PSS®SINCAL Datenbankinterface und Automatisierung

Dieses Dokument erläutert den Aufbau und die Struktur der PSS SINCAL Datenbank und zeigt, wie die Datenbank mit externen Programmen gefüllt werden kann. Die Automatisierungsfunktionen der Berechnungsmethoden werden ebenfalls erläutert.

1 Allgemeines

Zu den wichtigsten Eigenschaften von PSS SINCAL zählt die vollständige Transparenz der Daten. Diese Transparenz wird durch Speicherung aller für die Netzplanung erforderlichen Daten in einer relationalen Datenbank erreicht. In PSS SINCAL wird die Datenbank als zentrales Speichermedium für alle Daten verwendet und nicht so wie bei anderen Netzplanungssystemen nur beim "Export" befüllt.

Durch die offene Architektur der Berechnungsmethoden können diese auch relativ einfach für **individuelle Lösungen** genutzt werden. Hierbei wird im Normalfall ein bestehendes PSS SINCAL Netz verwendet, welches mit einem selbst erstellen "Tool" analysiert wird. Das "Tool" kann dabei den kompletten Funktionsumfang der PSS SINCAL Berechnungsmethoden nutzen. Zur Realisierung der individuellen Lösung kann jede beliebige Programmier- oder Scriptsprache verwendet werden (die einzige Voraussetzung ist, dass der Zugriff auf COM-Funktionen unterstützt werden muss).

Durch die Verwendung einer relationalen Datenbank ist auch eine **Kopplung zu einem Geografischen Informationssystem** (GIS), einem Netzinformationssystem (NIS) oder einem Netzleitsystem einfach realisierbar. Im Wesentlichen kann hier zwischen einer reinen Berechnungslösung und einem vollständigen Datenexport zu PSS SINCAL unterschieden werden. Die Wahl der Kopplungslösung hängt von den jeweiligen Zielsetzungen ab.

Reine Berechnungslösung

Die reine Berechnungslösung wird auch als Engine-Lösung bezeichnet. Hierbei wird eine Kopplung für den häufigen Bedarf von Basis-Berechnungsmethoden (wie z.B. Lastfluss und Kurzschluss) implementiert. Damit kann ein großer Anteil aller Berechnungen aus dem Quellsystem heraus gestartet und bedient werden. Die Datenhaltung und auch die Präsentation der Ergebnisse, die Darstellung von Ergebniswerten und Einfärbungen erfolgen in diesem Fall im Quellsystem.

Bei dieser Lösung werden nur die technischen Daten aus dem Quellsystem zu PSS SINCAL exportiert. Anschließend wird mittels Automatisierung die gewünschte PSS SINCAL Berechnungsmethode gestartet und die Berechnungsergebnisse werden dann im Quellsystem visualisiert.

Datenexport zu PSS SINCAL

Hierbei werden die Daten aus dem Quellsystem in die PSS SINCAL Datenbank exportiert. Es werden dabei im Normalfall sowohl die technischen Daten der Betriebsmittel als auch die grafischen Standortdaten exportiert.

Bei dieser Lösung erfolgt die Planung und Auswertung der Netze mit PSS SINCAL. Dabei kann der vollständige Funktionsumfanges des Produktes ausgenutzt werden.

Das folgende Bild zeigt die Kopplungslösung von Mettenmeier GmbH (siehe Referenz – Vorgefertigte Kopplungslösungen). Hierbei handelt es sich um eine Anbindung an das Smallworld GIS. Das Bild zeigt jeweils den gleichen Netzbereich im GIS und in PSS SINCAL.

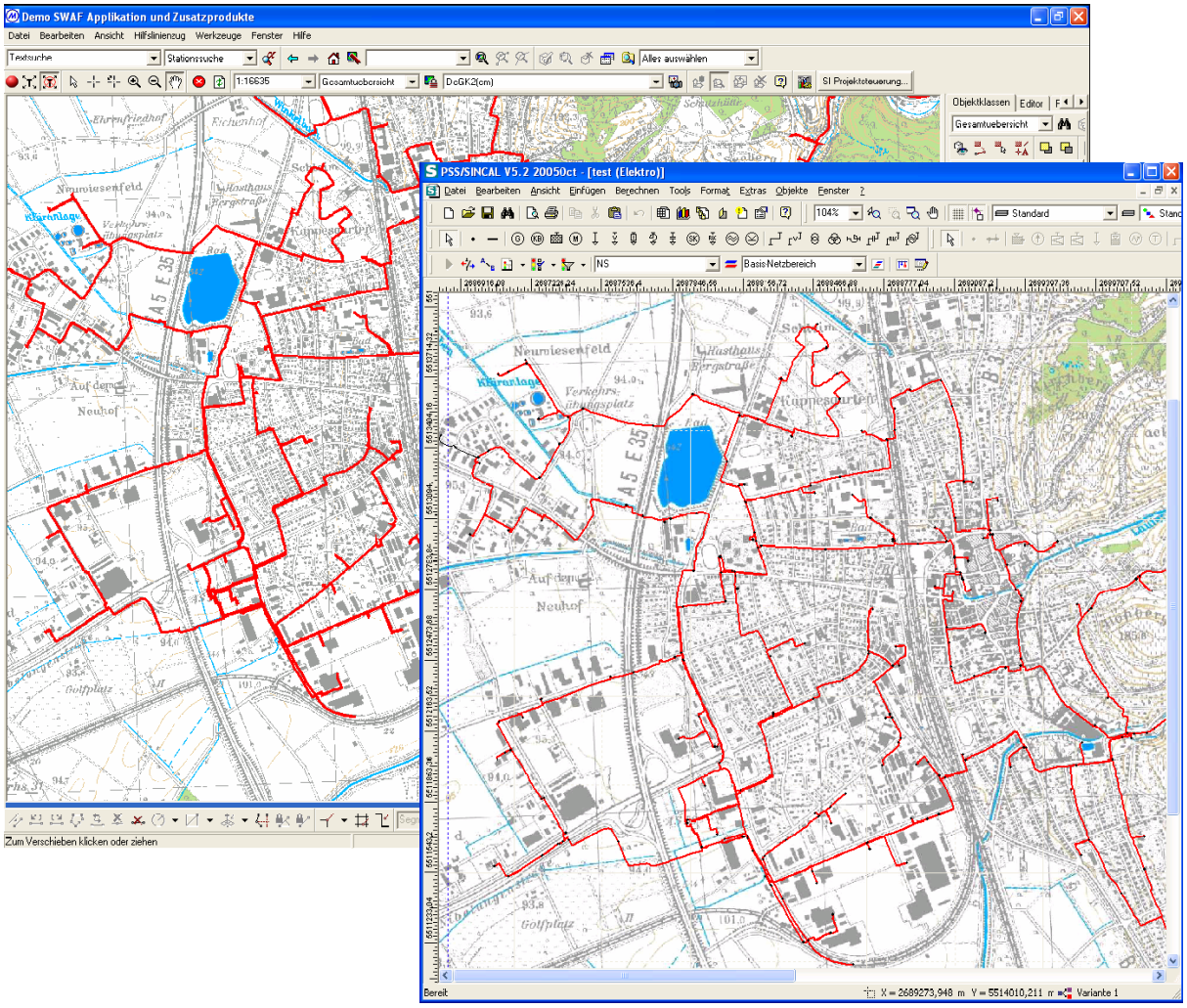


Bild: Smallworld Kopplung von Mettenmeier GmbH

Im Allgemeinen erfordert diese Kopplungslösung einen höheren Implementierungsaufwand, da neben den technischen Daten der Betriebsmittel auch die entsprechende Netzgrafik generiert werden muss. Auch den passenden Detaillierungsgrad zu finden ist nicht immer einfach: So ist die Detaillierung im GIS im Normalfall um ein Vielfaches höher als in einem Netzplanungssystem. Die Vielzahl der Daten wäre sogar störend beim Planen. D.h. es muss bei dieser Kopplung der passende Detaillierungsgrad gefunden werden, damit die Bearbeitung und Planung in PSS SINCAL produktiv möglich ist.

Die Vorteile dieser Lösung liegen aber vor allem in der universellen Nutzbarkeit. Denn hier kann – wie bereits erwähnt – der volle Funktionsumfang von PSS SINCAL zur Planung und Auswertung der Netze genutzt werden.

2 Aufbau der PSS SINCAL Datenbank

In diesem Kapitel wird der Aufbau der PSS SINCAL Datenbank beschrieben und anhand eines kleinen Beispielnetzes wird das Datenmodell detailliert erläutert.

