integer  | Ratio of max. to rated excitation voltage (0="1.3", 1="1.6", 2="2.0")
Erdtype         | integer  | Earthing type (0: direct, 1: impedance, 2: isolated)
```

```
      Lftype          | integer  | Node type for LF (0=PQ, 1=PV, 2=SL, 3=PC)
      ControlType     | integer  | Cos(phi) regulation (0:no, 1:cos(phi) charact., 2:Q-reg, 3:QP-reg
      E_model         | integer  | Model for Stability: (0:classical, 1:transient, 2:subtransient, 3: SIMPOW type 1,
4: SIMPOW ST 33, 5: SIMPOW SP1
      MachineType     | integer  | Machine type (0=salient pole, 1=round rotor)
      SatType         | integer  | Saturation type: 0=parameter, 1=curves
      SatParType_d    | integer  | Saturation parameter type for d-axis: (0:ia,ib,ic, 1:A,B
      SatParType_q    | integer  | Saturation parameter type for q-axis: (0:ia,ib,ic, 1:A,B
      Tim_const_sc    | integer  | Time Constant Type: (0=open circuit, 1=short circuit)
      RelPriority     | integer  | Priority for power station re-connection
      RelLoadCurve    | integer  | Load curve index (RAMSES system states)
      StUpPrio        | integer  | Start-up priority (1..10)
      RefYear         | integer  | Reference year
      HarmSource      | String   | Harmonic source type
      KwGen           | BOOL     | power station unit
      Daempferw       | BOOL     | Damper winding?
      Motor           | BOOL     | Operation as motor ?
      Capacitive_oper| BOOL     | Operating cos(phi) is capacitive?
      Capacitive_min  | BOOL     | Minimum cos(phi) is capacitive?
      Capacitive_max  | BOOL     | Maximum cos(phi) is capacitive?
      Conversion      | BOOL     | Internal conversion to EQ circuit parameter
      ConsiderSat     | BOOL     | Has saturation to be considered?
      CapabilityCurve| BOOL     | Capability curve active?
      TorquePower     | BOOL     | Torque power considered
      RefMachine      | BOOL     | Machine is reference machine
      GeneratorAsHarmSource | BOOL     | Is SM a harmonic current source
      ReadMeasurementData | BOOL     | Read measurement data?

  //{{DESCRIPTION_PART(CNPTechFuseElec)
      Ir              | double   | rated current

  //{{DESCRIPTION_PART(CNPTechFilterElec)
      Ur              | double   | Rated voltage
      R               | double   | Resistance
      L               | double   | Inductance
```

```
    C               | double   | Capacitance
    Cs              | double   | Auxiliary Capacitance
    Df              | double   | Damping factor
    Rd              | double   | Damping resistance
    C_filter        | BOOL     | C filter
    HP_filter       | BOOL     | HP filter
    Isolated        | BOOL     | filter isolated (no ground connection)

//{{DESCRIPTION_PART(CNPTechFeederElec)
    Sk2max          | double   | max. SC power Sk" maximum
    Sk2min          | double   | min. SC power Sk" minimum
    Ik2max          | double   | max. SC current Ik" maximum
    Ik2min          | double   | min. SC current Ik" minimum
    Z0zuZ1_max      | double   | Ratio Z0 / Z1 maximum
    Z0zuZ1_min      | double   | Ratio Z0 / Z1 minimum
    R1zuX1_max      | double   | Ratio R1 / X1 (positive system) maximum
    R1zuX1_min      | double   | Ratio R1 / X1 (positive system)  minimum
    R0zuX0_max      | double   | Ratio R0 / X0 (zero system) max.
    R0zuX0_min      | double   | R0 / X0 (zero system) minimum
    C1              | double   | Capacitance
    Ubetr           | double   | Voltage
    Uwbetr          | double   | Voltage Angle
    Pbetr           | double   | Active power
    Qbetr           | double   | Reactive power
    Sl_anteil       | double   | Slack Portion
    EOper           | double   | Operating voltage
    c0              | double   | Costs factor: constant term
    c1              | double   | Costs factor: linear term
    c2              | double   | Costs factor: quadratic term
    CostMultipl     | double   | Multiplication factor for costs
    Lftype          | integer  | Type: (0=SL, 1=PQ )
    IEC             | BOOL     | Sk2 according to IEC

//{{DESCRIPTION_PART(CNPTechEarthSwitchElec)
    Ir              | double   | Rated current
```

```
    Ipmax           | double   | max. peak current
    Iamax           | double   | max. breaking current
    Ur              | double   | Rated voltage
    Ik2max          | double   | max. Ik"
    R1              | double   | Resistance positive system
    X1              | double   | Reactance positive system
    R0              | double   | Resistance zero system
    X0              | double   | Reactance zero system
    Remote_controll| BOOL     | remote controlled

//{{DESCRIPTION_PART(CNPTechDiscSwitch2Elec)
    Ir              | double   | Rated current
    Ipmax           | double   | max. peak current
    Iamax           | double   | max. breaking current
    Ur              | double   | Rated voltage
    Ik2max          | double   | max. Ik"
    R1              | double   | Resistance positive system
    X1              | double   | Reactance positive system
    R0              | double   | Resistance zero system
    X0              | double   | Reactance zero system
    RelFailureGroup| integer  | Failure group
    ReliabilityType| String   | Reliability type
    ReliabilityIdeal| BOOL     | Element ideal
    Remote_controlled| BOOL     | remote controlled

//{{DESCRIPTION_PART(CNPTechDCVoltageSourceElec)
    USet            | double   | Voltage
    RemoteControlle| BOOL     | remote controlled?