2.1 PSS SINCAL Netze

Ein PSS SINCAL Netz wird durch ein Ordnerpaar, bestehend aus einer SIN Datei sowie einem zusätzlichen Verzeichnis, gebildet.

- {Netzname}.sin
- {Netzname}_files

Die Datei mit der Endung ".sin" ist eine Hilfsdatei der PSS SINCAL Benutzeroberfläche, um die Verwaltung von Netzen zu vereinfachen. In dieser Datei werden verschiedenste netzspezifische Einstellungen der Benutzeroberfläche sowie die Hilfsgrafikobjekte des Netzes gespeichert. Beim Öffnen eines Netzes in der PSS SINCAL Benutzeroberfläche wird diese Datei ausgewählt.

Das Verzeichnis mit dem Suffix "_files" enthält alle weiteren Netzdaten. In diesem Verzeichnis werden die eigentliche Netzdatenbank, die Diagrammdateien und auch verschiedenste Ergebnisdateien und Log-Files gespeichert.

```
Example Ele1.sin
Example Ele1_files
|   database.ini
|   database.mdb
|   database.dia
|
\---NETO
        network.bat
        network.ctl
        ...
```

Die Datei mit der Endung ".ini" ist eine Konfigurationsdatei. Damit kann konfiguriert werden, wie die Datenbank von PSS SINCAL verwendet wird.

Die Datei mit der Endung ".mdb" ist die eigentliche Datenbank im Microsoft Access Format. In dieser sind alle Daten, die das Netz beschreiben, gespeichert.

Falls ein Server Datenbanksystem wie z.B. Oracle oder SQL Server verwendet wird, dann ist die ".mdb" Datei nicht vorhanden. In diesem Fall werden die Netzdaten direkt in der zentralen Server-Datenbank gespeichert.

Unterstützte Datenbanksysteme

Derzeit werden von PSS SINCAL folgende Datenbanksysteme unterstützt:

- Microsoft Access (2003, 2007 und 2010)
- Oracle 9i, 10g und 11g
- SQL Server Express 2008 und 2008 R2

- SQL Server 2008 und 2008 R2

2.2 Allgemeine Designrichtlinien im Datenmodell

Das Datenmodell wurde nach folgenden Gesichtspunkten realisiert:

- Das Datenmodell ist objektorientiert.
- Alle Objekte sind eindeutig identifizierbar und über Primärschlüssel ansprechbar.
- Durch die weitgehende Verwendung eines normalisierten Entwurfes werden einerseits Redundanzen vermieden, andererseits wird die Konsistenzüberprüfung und -sicherung vereinfacht.
- Das Datenmodell ist in verschiedene Kategorien gegliedert, um Auswertungen und Verwaltung zu vereinfachen.
- Das gewählte Datenmodell kann ohne Einschränkungen in jedem relationalen Datenbank-Management-System implementiert werden.

Tabellennamen

Für die Tabellennamen wurden aussagekräftige englische Begriffe gewählt. Zur besseren Lesbarkeit wird Groß-/Kleinschreibung verwendet. Im Wesentlichen wird bei den Tabellennamen zwischen Eingabedaten, Grafikdaten und Ergebnissen unterschieden.

Für die Tabellenbezeichnung der **Eingabedaten** wurden möglichst einfache, aber aussagekräftige Begriffe gewählt:

- Line
- Infeeder
- TwoWindingTransformer
- usw.

Die Tabellenzeichnungen der **Grafikdaten** beinhalten den Präfix "Graphic" im Namen:

- GraphicElement
- GraphicLayer
- usw.

Die Tabellenbezeichnung der **Ergebnisdaten** enthalten als Präfix eine Kurzbezeichnung des Berechnungsverfahren und als Suffix den Begriff "Result":

- LFNodeResult
- LFBranchResult
- SC1NodeResult
- SC1BranchResult

- usw.

Schlüssel

Die Tabellen des Datenmodells enthalten zur eindeutigen Identifikation der Daten **Primärschlüssel**. Der Primärschlüssel in einer Tabelle beinhaltet den Namen der Tabelle mit dem Zusatz "_ID".

Die Beziehungen im Datenmodell werden durch **Fremdschlüssel** realisiert. Die Fremdschlüssel enthalten den Namen der bezogen Tabelle mit dem Zusatz "_ID".

Die Schlüssel sind immer mit dem Datentyp "Long Integer" realisiert. Beispiele für Primär- und Fremdschlüssel:

- Tabelle **Element**
  Primärschlüssel: Element_ID
  Fremdschlüssel: VoltLevel_ID, Variant_ID
- Tabelle **Terminal**
  Primärschlüssel: Terminal_ID
  Fremdschlüssel: Element_ID, Node_ID

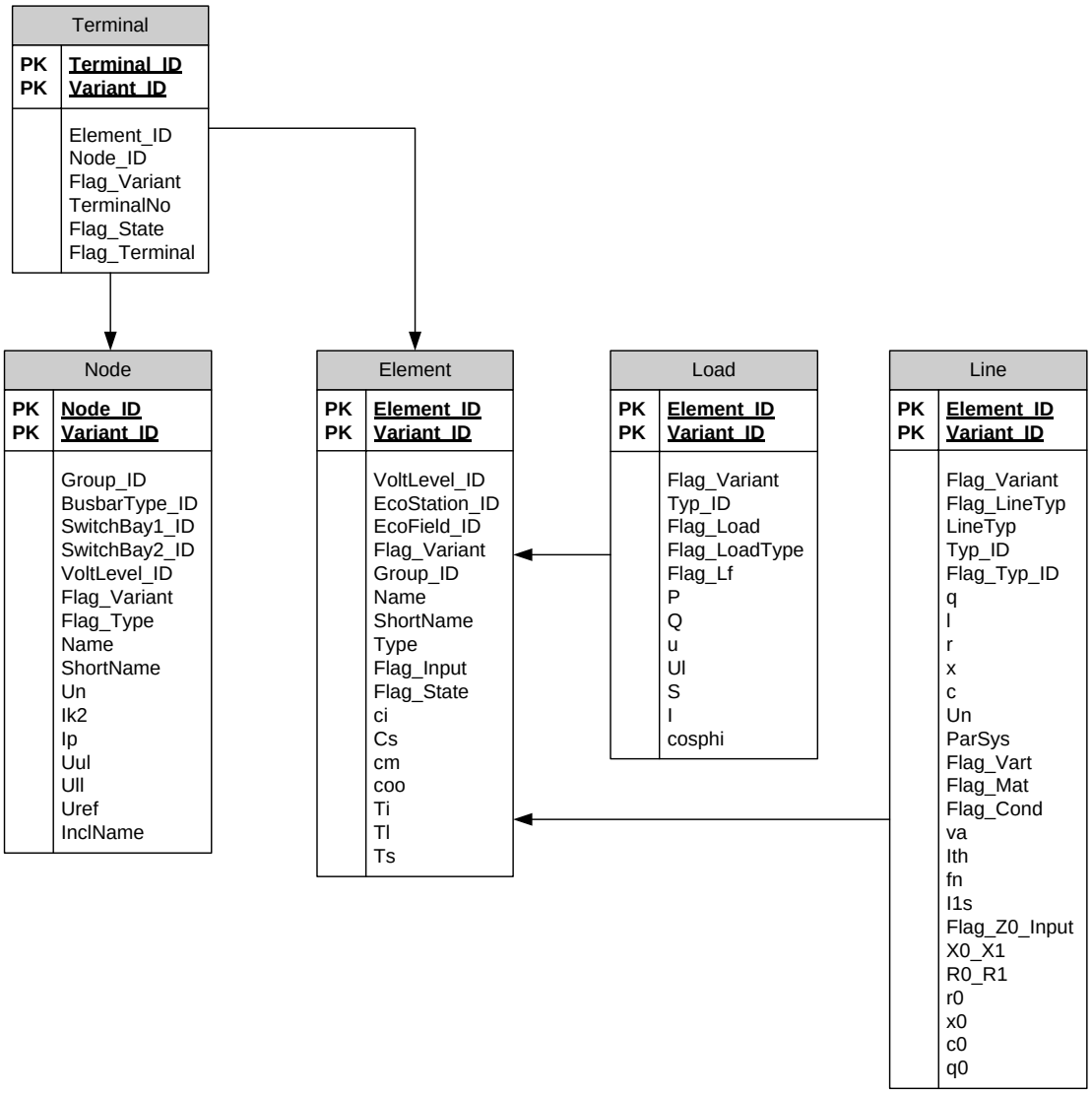
Achtung: PSS SINCAL nutzt spezielle Algorithmen zur Verwaltung der Schlüsselfelder. Es ist daher darauf zu achten, dass die Schlüssel in der Datenbank beginnend mit der kleinsten ID (1) aufsteigend generiert werden. Lücken bei den Schlüsseln sind problemlos möglich, aber es sollte unbedingt vermieden werden, sehr große Zahlenwerte zu speichern, da ansonsten Probleme mit PSS SINCAL GUI Funktionen und dem Variantenmanagement vorprogrammiert sind. Auch die direkte Speicherung von eindeutigen GIS Schlüsseln in den Primärschlüsselfeldern ist nicht zulässig. Für diesen Zweck ist die spezielle Mapping-Tabelle **MasterResource** verfügbar.

Attributsnamen

- Attributsnamen wurden – wenn möglich – gleich wie die entsprechenden Formelzeichen gewählt.
- Bei den Attributsnamen wurde versucht, möglichst sprachneutrale bzw. englische Bezeichnungen zu verwenden.
- Attributsnamen sind eindeutig in Klein- und Großschreibung.
- Hochgestellte Kommas werden durch Zahlen ersetzt: Ik" → ik2
- Unterstriche dienen als Platzhalter für Schrägstriche sowie als Bindungsglied von Ausdrücken.
- Fremd- und Primärschlüssel besitzen die Kennung "_ID": Element_ID, Variant_ID

2.3 Aufbau der Datenbank

Die folgende Übersicht stellt die grundlegende Struktur des Datenmodells anhand einiger ausgewählter Tabellen dar.



Über die Tabelle **Node** (Knoten) wird die topologische Grundstruktur des Netzes beschrieben. Alle Netzelemente werden über ein oder mehrere **Terminals** (Anschluss) an Knoten angeschlossen. Dadurch wird die vollständige Netztopologie gebildet.

Im Mittelpunkt des Datenmodells steht die Tabelle **Element**. Damit werden die eigentlichen Netzelemente beschrieben. Dieser Tabelle ist zur detaillierten Beschreibung des jeweiligen Netzelementes eine weitere Tabelle zugeordnet:

- Leitung: Element + Line
- Verbraucher: Element + Load
- usw.

Zustand der Eingabedaten

In den Datentabellen der Elemente sind die Daten für unterschiedlichste Berechnungsverfahren zusammengefasst, um zu verhindern, dass das Datenmodell zu komplex wird. So enthält beispielsweise die Datentabelle Leitung die Daten zur Lastflussberechnung, Kurzschlussberechnung, Oberschwingungsberechnung, usw.

Alle Attribute der Netzelemente sind Kategorien zugeordnet. Dies ermöglicht trotz der Zusammenfassung der unterschiedlichen Attribute für die verschiedenen Berechnungsmethoden in einer gemeinsamen Tabelle, die Dateneingabe auf die für die jeweilige Berechnungsmethode erforderlichen Daten einzuschränken bzw. den aktuellen Status der Dateneingabe zu erkennen.

Die Tabelle **Element** besitzt das Attribut **Flag_Input**, welches den aktuellen Zustand der Dateneingabe für jede Kategorie speichert, d.h. anhand des Flags kann erkannt werden, welche Daten noch nicht bzw. bereits eingegeben sind. Das Flag wird durch die binäre Oder-Verknüpfung von Konstanten für jede Kategorie erzeugt.

Elektronetze

- Bit 0: Kurzschlussdaten
- Bit 1: Lastflussdaten
- Bit 2: Nullsystemdaten
- Bit 3: Gegensystemdaten
- Bit 4: Oberschwingungsdaten
- Bit 5: Dynamikdaten
- Bit 6: Schutzdaten
- Bit 7: Regelbanddaten
- Bit 8: Zuverlässigkeitsdaten
- Bit 9: Zusatzdaten
- Bit 10: Messwerte
- Bit 11: Motoranlaufdaten
- Bit 12: Traforeglerdaten
- Bit 13: Daten Distanzschutz
- Bit 14: Blockdaten für Generator
- Bit 15: Blockdaten für Transformator
- Bit 16: Direkteinspeisung für Generator
- Bit 17: Ersatzelement
- Bit 39: Dynamische Daten

Strömungsnetze

- Bit 24: Flussdaten
- Bit 25: Wasserdaten
- Bit 26: Gasdaten

- Bit 27: Wärme-/Kältedaten
- Bit 28: Stationäre Daten
- Bit 30: Ersatzelement

Beispiel für die Kodierung von Kurzschluss- und Lastflusseingabedaten in Elektronetzen:

Bit 0 + Bit 1 = 1 + 2 = 3

2.4 Datenbankanalyse anhand eines Beispielnetzes

Im Folgenden soll das Datenmodell anhand eines kleinen Beispielnetzes erläutert werden. Hierzu wird mit der PSS SINCAL Benutzeroberfläche ein entsprechendes Netz erfasst und dabei die Datenbank analysiert.

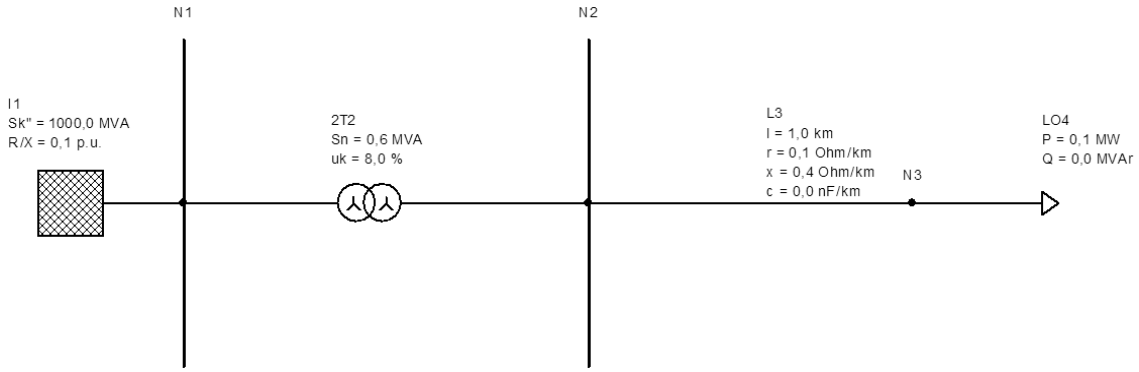


Bild: Beispielnetz mit Eingabedaten

Das hier dargestellte Netz wird schrittweise erfasst. Um die Zusammenhänge im Datenmodell darzustellen, wird nach jedem Bearbeitungsschritt der Inhalt der Datenbank analysiert. Folgende Schritte sind erforderlich, um das Beispielnetz zu erfassen:

- Schritt 1: Anlegen eines neuen Netzes
- Schritt 2: Erstellen der Netzebenen
- Schritt 3: Erfassen der Sammelschienen
- Schritt 4: Erzeugen der Einspeisung
- Schritt 5: Anschließen des Zweiwicklungstransformators
- Schritt 6: Erzeugen der Leitung
- Schritt 7: Anschließen des Verbrauchers

Schritt 1: Anlegen eines neuen Netzes

In der PSS SINCAL Benutzeroberfläche wird ein neues schematisches Elektronetz angelegt. Hierzu wird der Menüpunkt **Datei – Neu** aufgerufen.

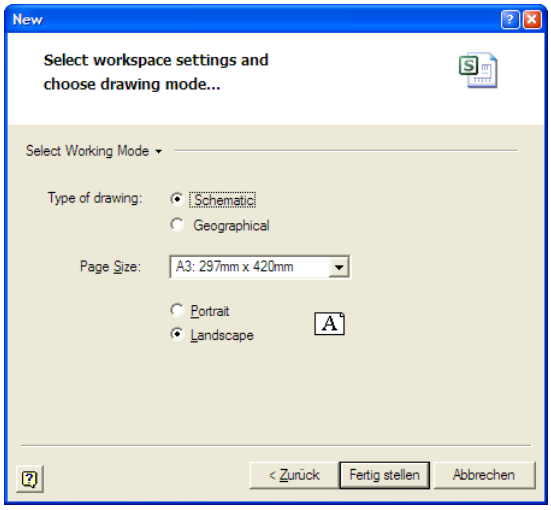


Bild: Dialog zum Anlegen eines neuen Netzes

Für das Beispielnetz wird der Zeichnungstyp **Schematisch** ausgewählt und das Blattformat auf **A3 Querformat** eingestellt.