//{{DESCRIPTION_PART(CNPTechDCShuntElec)
    R               | double   | Resistance
    L               | double   | Inductance
    C               | double   | Capacitance
    Lftype          | integer  | Shunt type (0: RC, 1: RL)
```

```
  //{{DESCRIPTION_PART(CNPTechDCReactorElec)
      R               | double   | Resistance
      L               | double   | Inductance

  //{{DESCRIPTION_PART(CNPTechDcLoad)
      Rset            | double   | Load Resistance
      Pset            | double   | Load Power
      Iset            | double   | Load Current
      Lftype          | integer  | Node type (0=P,1=I,2=R)

  //{{DESCRIPTION_PART(CNPTechDcLineElec)
      R_pos           | double   | Positive sequence resistance in Ohm/km or see Units.
      L               | double   | Inductance in mH (SIMPOW)
      Ir_min          | double   | Minimal rated current in A.
      Ir_max          | double   | Maximum rated current in A.
      Units           | integer  | Units for the input values below.
      SimpowDSLArgume| String   |

  //{{DESCRIPTION_PART(CNPTechDCGroundElec)
      R               | double   | Resistance of ground
      L               | double   | Reactance of ground
      C               | double   | Capacitance of ground

  //{{DESCRIPTION_PART(CNPTechVoltageTrafoElec)
      Ur1             | double   | Rated voltage at primary side
      Ur2             | double   | Rated voltage at secondary side

  //{{DESCRIPTION_PART(CNPTechFaultElec) Fault element
      FaultDescription | String   | Fault descrption
      NameInserted     | String   | name of the operator who inserted the fault
      StateInserted    | BOOL     | State inserted is active

//{{DESCRIPTION_PART(CNPTechACGenericCompElec) Dispersed generator
      Uset      | double   |   Voltage set value (operating)
      Uwset     | double   |   Voltage angle set value (operating)
```

```
Pset       | double   |   Active power set value (operating)
Sr         | double   |   Rated apparemt power
Ur         | double   |   Rated voltage
Cosr       | double   |   Rated power factor
Qset       | double   |   Reactive power set value
Iset                  | double   |   Current set value (operating)
Cosset                | double   |   Power factor set value (operating)
Sl_anteil             | double   |   Slack portion
Sim_fak_LoadBalance   | double   |   Simultaneity factor for active power
Sim_fak_LoadBalanceQ  | double   |   Simultaneity factor for reactive power
Sk2max                | double   |   Max. short circuit power
Sk2min                | double   |   Min. short circuit power
R1zuX1_max            | double   |   Ratio R1/X1 for positive sequence system (max. value)
R1zuX1_min            | double   |   Ratio R1/X1 for positive sequence system (min. value)
EOper                 | double   |   EMF (opertaing value) for SC calculation
CosPhiLimCap          | double   |   Power factor limit (capacitive)
CosPhiLimInd          | double   |   Power factor limit (reactive)
Lftype                | integer  |   Loadflow type
StaticGeneratorType   | integer  |      Internal use
Units                 | integer  |   Units for input values
Anzahl                | integer  |   Number of parallel generators
HarmSource            | string   |   Name of Harmonic source
RemoteControlled      | BOOL     |   Is remote controlled?
Sim_fak_var           | BOOL     |    Internal use
Isolated              | BOOL     |   Is isloated (star point)?
SCRueckSpeisung       | BOOL     |   Is reversible (for SC calculation)?
Srspsg                | BOOL     |   Is converter drived?
StaticGenAsHarmSource| BOOL     |   Is harmonic source?
Proz_wert             | BOOL     |   Harmonic currents in % for harmonic source?
IEC                   | BOOL     |   IK"" according to IEC?
F_in_Hz               | BOOL     |   Frequency values in Hetz for harmonic source?
Capacitive            | BOOL     |   Is power factor capacitive?

//////////////////////////////////////////////////////////////////////////////
```

```
//Description of the paramter access for Water, Gas and Disrict Heating Elements
////////////////////////////////////////////////////////////////////////////////

//Description  of the Centrifugal Pump
      H                         | double   | Elevation in m
      Calorific                 | double   | Calorification factor
      Pmax                      | double   |
      Pmin                      | double   |
      Qmax                      | double   |
      Qmin                      | double   |
      Step                      | integer  | Defines the step position of the pump
      DisplayUnit               | integer  | Displaying Units (Gas)
      Qunit                     | integer  | unit for the input value of the load Q (Water,Heating: l/s =0, l/min=1
or m3/h=2, t/h=3, kg/s, kW=4). Gas(m3/h=0 k*m3/h=1)
      UseCharacModel            | BOOL     | If this flag is active the characteristic model will be used instead of
the parabola model

//Description  of the Circulation Pump
      Pmax                      | double   |
      Pmin                      | double   |
      Qmax                      | double   |
      Qmin                      | double   |
      DisplayUnit               | integer  | Displaying Units (Gas)
      Qunit                     | integer  | unit for the input value of the load Q (Water,Heating: l/s =0, l/min=1
or m3/h=2, t/h=3, kg/s, kW=4). Gas(m3/h=0 k*m3/h=1)
      UseCharacModel            | BOOL     | If this flag is active the characteristic model will be used instead of
the parabola model

//Description  of the Fitting element
      D                         | double   | Diameter in mm
      ZetaPlus                  | double   | Zeta in directions from - to node
      m_dZetaMinus              | double   | Zeta in directions to - from node
      DisplayUnit               | integer  | Units

//Description  of the Heat Load
```

```
      Demand                    | double   | Demand in units depending on "Qunit"
      SimultFactor              | double   | Simultaneity factor, which is the same for all heat exchangers having
the same load type
      ReturnTemp                | double   | Return temperature in Celsius
      Qmax                      | double   | Max. possible heating load in kW.
      DeltaPmin                 | double   | Min. pressure difference
      D                         | double   | The diameter of the fitting in mm
      ZetaPlus                  | double   | Zeta in directions from - to node
      ZetaMinus                 | double   | Zeta in directions to - from node
      Qunit                     | integer  | unit for the input value of the load Q (Water,Heating: l/s =0, l/min=1
or m3/h=2, t/h=3, kg/s, kW=4). Gas(m3/h=0 k*m3/h=1)
      LoadType                  | String   | To each heat exchanger a load type can be assigned (e.g. house,
industry, etc.)
      CheckQmax                 | BOOL     | If this flag is on, then the program checks the max. possible flow
      CheckPmin                 | BOOL     | If this flag is on, then the program checks the min. difference pressure
over the heat exchanger
      CheckBackFlow             | BOOL     | If this flag is on, then the program disables a negative flow through
the heat exchanger

//Description  of the Heat Plant
      Temp                      | double   | Feeding temperature in Celsius of the Power Plant
      FeedingPower              | double   | The feeding heating power in kW
      D                         | double   | The diameter of the fitting in mm
      ZetaPlus                  | double   | Zeta in directions from - to node
      ZetaMinus                 | double   | Zeta in directions to - from node
      TempPower                 | integer  | This flag selects, if the power plant feeds with constant
temperature(=0) or constant heating power(=1)

//Description  of the Line Load
      Qdemand                   | double   | Demand in units depending on "Qunit"
      Distance                  | double   | Distance of the load in % or in m from the starting node of the line
      SimultFactor              | double   | Simultaneity factor, which is the same for all consumers having the same
                                           load type
      Tau                       | double   | For gas only: Tau factor
      Zeta                      | double   | Additional Zeta losses
```

```
      Qunit                     | integer  | unit for the input value of the load Q (Water,Heating: l/s =0, l/min=1
                                           or m3/h=2, t/h=3, kg/s, kW=4). Gas(m3/h=0 k*m3/h=1)
      StreetNr                  | integer  | Number of the house, where the load is installed
      StreetAB                  | String   | Addition to the street number
      StreetName                | String   | Street name, where the load is installed
      SimultanityType           | String   | To each consumer a load type can be assigned (e.g. house, industry,
                                           etc.)

//Description  of the Pipe Line
      D                         | double   | Inner diameter of the pipeline in mm.
      Dout                      | double   | Outer diameter of the pipeline in mm.
      Q                         | double   | Demand along the pipeline per km
      K                         | double   | k-value according to Colebrook in mm
      PrsssuereLevel            | double   |
      Vmax                      | double   | Maximum allowable velocity in m/s.
      u                         | double   | Heat-transfer coefficient in W/(mK). This value will only be used in the
                                           district heating module to calculate the heat losses.
      Zeta                      | double   | Additional Zeta losses
      Sound                     | double   | Sound velocity in m/s
      Length                    | double   | Line length
      DisplayUnit               | integer  | Displaying Units (Gas)
      Qunit                     | integer  | unit for the input value of the load Q (Water,Heating: l/s =0, l/min=1
                                           or m3/h=2, t/h=3, kg/s, kW=4). Gas(m3/h=0 k*m3/h=1)
      Year                      | integer  | Year of Installation
      QType                     | String   | To each load along a line a load type can be assigned (e.g. city,
                                           country, center, etc.)
      KType                     | String   | k-Type. This k-type allows all k-values to be changed globally
      Material                  | String   |
      ProfileSelected           | BOOL     | If this option is checked, a profile for this line can be displayed

//Description  of the Gas,Water Node
      H                         | double   | Elevation in m
      Pnom                      | double   | The nominal pressure in bar of the node
      Qdemand                   | double   | Demand in units depending on "Qunit"
```

```
      SimultFactor              | double   | Simultaneity factor, which is the same for all consumers having the same
load type
      Tau                       | double   | For gas only: Tau factor
      Pconst                    | double   | This is the constant pressure which will be applied to this node if
                                           "Calc Type" Pconst is chosen.
      Temp                      | double   | Nominal temperature of the medium at this node
      DisplayUnit               | integer  | Displaying Units (Gas)
      Qunit                     | integer  | unit for the input value of the load Q (Water,Heating: l/s =0, l/min=1
                                               or m3/h=2, t/h=3, kg/s, kW=4). Gas(m3/h=0 k*m3/h=1)
      NodeType                  | integer  | Node type. Four types are possible (possible values: 0,1,2,3)
      NodeType                  | integer  | With this integer the node model can be changed (0=Qconst; 1=Pconst)
      SimultanityType           | String   | To each consumer a load type can be assigned (e.g. house, industry,
etc.)
      MediumTable               | String   | This table defines the temperature dependent physical values of the
medium
      ForwardNet                | BOOL     | Forward/Return Network