Schritt 2: Erstellen der Netzebenen

In PSS SINCAL müssen die Netzelemente einer Netzebene zugeordnet werden. Die Netzebene wird zur Vorgabe von allgemein gültigen Daten für Netzelemente (z.B. Nennspannung in Elektronetzen) verwendet.

Standardmäßig wird beim Generieren eines neuen Netzes automatisch eine Netzebene erzeugt. Diese ist mit Standardwerten befüllt und kann angepasst werden. Dies erfolgt über den Menüpunkt **Einfügen – Netzebene**.

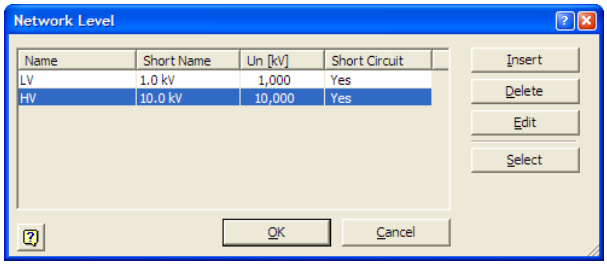


Bild: Dialog zum Anlegen der Netzebenen

Für das Beispielnetz werden folgende Netzebenen erzeugt: LV = 1kV, HV = 10kV

Nach dem Erstellen der Netzebenen sind folgende Werte in der Tabelle **VoltageLevel** enthalten:

VoltageLevel									
VoltLevel_ID	**Name**	**ShortName**	**Un**	**Uop**	**c**	**cmax**	**…**	**Flag_Variant**	**Variant_ID**
1	LV	1.0 kV	1,0	1,0	1,1	1,1		1	1
2	HV	10.0 kV	10,0	10,0	1,1	1,1		1	1

Schritt 3: Erfassen der Sammelschienen

Nun können die Knoten bzw. Sammelschienen erzeugt werden. Hierzu wird über das Menü **Einfügen – Knoten/Sammelschiene – Sammelschiene** das Erfassen aktiviert. Anschließend werden die beiden Sammelschienen N1 und N2 im Grafikeditor erzeugt.

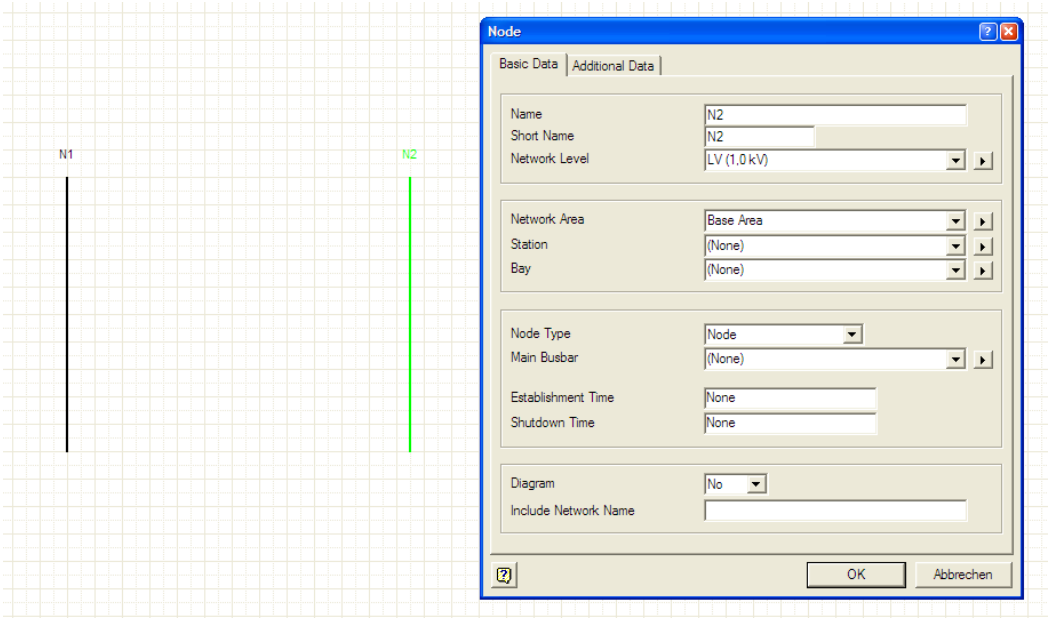


Bild: Netzgrafik mit zwei erfassten Sammelschienen

Nach dem Erfassen der beiden Sammelschienen sind die folgenden Werte in der Tabelle **Node** enthalten.

Node							
Node_ID	**Group_ID**	**VoltLevel_ID**	**Name**	**Un**	**…**	**Flag_Variant**	**Variant_ID**
1	1	2	N1	10,0		1	1
2	1	1	N2	1,0		1	1

Hier kann man schon gut die Zuordnung der Knoten/Sammelschienen (Node) mit Hilfe der Fremdschlüssel zu den Netzebenen (VoltLevel) erkennen. Die Sammelschiene N1 ist der 10 kV Netzebene HV zugeordnet. Diese hat die ID 2. Die Sammelschiene N2 ist der 1 kV Netzebene LV mit der ID 1 zugeordnet.

Schritt 4: Erzeugen der Einspeisung

Der nächste Schritt ist das Erfassen einer Einspeisung. In PSS SINCAL sind hierzu verschiedene Netzelemente verfügbar, mit denen Einspeisungen nachgebildet werden können. Im folgenden Beispiel wird eine Netzeinspeisung erzeugt. Hierzu wird über das Menü **Einfügen – Knotenelemente – Netzeinspeisung** das Erfassen aktiviert.

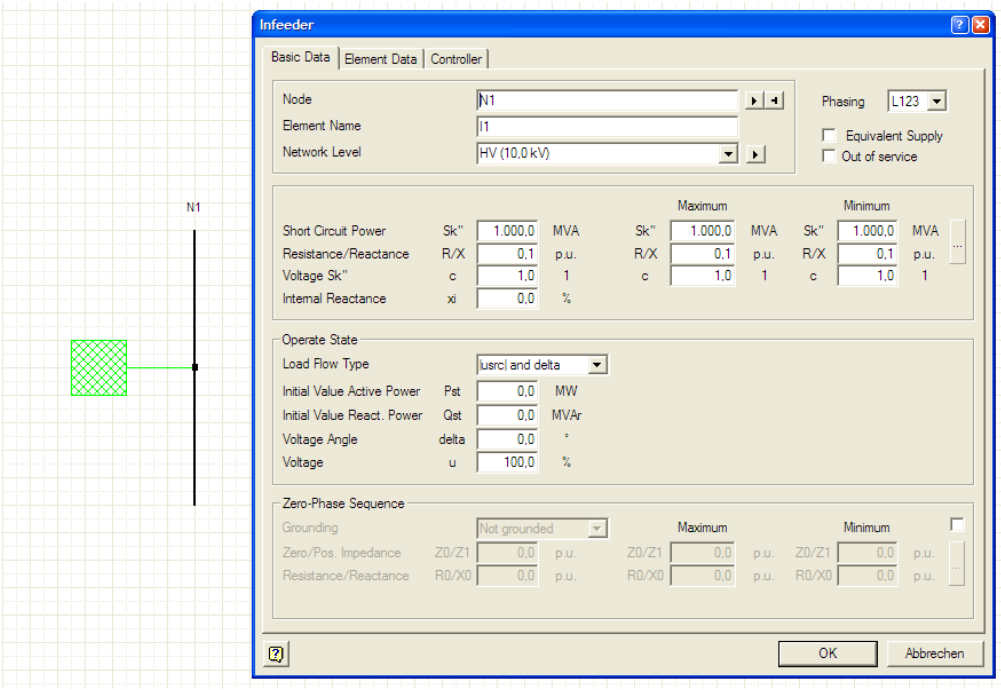


Bild: Netzgrafik mit der neuen Netzeinspeisung

Die Netzeinspeisung ist ein Netzelement mit einem Anschluss. Dieses Netzelement wird daher auch als Knotenelement bezeichnet. In der Datenbank wird die Netzeinspeisung mit folgenden Tabellen beschrieben:

- **Element**: dies ist der Basisdatensatz für das Netzelement
- **Infeeder**: diese Tabelle enthält die spezifischen Attribute der Netzeinspeisung
- **Terminal**: mit dieser Tabelle wird die Verbindung des Netzelementes zu den Knoten/Sammelschienen hergestellt

Element								
Element_ID	**VoltLevel_ID**	**Group_ID**	**Name**	**Type**	**Flag_Input**	**…**	**Flag_Variant**	**Variant_ID**
1	2	1	I1	Infeeder	3		1	1

Die Basisdaten für das Netzelement werden in der Tabelle Element gespeichert. Der Datensatz enthält im Feld **Element_ID** den Primärschlüssel des Netzelementes. Damit ist das Netzelement eindeutig identifizierbar.