//Description  of the Reservoir
      H                         | double   | Elevation of the water in m (m+NN).
      Qmax                      | double   | Maximum water, which the reservoir is able to supply
      InFlow                    | double   | Water supply into the reservoir
      VolumeAt0                 | double   | Water volume at time zero in m3
      ExtinguishWater           | double   | Extinguish water of the reservoir in m3
      A                         | double   | Surface of the reservoir in m2
      Calorific                 | double   | Calorification factor
      DisplayUnit               | integer  | Displaying Units (Gas)
      Qunit                     | integer  | unit for the input value of the load Q (Water,Heating: l/s =0, l/min=1
                                               or m3/h=2, t/h=3, kg/s, kW=4). Gas(m3/h=0 k*m3/h=1)
      RegulatePressure          | BOOL     | If TRUE, the regulator will be simulated with constant pressure which is
                                               indicated in the elevation field otherwis fixed Q will be supplied
      Emptying                  | BOOL     | If TRUE, the reservoir is taken into account during the time simulation
      BlockInFlow               | BOOL     | Flow into the reservoir will be be blocked during simulation
      ConsiderQmax              | BOOL     | If during a simulation the flow Q exceeds this limit, the pressure
                                           regulation will be stopped and the flow will be regulated to this value
```

```
//Description  of the Special Load
      Pressure                  | double   | Value to which the output pressure will be regulated, if the option
                                                "RegulatePressure" is TRUE
      Qdemand                   | double   | Demand in units depending on "Qunit"
      DisplayUnit               | integer  | Displaying Units (Gas)
      Qunit                     | integer  | unit for the input value of the load Q (Water,Heating: l/s =0, l/min=1
                                                or m3/h=2, t/h=3, kg/s, kW=4). Gas(m3/h=0 k*m3/h=1)
      RegulatePressure          | BOOL     | If TURE, then the output pressure will be regulated to value of
                                           "Pressure"

//Description  of the Valve and Pressure Regulator
      Pmax                      | double   | Defines the first point of the valve characteristic together with Qmin.
      Pmin                      | double   | Defines the second point of the valve characteristic together with Qmax
      Qmax                      | double   | Defines the second point of the valve characteristic together with Pmin
      Qmin                      | double   | Defines the first point of the valve characteristic together with Pmax
      Pressure                  | double   | The output or input pressure will be regulated to this value, if the
                                                corresponding regulation type has been selected
      Qregulated                | double   | The flow through the valve will be regulated to this value
      QmaxForPregulation        | double   | If during a simulation the flow Q exceeds this limit, the pressure
                                           regulation will be stopped and the flow will be regulated to this value
      DisplayUnit               | integer  | Displaying Units (Gas)
      Qunit                     | integer  | unit for the input value of the load Q (Water,Heating: l/s =0, l/min=1
                                           or m3/h=2, t/h=3, kg/s, kW=4). Gas(m3/h=0 k*m3/h=1)
      PoutParabelPinpQfix       | integer  | Model Type: 0=Poutput regulation; 1=Parabel; 2:Input regulation; 3=Flow
                                           Q regualtion
      BlockQnegative            | BOOL     | If the flow Q becomes negative during a simulation, then the flow
                                           through the valve will be blocked
      RegulationActive          | BOOL     | If TRUE, then regulation will be deactivated. The equation of the valve
                                           model is then Pout=Pin
```

4d) Appendix : List of the of regulator types

```
////////////////////////////////////////////////////////////////////////////
// Function to add a regulator
// example:
// BOOL bAddedReg = AddRegulator(lAddedElementID, _T("REGULATOR"), _T("NewExciter"),
//                               EXCITER, _T("EXCITERS IEEET4"));
////////////////////////////////////////////////////////////////////////////

BOOL AddRegulator(unsigned long& lElementID, TCHAR* wcElementType, TCHAR* wcElementName,
                  int nRegulatorType, TCHAR* wcRegulatorName)

      Output: returns TRUE if the regulator is added succesfully
      Output: returns the lElementID of the added regulator
      Input: wcElementType is the element type of the regulator
                  _T("REGULATOR")
                  _T("TURBINE")
                  _T("MECHLOAD")
                  _T("INERTIA")
                  _T("TABLE")
      Input: wcElementName, is the name of the regulator to be added
      Input: nRegulatorType is one of the following types
            enum{EXCITER, PSS, U_EXITER_LIMIT, SVC, TRAFO, TRAFO_FI, CAPACITOR,
                   GOVERNOR, FIELD_LIM, STATOR_I_LIM,
                   RESERVED_1, RESERVED_2,
                   HVDC_CR, HVDC_COR, HVDC_VDCOL, HVDC_GR,
                   HVDC_CC, HVDC_DCR, HVDC_FPD,
                   HVDC_N_CFC, HVDC_N_CCA, HVDC_N_COL, HVDC_N_MC, HVDC_N_VCR,PWM_MA,PWM_FI,
                   TURBINE=70, MISC=90
                  };
      Input: wcRegulatorName is the name of the regulator (see regulator library).

////////////////////////////////////////////////////////////////////////////
// Function to change the regulator data
```

```
// example:
// BOOL bOk = SetParameterDouble(lAddedElementID, _T("KR"), 0.0034);
////////////////////////////////////////////////////////////////////////////
BOOL  SetParameterInt(unsigned long lID,TCHAR* wcParameter, int nValue)
BOOL  SetParameterBool(unsigned long lID,TCHAR* wcParameter, bool bValue)
BOOL  SetParameterString(unsigned long lID, TCHAR* wcParameter, TCHAR* wcValue)
BOOL  SetParameterDouble(unsigned long lID,TCHAR* wcParameter, double dValue)

      Output: returns TRUE if the data has changed succesfully
      Input: lAddedElementID, ID of the regulator
      Input: wcParameter, name of the parameter to change (see manual of the regulators)
      Input: dValue, value of the parameter.
```

4e) Appendix : List of the results types

This list will be used for the NPL function:
BOOL GetResultDouble(unsigned long lID,TCHAR* sParameter, double& dValue);

// Gas/Water/District Heating element results

_T("Q")
_T("Plosses")
_T("Jv")
_T("Velocity")
_T("Time")
_T("NoneRegOpenErr")
_T("DispUnit")
_T("TempLosses")
_T("Rho")
_T("Nue")