Mit dem Feld **VoltLevel_ID** wird eine Verbindung zur Netzebene hergestellt. In diesem Fall ist dies die 10 kV Netzebene HV: VoltLevel_ID = 2

Im Feld **Type** wird der Typ des Netzelementes gespeichert. Dies ist ein ASCII Feld und beinhaltet exakt den Namen der Datentabelle für das Netzelement: Type = "Infeeder"

Mit dem Feld **Flag_Input** wird der Eingabestatus des Netzelementes codiert. Hier ist der binäre Wert 3 eingetragen; dies entspricht dem Bit 0 + Bit 1 → Kurzschlussdaten + Lastflussdaten.

Infeeder								
Element_ID	**Typ_ID**	**Flag_Typ_ID**	**Flag_Typ**	**Sk2**	**cact**	**R**	**X**	**…**
1	0	0	2	1000,0	1,0	0,0	0,0	

Die spezifischen Daten der Netzeinspeisung werden in der Tabelle Infeeder gespeichert. Hier wird mit dem Fremdschlüssel **Element_ID** die Verbindung zur Tabelle Element hergestellt: Element_ID = 1

Terminal							
Terminal_ID	**Element_ID**	**Node_ID**	**TerminalNo**	**Flag_State**	**…**	**Flag_Variant**	**Variant_ID**
1	1	1	1	1		1	1

Die Tabelle Terminal wird verwendet, um die Verbindung des Netzelementes zu den Knoten/Sammelschienen herzustellen. Dies erfolgt mit den Feldern **Element_ID** und **Node_ID**. Im vorliegenden Beispiel ist die Netzeinspeisung I1 an die Sammelschiene N1 angeschlossen: Element_ID = 1, Node_ID = 1

Das Feld **TerminalNo** ist ein Zähler für die Anschlussnummer. Bei der Netzeinspeisung (= Knotenelement) ist dieser immer 1, da dieses Element nur über einen Anschluss verfügt.

Schritt 5: Anschließen des Zweiwicklungstransformators

Nun wird ein Zweiwicklungstransformator zwischen den beiden Sammelschienen erfasst. Hierzu wird über das Menü **Einfügen – Zweigelemente – Zweiwicklungstransformator** das Erfassen aktiviert und anschließend das Element im Grafikeditor erzeugt.

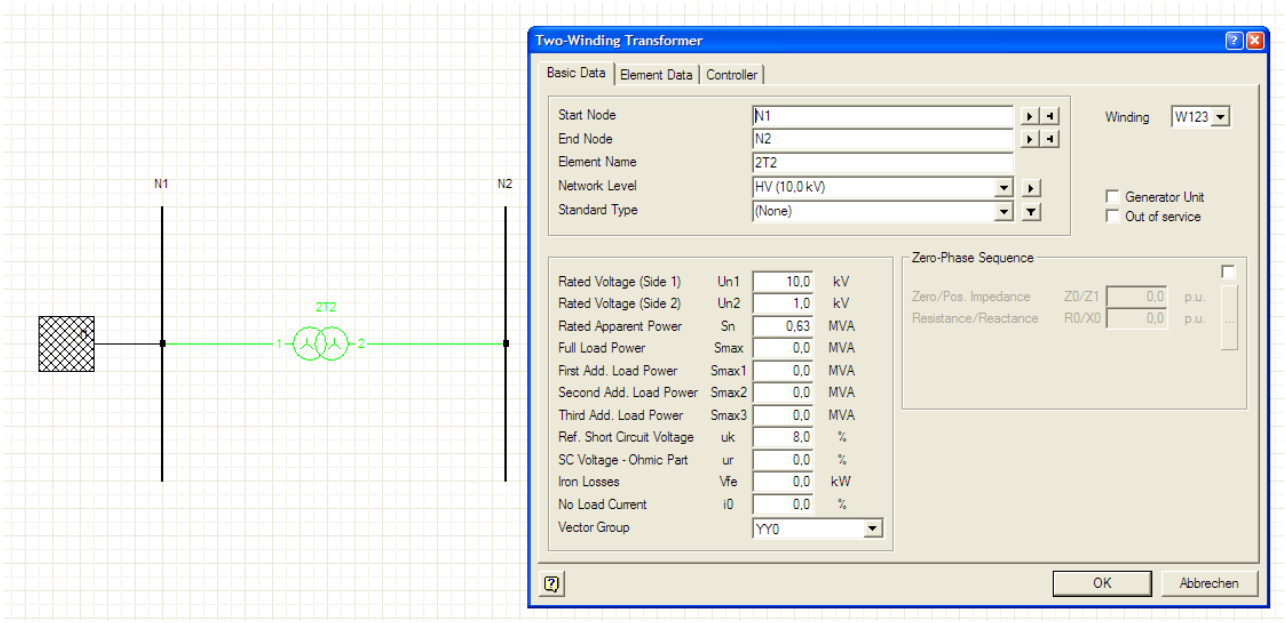


Bild: Netzgrafik mit Zweiwicklungstransformator

Der Zweiwicklungstransformator ist ein Zweigelement. D.h. er hat zwei Anschlüsse, mit denen er an

zwei Knoten/Sammelschienen angeschlossen wird. Im dargestellten Beispiel sind dies die Sammelschienen N1 und N2.

Dieses Zweigelement wird in der Datenbank mit folgenden Tabellen beschrieben:

- **Element**: dies ist der Basisdatensatz des Netzelementes
- **TwoWindingTransformer**: diese Tabelle enthält die spezifischen Attribute des Zweiwicklungstransformators
- **Terminal**: mit dieser Tabelle wird die Verbindung des Netzelementes zu den Knoten/Sammelschienen hergestellt

Element								
Element_ID	**VoltLevel_ID**	**Group_ID**	**Name**	**Type**	**Flag_Input**	**…**	**Flag_Variant**	**Variant_ID**
1	2	1	I1	Infeeder	3		1	1
2	2	1	2T2	TwoWinding Transformer	3		1	1

Die Daten in der Element-Tabelle für den Zweiwicklungstransformator entsprechen jenen, die zuvor bei der Netzeinspeisung beschrieben wurden.

Wie bei der Netzeinspeisung wird auch hier im Feld **Type** der Typ des Netzelementes gespeichert: Type = " TwoWindingTransformer "

TwoWindingTransformer							
Element_ID	**Typ_ID**	**Flag_Typ_ID**	**Un1**	**Un2**	**Sn**	**uk**	**…**
2	0	0	10,0	1,0	0,63	8,0	

Die Eingabedaten des Zweiwicklungstransformators werden in der Tabelle TwoWindingTransformer gespeichert. Hier wird mit dem Fremdschlüssel **Element_ID** die Verbindung zur Tabelle Element hergestellt: Element_ID = 2

Terminal							
Terminal_ID	**Element_ID**	**Node_ID**	**TerminalNo**	**Flag_State**	**…**	**Flag_Variant**	**Variant_ID**
1	1	1	1	1		1	1
2	2	1	1	1		1	1
3	2	2	2	1		1	1

Die Tabelle Terminal wird verwendet, um die Verbindung des Netzelementes zu den Knoten/Sammelschienen herzustellen. Dies erfolgt mit den Feldern **Element_ID** und **Node_ID**.

Der Zweiwicklungstransformator ist ein Zweielement und verfügt daher über zwei Anschlüsse. Mit diesen ist der Transformator 2T2 an die Sammelschienen N1 und N2 angeschlossen:

- Anschluss 1 (Terminal_ID = 2): Element_ID = 2, Node_ID = 1, TerminalNo = 1
- Anschluss 2 (Terminal_ID = 3): Element_ID = 2, Node_ID = 2, TerminalNo = 2

Schritt 6: Erzeugen der Leitung

Im nächsten Schritt wird eine Leitung an die 1,0 kV Sammelschiene angeschlossen. Hierzu wird über das Menü **Einfügen – Zweigelemente – Leitung** das Erfassen aktiviert und anschließend das Element im Grafikeditor erzeugt.

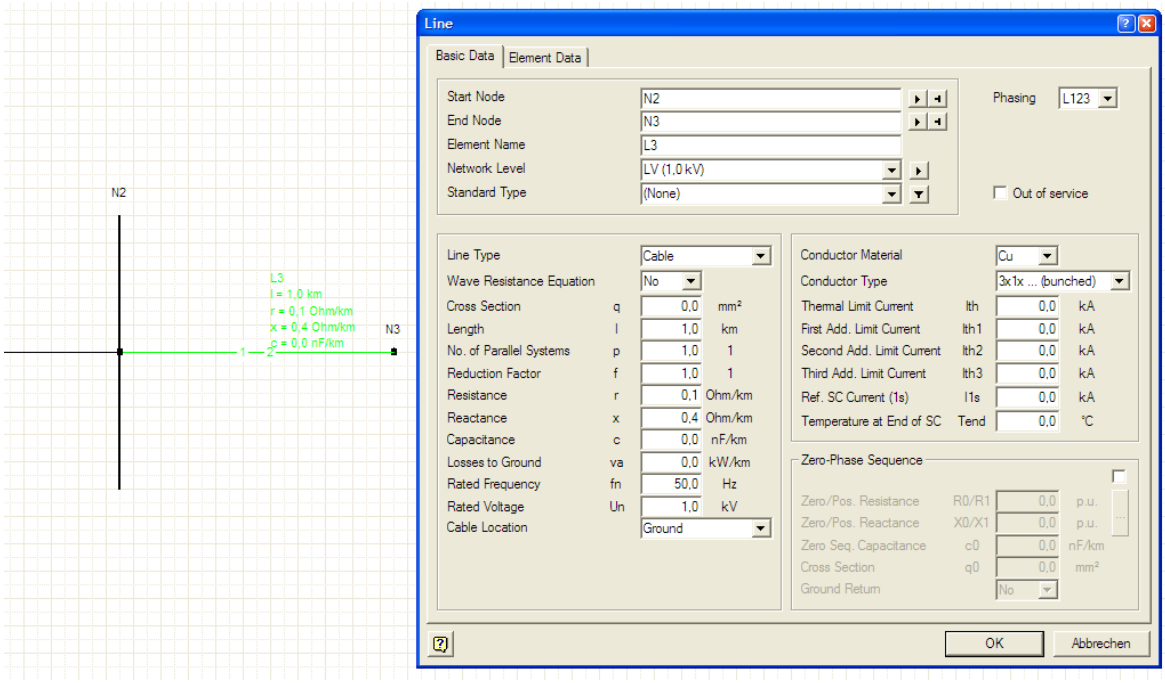


Bild: Netzgrafik mit Leitung

Die Leitung ist genau wie der Zweiwicklungstransformator ein Zweigelement. D.h. diese wird an zwei Knoten/Sammelschienen angeschlossen. Im vorliegenden Beispiel sind dies die Sammelschiene N2 und der Knoten N3.

Dieses Zweigelement wird in der Datenbank mit folgenden Tabellen beschrieben:

- **Element**: dies ist der Basisdatensatz des Netzelementes
- **Line**: diese Tabelle enthält die spezifischen Attribute der Leitung
- **Terminal**: mit dieser Tabelle wird die Verbindung des Netzelementes zu den Knoten/Sammelschienen hergestellt

Die Struktur und Semantik der Tabellen entspricht den vorherigen Erläuterungen. Im Folgenden werden daher die Tabellen nur kurz dargestellt. Zum besseren Verständnis sind die neuen Datensätze farblich gekennzeichnet.

Element								
Element_ID	**VoltLevel_ID**	**Group_ID**	**Name**	**Type**	**Flag_Input**	**…**	**Flag_Variant**	**Variant_ID**
1	2	1	I1	Infeeder	3		1	1
2	2	1	2T2	TwoWinding Transformer	3		1	1
3	1	1	L3	Line	3		1	1

Line							
Element_ID	Typ_ID	Flag_Typ_ID	q	l	ParSys	Flag_Vart	…
3	0	0	0,0	1,0	1	1	

Terminal							
Terminal_ID	Element_ID	Node_ID	TerminalNo	Flag_State	…	Flag_Variant	Variant_ID
1	1	1	1	1		1	1
2	2	1	1	1		1	1
3	2	2	2	1		1	1
4	3	2	1	1		1	1
5	3	3	2	1		1	1

Node							
Node_ID	Group_ID	VoltLevel_ID	Name	Un	…	Flag_Variant	Variant_ID
1	1	2	N1	10,0		1	1
2	1	1	N2	1,0		1	1
3	1	1	N3	1,0		1	1

Schritt 7: Anschließen des Verbrauchers

Mit dem Anschließen eines Verbrauchers wird das Erfassen des Beispielnetzes abgeschlossen. Hierzu wird über das Menü **Einfügen – Knotenelemente – Allgemeine Last** das Erfassen aktiviert und anschließend das Element im Grafikeditor erzeugt.

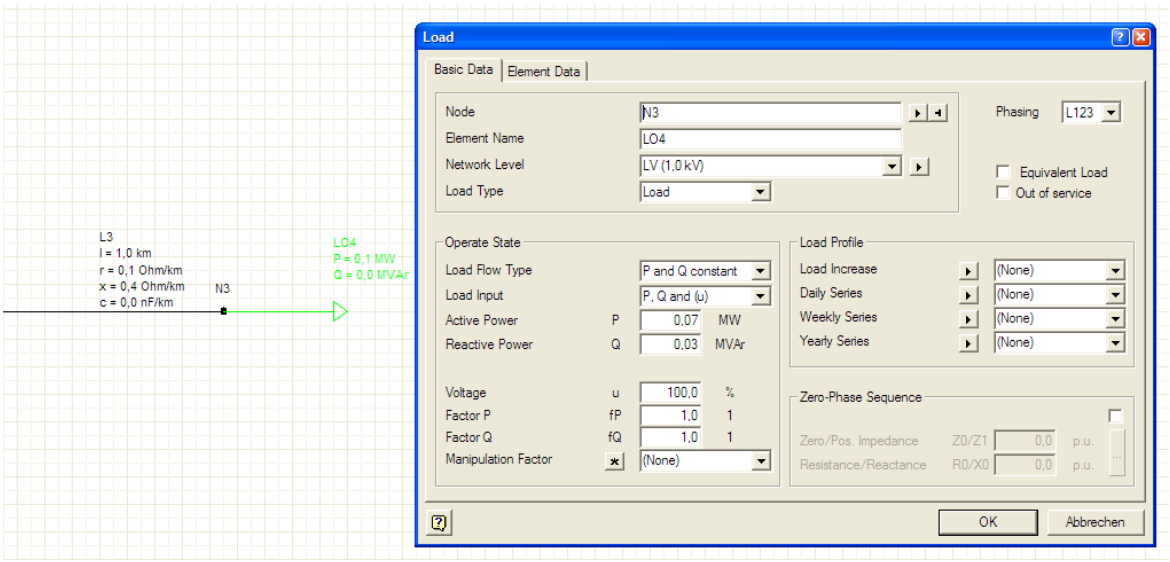


Bild: Netzgrafik mit Verbraucher

Der Verbraucher ist ein Knotenelement. Im vorliegenden Beispiel wird dieser an den Knoten N3 angeschlossen.

Dieses Knotenelement wird in der Datenbank mit folgenden Tabellen beschrieben:

- **Element**: dies ist der Basisdatensatz des Netzelementes
- **Load**: diese Tabelle enthält die spezifischen Attribute des Verbrauchers
- **Terminal**: mit dieser Tabelle wird die Verbindung des Netzelementes zu den Knoten/Sammelschienen hergestellt

Die Struktur und Semantik der Tabellen entspricht den vorherigen Erläuterungen.

2.4.1 Vollständiges Beispielnetz

Das schrittweise aufgebaute Beispielnetz ist nun vollständig.