```
    _T("Temperature")
    _T("kWLosses")
    _T("Cp")
    _T("QHydrant")

// Gas/Water/District Heating node results
    _T("Q")
    _T("AbsolutePressure")
    _T("OperatingPressure")
    _T("PressureLosses")
    _T("Buoyancy")
    _T("TimeMin")
    _T("TimeMax")
    _T("DisplayUnit")
    _T("Temperature")
    _T("Rho")
    _T("Cp")
    _T("QFire")
    _T("PFire")
    _T("FireConvergeInfo")
    _T("PFireMin")
    _T("PminBusID")
    _T("VmaxElemID")
    _T("VmaxFire")
```

// Power system element results

The following list shows the available power system element results variables. To get the results at the specific side of the element the values “–x1”, “-x2”, “-x3”, “-x4” should be added to each variable. To get the power P at node two of a line the following variable must be specified: _T(“P-x2”).

```
_T("P")
_T("Q")
_T("I")
_T("Iangle")
_T("Plosses")
_T("Qlosses")
_T("Pcomp")
_T("Qcomp")
_T("UMagOpenEnd")
_T("UAngOpenEnd")
_T("ElementLoading")
_T("PhaseL1Feeded")
_T("PhaseL2Feeded")
_T("PhaseL3Feeded")
_T("PL2")
_T("QL2")
_T("IL2")
_T("IangL2")
_T("PlossesL2")
_T("QlossesL2")
_T("PL3")
_T("QL3")
_T("IL3")
_T("IangL3")
```

```
_T("PlossesL3")
_T("QlossesL3")
```

// Power system node results load flow

```
_T("V")
_T("V_PC")
_T("ANG")
_T("LossSensP")
_T("PLossSensQ")
_T("PGen")
_T("QGen")
_T("PLoad")
_T("QLoad")
_T("QShunt")
_T("PhaseL1Feeded")
_T("PhaseL2Feeded")
_T("PhaseL3Feeded")
_T("V_L2")
_T("V_PC_L2")
_T("ANG_L2")
_T("PGen_L2")
_T("QGen_L2")
_T("PLoad_L2")
_T("QLoad_L2")
_T("QShunt_L2")
_T("V_L3")
_T("V_PC_L3")
_T("ANG_L3")
```

_T("PGen_L3")
_T("QGen_L3")
_T("PLoad_L3")
_T("QLoad_L3")
_T("QShunt_L3")

// Power system short circuit results

_T("Sk2")
_T("Ik2_R")
_T("Ik2_Ang_R")
_T("Ip_R")
_T("Ib_R")
_T("Ith_R")
_T("ID_R")
_T("IAsy_R")

// Power system reliability analysis results

Element results (GetResultDouble):

_T("RE-F")
_T("RE-Q")
_T("RE-P")
_T("RE-W")
_T("RE-C")
_T("RE-T")

System results (GetResultSummaryDouble):

_T("RE-SysLoad-F")
_T("RE-SysLoad-Q")
_T("RE-SysLoad-T")
_T("RE-SysLoad-W")
_T("RE-SysLoad-P")
_T("RE-SysLoad-C")
_T("RE-SysGen-F")
_T("RE-SysGen-Q")
_T("RE-SysGen-T")
_T("RE-SysGen-W")
_T("RE-SysGen-P")
_T("RE-SysGen-C")
_T("RE-SAIDI")
_T("RE-SAIFI")
_T("RE-CAIDI")
_T("RE-ASAI")
_T("RE-ASIDI")

System results (GetResultSummaryInt):

_T("RE-TotNumCust")

4f) Appendix : List of the import/export file types

The import/export types will be used by the following import/export library functions:

BOOL ImportFileElectrical(TCHAR* wcFileName, int nFileType, TCHAR* wcProtectionLibName=NULL, TCHAR* wcNeplanLibName=NULL);
BOOL ExportFileElectrical(TCHAR* wcFileName, int nFileType);
BOOL ImportFileGasWater(TCHAR* wcFileName, int nFileType, TCHAR* wcNeplanLibName=NULL);
BOOL ExportFileGasWater(TCHAR* wcFileName, int nFileType); BOOL ImportFileElectrical

```
// list of import/export types
#define  EDT_NEPLAN_FILE 0
#define  NDT_NEPLAN_FILE 1
#define  NDB_NEPLAN_FILE 2
#define  ZDB_NEPLAN_FILE 3
#define  UCTE_NEPLAN_FILE 4
#define  GIS_NEPLAN_FILE 6
#define  WET_NEPLAN_FILE 10
#define  WKT_NEPLAN_FILE 11
#define  GRT_NEPLAN_FILE 12
#define  MCB_NEPLAN_FILE 13
```

4f) Appendix : List of the calculation parameters