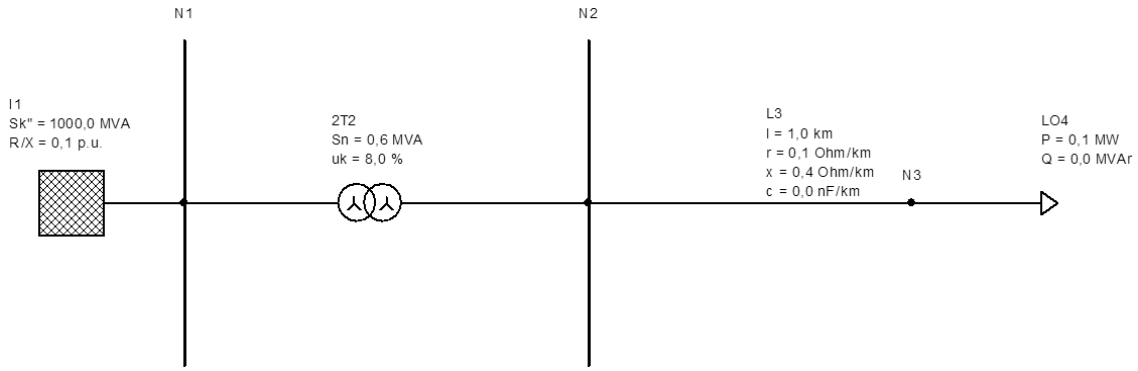


Bild: Vollständiges Beispielnetz

Im Folgenden werden die Inhalte der wichtigsten Tabellen kurz dargestellt, um die Beziehungen der Datensätze nochmals zu verdeutlichen. Zur besseren Übersicht sind die Datensätze der Netzelemente entsprechend ihrer Zugehörigkeit farblich gekennzeichnet:

- Netzeinspeisung: I1
- Zweiwicklungstransformator: 2T2
- Leitung: L3
- Verbraucher: LO4

VoltageLevel									
VoltLevel_ID	**Name**	**ShortName**	**Un**	**Uop**	**c**	**cmax**	**…**	**Flag_Variant**	**Variant_ID**
1	LV	1.0 kV	1,0	1,0	1,1	1,1		1	1
2	HV	10.0 kV	10,0	10,0	1,1	1,1		1	1

Node							
Node_ID	**Group_ID**	**VoltLevel_ID**	**Name**	**Un**	**…**	**Flag_Variant**	**Variant_ID**
1	1	2	N1	10,0		1	1
2	1	1	N2	1,0		1	1
3	1	1	N3	1,0		1	1

Element								
Element_ID	VoltLevel_ID	Group_ID	Name	Type	Flag_Input	…	Flag_Variant	Variant_ID
1	2	1	I1	Infeeder	3		1	1
2	2	1	2T2	TwoWinding Transformer	3		1	1
3	1	1	L3	Line	3		1	1
4	1	1	LO4	Load	3		1	1

Terminal							
Terminal_ID	Element_ID	Node_ID	TerminalNo	Flag_State	…	Flag_Variant	Variant_ID
1	1	1	1	1		1	1
2	2	1	1	1		1	1
3	2	2	2	1		1	1
4	3	2	1	1		1	1
5	3	3	2	1		1	1
6	4	3	1	1		1	1

Infeeder								
Element_ID	Typ_ID	Flag_Typ_ID	Flag_Typ	Sk2	cact	R	X	…
1	0	0	2	1000,0	1,0	0,0	0,0	

TwoWindingTransformer							
Element_ID	Typ_ID	Flag_Typ_ID	Un1	Un2	Sn	uk	…
2	0	0	10,0	1,0	0,63	8,0	

Line							
Element_ID	Typ_ID	Flag_Typ_ID	q	l	ParSys	Flag_Vart	…
3	0	0	0,0	1,0	1	1	

Load							
Element_ID	Flag_Load	Flag_LoadType	P	Q	u	Ul	…
4	0	0	0,07	0,03	100.0	0.0	

2.5 Tabellen der Netzgrafik

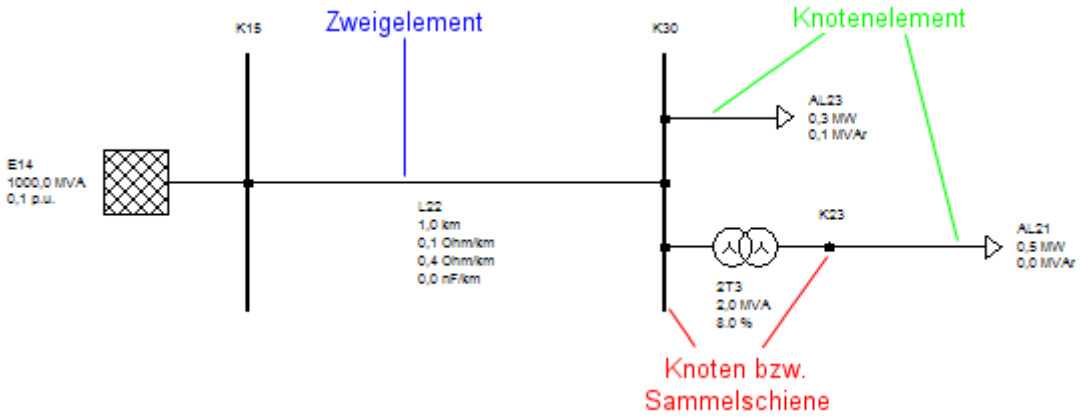
Mit den Grafiktabellen wird die grafische Darstellung des Netzes beschrieben. Diese Informationen werden aber nur zur Visualisierung und Bearbeitung in der Benutzeroberfläche benötigt. Zum Berechnen des Netzes werden die Grafiktabellen nicht benötigt. D.h. falls nur die Berechnungsmethoden von PSS SINCAL verwendet werden (z.B. für eine Engines-Lösung), ist es nicht erforderlich, die Grafiktabellen zu befüllen.

Die folgende Auflistung zeigt die wichtigsten Grafiktabellen von PSS SINCAL:

- **Basistabellen der Grafikelemente**
  - GraphicNode: Grafik für Knoten/Sammelschienen
  - GraphicElement: Grafik für Netzelemente
  - GraphicTerminal: Grafik für die Anschlüsse der Netzelemente
- **Zusatztabellen**
  - GraphicBucklePoint: Knickstellen
  - GraphicText: Texte
- **Grundstrukturen**
  - GraphicAreaTile: Gebiet und Kachel
  - GraphicLayer: Ebenen
  - GraphicObjectType: Objekttyp

2.5.1 Basistabellen der Grafikelemente

Ein Netz wird durch seine Knoten/Sammelschiene und die daran angeschlossenen Netzelemente strukturell beschrieben. Zweigelemente verbinden je zwei Knoten bzw. Sammelschienen miteinander. Die Knotenelemente werden an Knoten/Sammelschienen angeschlossen.



Die einfachsten Grafikelemente stellen die **Knoten/Sammelschienen** dar. Sie bestehen nur aus der Knotengrafik und den Texten. An ihnen werden die Netzelemente angeschlossen.

Die **Netzelemente** sind komplexere Grafikelemente. Sie setzen sich aus den Bausteinen Symbol, Anschluss und Text zusammen.

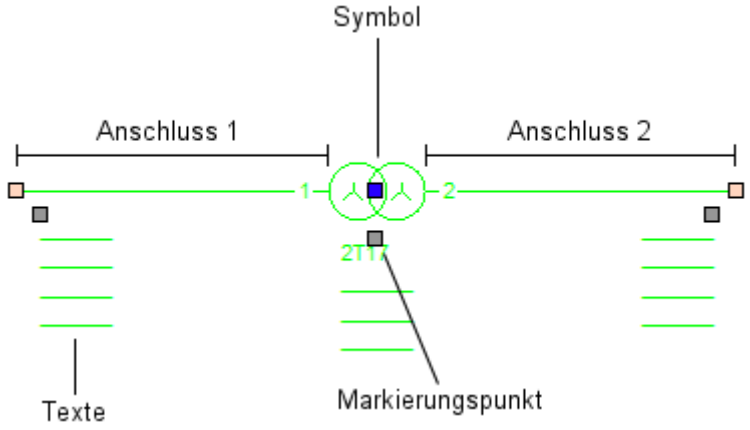


Bild: Markierter Zweiwicklungstransformator

Im dargestellten Beispiel ist ein Zweigelement – oder genauer ein Zweiwicklungstransformator – markiert. In der Markierungsdarstellung können die Bausteine der Netzelemente relativ einfach erkannt werden: Das dargestellte Zweigelement besteht aus einem Symbol, den beiden Anschlüssen (Verbindung vom Symbol zum Knoten) und den Texten.

Knotenelemente bestehen aus einem Anschluss, dem Symbol und den Texten.

Zweigelemente bestehen aus zwei Anschlüssen, dem Symbol und den Texten.

Eine Sonderform der Zweigelemente ist der Dreiwicklungstransformator. Dieser wird im Gegensatz zu allen anderen Zweigelementen an drei Knoten/Sammelschienen angeschlossen und besitzt daher drei Anschlüsse.

GraphicNode (Grafik für Knoten/Sammelschienen)

Mit dieser Tabelle werden die Grafikattribute für Knoten/Sammelschienen beschrieben.