```
////////////////////////////////////////////////////////////////////////////
//Description of the calculation paramter
////////////////////////////////////////////////////////////////////////////
_T("PARAM_GW")               | gas and water calculation
_T("PARAM_GWTS")             | gas and water time simulation
_T("PARAM_GWFIRE")           | water fire fighting analysis
_T("PARAM_ELEC")             | electrical, load and generator factors
_T("PARAM_LF")               | electrical, load flow
_T("PARAM_SC")               | electrical, short circuit calculation
_T("PARAM_TS")               | electrical, transient stability
_T("PARAM_SS")               | electrical, samll singnal stability
_T("PARAM_VS")               | electrical, voltage stability
_T("PARAM_MS")               | electrical, motor starting
_T("PARAM_OPF")              | electrical, optimal power flow
_T("PARAM_ATC")              | electrical, NTC calculation
_T("PARAM_CONT")             | electrical, contingency analysis
_T("PARAM_SIMPOW")           | electrical, Simpow dynamic analysis
_T("PARAM_SIMPOW_LA")        | electrical, Simpow linear analysis
_T("PARAM_DACF")            | electrical, DACF analysis
_T("PARAM_DP")              | electrical, disatnace protection analysis
_T("PARAM_RE")               | electrical, reliability analysis
_T("PARAM_HA")               | electrical, harmonic analysis
_T("PARAM_OV")               | electrical, optimal distribution analysis
_T("PARAM_TO")               | electrical, optimal separation point analysis
_T("PARAM_LOP")              | electrical, load profile analysis
_T("PARAM_CP")               | electrical, optimal capacitor placement
_T("PARAM_NR")               | electrical, network reduction
_T("PARAM_SEL")              | electrical, overcurrent selectivity analysis
_T("PARAM_INVA")             | electrical, investment analysis
_T("PARAM_FF")               | electrical, fault finding analysis
_T("PARAM_FERE")             | electrical, feeder reinforcement analysis
_T("PARAM_RESUPPLY")         | electrical, load restoration analysis
```

```
////////////////////////////////////////////////////////////////////////////
//Description of the gas, water and district heating calculation paramters
////////////////////////////////////////////////////////////////////////////
      LoadFactor                | double   | Load factor (all demands will be multiplied by this factor)
      Ny                        | double   | Cinematic viscosity in m2/s
      Temp                      | double   | Temperature in Celsius
      DensityRatio              | double   | Gas: Density ratio
      GasCalcTemp               | double   | Gas: Tau dependent caluclation temperature (in Celsius)
      Rho                       | double   | Water / District Heating: Density (kg/m3)
      AlgorithmNewtonCross      | integer  | Algorithm to be used during calcultaion (0=Newton Raphson; 1= Hardy
Cross)
      ResultFileName            | String   | Result file name

////////////////////////////////////////////////////////////////////////////
//Description of the electrical network scaling factors
////////////////////////////////////////////////////////////////////////////
      ScalingFactorGenP         | double   | Network scaling factor for P generators
      ScalingFactorLoadP        | double   | Network scaling factor for P loads
      ScalingFactorGenQ         | double   | Network scaling factor for Q generators
      ScalingFactorLoadQ        | double   | Network scaling factor for Q loads
      ScalingFactorShuntInd     | double   | Network inductive scaling factor for shunts
      ScalingFactorShuntCap     | double   | Network capacitive factor for shunts

////////////////////////////////////////////////////////////////////////////
//Description of the load flow parameters
////////////////////////////////////////////////////////////////////////////
      LF-MaxIteration           | integer | max. Number of iterations
      Lf-CalcMethod            | integer | calculation method (0=Ext.Newton-Raphson; 1=Current Iteration
                                                              2=Newton-Raphson; 3=Voltage Drop; 4=DC Load Flow
      LF-ExportFileName         | String   | Name of LF result file name
      LF-DoLoadBalancingCalculation | BOOL     | load flow calculation with load balanceing
      LF-DoAsymmerticalLoadFlow| BOOL     | asymmetrical load flow calculation for asymmetrical networks
      LF-LoadBalancingMode      | integer  | Load balancing mode
      LF-LoadingCheckMinMedMax | integer  | Elemnt loading check (min, middle, max)
```

```
      LF-UseOnLoadTapchanger    | BOOL      | calculate with on load tap changer
      LF-SetOnLoadTapchanger    | BOOL      | set the calculated on load tap changers after the calculation
                                              to the transformer element data
      LF-UseTransformerPhaseShift| BOOL     | consider the phase shift of the transformers
      LF-PrintYMatrix           | BOOL      | Print Y matrix after load flow calculation
      LF-WriteResultFile        | BOOL      | Write result file after load flow calculation
      LF-PrintOutWarnings       | BOOL      | Print out warnings during load flow calculation
      LF-MinPrintOut            | BOOL      | Print out minimal informnation during load flow calculation
      LF-Mismatch               | double    | LF mismatch


////////////////////////////////////////////////////////////////////////////
//Description of the short circuit parameters
////////////////////////////////////////////////////////////////////////////
      IthDuration               | double    | Duration in s for thermal current calculation
      IdcDuration               | double    | Duration in s for DC current in IDC calculation
      Cfactor                   | double    | C factor for IEC
      IbDelayTime               | double    | Delay time for CB in s for breakin current calculation Ib
      MaxLoading                | double    | Max. loading of elements in %
      FaultType                 | integer   | Fault  type (0=3-Phase; 1=1-Phase to ground; 2=2-Phase
                                              3= 2-Phase to ground)
      ScCalcMethod              | integer   | Calculation method (0=IEC60909/2001; 1= IEC909/1988; 2=superp.
                                              without LF; 3=superp. with LF; 4=ANSI 37.10; 5=  ANS 37.13
      FaultDistance             | integer   | Fault distance
      ExportFileName            | String    | Name of result file name
      CalcI2max                 | BOOL      | Calculate Ik" max.

////////////////////////////////////////////////////////////////////////////
//Description of the load restoration parameters
////////////////////////////////////////////////////////////////////////////
      SelectedPlan                  | integer   | The actual selected plan
      LabelSize                     | integer   | The size of the label
      WidthAffectedElement          | integer   | Line width of the affected elements
      ResultFileName                | String    | Result file name
      ResultTextFileName            | String    | Result text file name
```

```
      Name                           | String   | Name of the restoration
      Description                    | String   | Description of the restoration
      OperatorNameInserted           | String   | Name of the operator which inserted the fault
      OperatorNameIsolated           | String   | Name of the operator which isolated the fault
      OperatorNameResupplied         | String   | Name of the operator which normalized the network
      ShowParamDlgBeforeAnalysis     | BOOL     | Show optimization selection dialog before analysis
      ShowReSuppPlanDlgAfterCalc     | BOOL     | Show selection plan dialog after analysis
      ShowReSuppStateDlgAfterCalc    | BOOL     | Show the dialog for the state selection after analysis
      StateInserted                  | BOOL     | If TRUE, then the state „inserted" is active
      StateIsolated                  | BOOL     | If TRUE, then the state „isolated" is active
      StateResupplied                | BOOL     | If TRUE, then the state „load restored" is active
      StateNormalized                | BOOL     | If TRUE, then the state „network is normalized" is active
      CalcLosses                     | BOOL     | Run analysis with minimization of the losses
      CalcOverloads                  | BOOL     | Run analysis with minimization of the overloads
      CalcLoading                    | BOOL     | Run analysis with minimization of the element loading
      CalcVoltages                   | BOOL     | Run analysis with maximization of the node voltages
      ChangeSwitchesAutomaticallay   | BOOL     | Change switches automatically if the user changes the stages

////////////////////////////////////////////////////////////////////////////
//Description of the reliability calculation parameters
////////////////////////////////////////////////////////////////////////////
      RE-DefSwBayRemote              | BOOL     | Remot controlled switch bay
      RE-PartialNetworkUseNeighbours | BOOL     | Use neighbours for resupply
      RE-UnderVoltLoadShedding       | BOOL     | Under voltage load shedding
      RE-TimeDepLoadLim              | BOOL     | Time dependet loading limits
      RE-LoadStates                  | BOOL     | Use load characteristics
      RE-LoadDurationCurves          | BOOL     | Use laod duration curves
      RE-TableShowZero               | BOOL     | Show zeros in table
      RE-FilterOutages               | BOOL     | Filter outages
      RE-FilterLoadsGen              | BOOL     | Filter loads and generators
      RE-FilterByFMode               | BOOL     | Filter by F modes
      RE-FilterByTOut                | BOOL     | Filter by T outage
      RE-FEADispLoads                | BOOL     | Show FEA loads
      RE-FEADispCongestions          | BOOL     | Show FEA congestions
      RE-FEAFilterLoads              | BOOL     | FEA filter loads
```

```
RE-FEAFilterFailingEle        | BOOL     | FEA filter failed elements
RE-MultFailSameGroup          | BOOL     | Failed elements belong to the same failure group
RE-MultFailGalvConn           | BOOL     | Failed elements must be galvanically connected
RE-FaultLocSimulate           | BOOL     | Simulate fault locating process
RE-FaultLocRestoration        | BOOL     | Try restoration during fault locating
RE-BusbarDataSwitchbayDep     | BOOL     | Busbar data dependent on number of switchgear
RE-FailureModeSingleShort     | BOOL     | Single independent failure, short
RE-FailureModeSingleLong      | BOOL     | Single independent failure, long
RE-FailureModeManualDisconDelayed   | BOOL     | Manual disconnection, delayed
RE-FailureModeManualDisconPromp     | BOOL     | Manual disconnection, prompt
RE-FailureModeCommonMode      | BOOL     | Common Mode failure
RE-FailureModeLineToGround    | BOOL     | Line-to-ground failure
RE-FailureModeSwitcOpenUnintend  | BOOL     | Unintendent switch opening
RE-FailureModeMultipleShort   | BOOL     | Multiple independent failures
RE-FailureModeSinglePlusMaint | BOOL     | Single independent failure + determined outage
RE-FailureModeSinglePlusManual| BOOL     | Single independent failure + manual diconnection
RE-FailureModeSinglePlusCM    | BOOL     | Single independent failure + common mode
RE-FailureModeSinglePlusLTGF  | BOOL     | Single independent failure + line-to-ground failure
RE-FailureModeMaintPlusManualDiscon | BOOL     | Determined outage + manual disconnection
RE-FailureModeMaintPlusCM     | BOOL     | Determined outage + Common mode
RE-FailureModeMaintPlusLTGF   | BOOL     | Determined outage + line-to-ground failure
RE-FailureModeMultipleManualDiscon  | BOOL     | Multiple manual disconnection
RE-FailureModeManualDisconCM  | BOOL     | Manual disconnection + common mode
RE-FailureModeManualDisconLTGF  | BOOL     | Manual disconnection + line-to-ground failure
RE-FailureModeMultipleCM      | BOOL     | Multiple common mode failures
RE-FailureModeCMPlusLTGF      | BOOL     | Common mode + line-to-ground failure
RE-FailureModeMultipleLTGF    | BOOL     | Muliple  + line-to-ground failure
RE-FailureModeSinglePlusProtection  | BOOL     | Single independent failure + plus protection
RE-FailureModeSinglePlusOverfunction| BOOL     | Single independent failure + plus overfunction
RE-MinimumStateProbability    | double   | Minimum state probability
RE-TimeSwitchOpenRemote       | double   | Duration for remote switching
RE-TimeSwitchOpenLocal        | double   | Duration for manual switching
RE-VoltageSagU1               | double   | VoltageSagU1 (later used)
RE-VoltageSagU2               | double   | VoltageSagU2 (later used)
RE-VoltageSagT1               | double   | VoltageSagT1 (later used)
```

```
RE-VoltageSagT2               | double   | VoltageSagT2 (later used)
RE-DefaultTrippingTimeCB      | double   | Default tripping time CB
RE-TOutMin                    | double   | Filter option for result display (min)
RE-TOutMax                    | double   | Filter option for result display (max)
RE-LoadSheddingMinStep        | double   | Load shedding min step
RE-FaultLocTravelTimeFirst    | double   | Fault location travel time to first station
RE-FaultLocTravelTime         | double   | Fault location travel time between 2 stations
RE-FaultLocMeasTime           | double   | Fault location: time for measurements
RE-FaultLocEmergencyTime      | double   | Fault location: for emergency poer supply
RE-FaultLocSwitchTime         | double   | Fault location: switching time
RE-FaultLocAccessTime         | double   | Fault location: access time
RE-FaultLocAutoLocAccuracy    | double   | Fault location: accuracy for fault location


RE-TypeLfAlgorithm            | integer  | Type of Load flow algorithm
                                           (Load-flow algorithm - 0:ConnCheck  1:CapacityFlow  2:AC-LF)
RE-LFNotConverged             | integer  | Algorithem in cas LF does not converge
RE-DefaultSwitchBayConfig     | integer  | Default switch bay configuration
RE-FaultLocIndicator          | integer  | Type of short circuit indicator
```

5) Appendix : List of the variant parameters

```
////////////////////////////////////////////////////////////////////////////
//Description of the variant paramters
////////////////////////////////////////////////////////////////////////////
      DoPartialNetworkColoring      | BOOL     | Color Network according to partial networks
      DoPhasesColoring              | BOOL     | Color Network according to phases
      DoGalvanicNetworkColoring     | BOOL     | Color Network according to galvanic separated networks
      VariantName                   | String   | Name of the variant
      VariantDescription            | String   | Description of the variant
      GraphicFileName               | String   | Name of the graphic file which will be loaded for that variant
      LoadFileName                  | String   | Name of the load data file which will be loaded for that variant
      TopologyFileName              | String   | Name of the topology file which will be loaded for that variant
```

6) Appendix : RunAnalysisTD inputs

```
////////////////////////////////////////////////////////////////////////////
//Description of the inputs of RunAnalysisTD function
////////////////////////////////////////////////////////////////////////////
```

The inputs of the function RunAnalysisTD are the following:

DOUBLE dTEND: simulation time, in seconds.

TCHAR* wcDisturbanceFilePathName: full path of disturbance file (see below for templates); if NULL the disturbances defined in the NEPLAN file will be considered.

TCHAR* wcPlotFilePathName: full path of screenplot file (see below for templates); if NULL the screenplots defined in the NEPLAN file will be considered.

TCHAR* wcResFileName: full path of result file, where the calculations results will be written.

TCHAR* wcDynamicDataFileName: full path of regulator dynamic data file (see below for templates); NULL if no changes are needed.

```
////////////////////////////////////////////////////////////////////////////////
//Templates of disturbance file (*.nepdist)
////////////////////////////////////////////////////////////////////////////////
TEMPLATE 1
disturbance={};
entry={name="Bus2",disturbancename="DIS_BUS_SETTING_3_PHASE_SC_FAULT",parametername="FAULT3PG",newvalue=1,time=0.1}
entry={name="Bus2",disturbancename="DIS_BUS_SETTING_3_PHASE_SC_FAULT",parametername="XFAULT",newvalue=0.01,time=0.1}
entry={name="Bus2",disturbancename="DIS_BUS_REMOVING_SHORT_CIRCUIT_FAULT",parametername="FAULT3PG",newvalue=0,time=0
.2}

TEMPLATE 2
disturbance={};
entry={name="AVR-GEN",disturbancename="DIS_CHANGE_M0DEL_PARAMETERS",parametername="VSTEP",newvalue=-0.05,time=0.5}


////////////////////////////////////////////////////////////////////////////////
//Templates of screenplot file (*.nepplts)
////////////////////////////////////////////////////////////////////////////////
TEMPLATE 1
variable={name="GEN",varname="EFD",position=1,fileoutput=1}
variable={name="GEN",varname="VT",position=2,fileoutput=1}

TEMPLATE 2
variable={name="GEN",varname="EFD",position=1,fileoutput=1}
variable={name="GEN",varname="VT",vartype="TDVAR_U",conversionFrom="USERDEF_BASE",conversionTo="SIUNIT_BASE",unitcon
version=1,invertsign=1,position=2,fileoutput=1,baseFrom=100,baseTo=200}
```

```
////////////////////////////////////////////////////////////////////////////
//Templates of regulator dynamic data file (*.nepctrl)
////////////////////////////////////////////////////////////////////////////
TEMPLATE 1
regulator={name="AVR-GEN",controller="AVR",type="EXCITER
AC1A",element="GEN",TR=0.0,TB=0.001,TC=0.001,KA=400,TA=0.02,VAMAX=14.5,VAMIN=-
14.5,TE=0.8,KF=0.03,TF=1.0,KC=0.2,KD=0.38,KE=1,E1=3.14,SE1=0.03,E2=4.18,SE2=0.1,VRMAX=6.03,VRMIN=-5.43}

TEMPLATE 2
regulator={name="AVR-GEN",controller="AVR",type="EXCITER
AC2A",element="GEN",TR=0.0,TB=0.001,TC=0.001,KA=400,TA=0.01,VAMAX=8,VAMIN=-8,KB=25,VRMAX=105,VRMIN=-
95,TE=0.6,VFEMAX=4.4,KH=1.0,KF=0.03,TF=1,KC=0.28,KD=0.35,KE=1,E1=3.3,SE1=0.012,E2=4.4,SE2=0.037}
```