Attributname	Datentyp	Einheit	Beschreibung
GraphicNode_ID	Long Integer		Primärschlüssel für Grafikknoten
GraphicLayer_ID	Long Integer		Fremdschlüssel für Layer
GraphicType_ID	Long Integer		Fremdschlüssel für Objekttyp
GraphicText_ID1	Long Integer		Fremdschlüssel für Textattribut 1
GraphicText_ID2	Long Integer		Fremdschlüssel für Textattribut 2
Node_ID	Long Integer		Fremdschlüssel für Knoten (Sachdaten)
FrgndColor	Long Integer	RGB	Linienfarbe
BkgndColor	Long Integer	RGB	Füllfarbe
PenStyle	Integer		Strichart 0: Strich 1: Strichliert 2: Punktiert 3: Strich-Punkt-Strich 4: Strich-Punkt-Punkt-Strich
PenWidth	Integer	0.25mm	Linienstärke
NodeSize	Integer	0.25mm	Symbolgröße
NodeStartX	Double	m	Knotenanfangspunkt X-Koordinate
NodeStartY	Double	m	Knotenanfangspunkt Y-Koordinate
NodeEndX	Double	m	Knotenendpunkt X-Koordinate
NodeEndY	Double	m	Knotenendpunkt Y-Koordinate

SymType	Integer		Art des Knotensymbols 0: Kein Symbol 1: Kreis 2: Rechteck 3: Sammelschiene
Flag	Long Integer		Flag
Variant_ID	Long Integer		Fremdschlüssel für Variante
Flag_Variant	Integer		Element der aktuellen Variante 0: Nein 1: Ja
GraphicArea_ID	Long Integer		Fremdschlüssel für Gebiet/Kachel

Über den Fremdschlüssel **GraphicText_ID1** kann ein grafisches Textobjekt zugewiesen werden. D.h. es wird dann ein individuelles Textobjekt mit eigener Position und Grafikattributen im Grafikeditor angezeigt. Es ist auch zulässig, das Feld mit "NULL" zu initialisieren. Dann wird der Text mit Defaultattributen im Grafikeditor angezeigt, kann aber nicht manuell bearbeitet werden.

Der Fremdschlüssel **GraphicText_ID2** wird derzeit nicht verwendet und sollte daher immer mit "NULL" initialisiert werden.

GraphicElement (Grafik für die Netzelementsymbole)

Mit dieser Tabelle werden die Grafikattribute für die Netzelementsymbole beschrieben.

Attributname	Datentyp	Einheit	Beschreibung
GraphicElement_ID	Long Integer		Primärschlüssel für Grafikelement
GraphicLayer_ID	Long Integer		Fremdschlüssel für Layer
GraphicType_ID	Long Integer		Fremdschlüssel für Objekttyp
GraphicText_ID1	Long Integer		Fremdschlüssel für Textattribut 1
GraphicText_ID2	Long Integer		Fremdschlüssel für Textattribut 2
Element_ID	Long Integer		Fremdschlüssel für Element
SymbolDef	Long Integer		Symboleigenschaften übernehmen
FrgndColor	Long Integer	RGB	Linienfarbe
BkgndColor	Long Integer	RGB	Füllfarbe
PenStyle	Integer		Strichart 0: Strich 1: Strichliert 2: Punktiert 3: Strich-Punkt-Strich 4: Strich-Punkt-Punkt-Strich
PenWidth	Integer	0.25mm	Linienstärke
SymbolSize	Integer	0.25mm	Symbolgröße
SymCenterX	Double	m	Symbolmittelpunkt X-Koordinate
SymCenterY	Double	m	Symbolmittelpunkt Y-Koordinate

SymbolType	Long Integer		**Art des Symbols Elektronetze:** 9: Synchronmaschine 10: Kraftwerksblock 11: Netzeinspeisung 12: Asynchronmaschine 13: Allgemeine Last 15: Querimpedanz 16: Querdrossel 17: Querkondensator 18: Statischer Kompensator 19: Leitung 20: Zweiwicklungstransformator 21: Dreiwicklungstransformator 22: Längsdrossel 23: Längskondensator 24: Querrundsteuersender 25: Längsrundsteuersender 26: Quer RLC-Kreis 27: Läng RLC-Kreis 29: Resonanznetz 193: DC-Einspeisung 194: DC-Längselement 123: Variables Querelement 124: Variables Längselement **Art des Symbols Strömungsnetze:** 9: Hochbehälter 10: Pumpeinspeisung 11: Einspeisung Gas 12: Einspeisung Wärme/Kälte 13: Verbraucher 14: Druckbuffer 15: Leck 16: Temperaturregler 17: Leitung 18: Druckverstärkerpumpe 19: Konst. Druckabfall/Konst. Fluss 20: Druckregler 21: Kompressor 22: Wärmetauscher 23: Schieber/Rückschlagventil
SymbolNo	Integer		Symbolnummer
Flag	Long Integer		Flag
Variant_ID	Long Integer		Fremdschlüssel für Variante
Flag_Variant	Integer		Element der aktuellen Variante 0: Nein 1: Ja
GraphicArea_ID	Long Integer		Fremdschlüssel für Gebiet/Kachel

Das Feld **SymbolType** ist hier besonders wichtig. Es muss korrekt initialisiert werden, ansonsten erfolgt keine/oder eine fehlerhafte Zuordnung der Grafikdaten zu den Netzelementdaten in der PSS SINCAL Benutzeroberfläche.

Das Feld **SymbolDef** wird zur erweiterten Steuerung des Netzelementsymbols verwendet. Für Kopplungslösungen sollte dies mit "-1" initialisiert werden.

Über den Fremdschlüssel **GraphicText_ID1** kann ein grafisches Textobjekt zugewiesen werden. D.h. es wird dann ein individuelles Textobjekt mit eigener Position und Grafikattributen im Grafikeditor angezeigt. Es ist auch zulässig, das Feld mit "NULL" zu initialisieren. Dann wird der Text mit Defaultattributen im Grafikeditor angezeigt, kann aber nicht manuell bearbeitet werden.

Der Fremdschlüssel **GraphicText_ID2** wird derzeit nicht verwendet und sollte daher immer mit "NULL" initialisiert werden.

GraphicTerminal (Grafik für die Anschlüsse von Netzelementen)

Mit dieser Tabelle werden die Grafikattribute für die Anschlüsse der Netzelemente beschrieben.

Attributname	Datentyp	Einheit	Beschreibung
GraphicTerminal_ID	Long Integer		Primärschlüssel für Grafikanschluss
GraphicElement_ID	Long Integer		Fremdschlüssel für Grafikelement
GraphicText_ID	Long Integer		Fremdschlüssel für Grafiktext
Terminal_ID	Long Integer		Fremdschlüssel für Terminal
PosX	Double	m	X-Koordinate
PosY	Double	m	Y-Koordinate
FrgndColor	Long Integer	RGB	Linienfarbe
PenStyle	Integer		Strichart 0: Strich 1: Strichliert 2: Punktiert 3: Strich-Punkt-Strich 4: Strich-Punkt-Punkt-Strich
PenWidth	Integer	0.25mm	Linienstärke
SwtType	Integer		Schaltertyp 0: Kein Typ 1: Typ 1 2: Typ 2 3: Typ 3 4: Typ 4 5: Typ 5 6: Typ 6
SwtAlign	Integer		Schalterausrichtung 0: Automatisch 1: Position 1 2: Position 2 3: Position 3 4: Position 4
SwtNodePos	Double	0.25mm	Schalterabstand zum Knoten
SwtFactor	Integer	0.25mm	Schaltergrößenfaktor
SwtFrgndColor	Long Integer	RGB	Schalterlinienfarbe
SwtPenStyle	Integer		Schalterstrichart 0: Strich 1: Strichliert 2: Punktiert 3: Strich-Punkt-Strich 4: Strich-Punkt-Punkt-Strich
SwtPenWidth	Integer	0.25mm	Schalterlinienstärke
SymbolType	Integer		Symboltyp
SymbolAlign	Integer		Symbolausrichtung
SymbolNodePos	Double	0.25mm	Symbolabstand zum Knoten
SymbolFactor	Integer		Symbolgrößenfaktor
SymbolFrgndColor	Long Integer	RGB	Symbollinienfarbe
SymbolPenStyle	Integer		Symbolstrichart
SymbolPenWidth	Integer	0.25mm	Symbollinienstärke
TextAlign	Integer		Ausrichtung Text
Flag	Long Integer		Flag
Variant_ID	Long Integer		Fremdschlüssel für Variante
Flag_Variant	Integer		Element der aktuellen Variante 0: Nein 1: Ja
GraphicArea_ID	Long Integer		Fremdschlüssel für Gebiet/Kachel

Mit den Feldern **Pos_X** und **Pos_Y** wird der Verbindungspunkt des Anschlusses zum Knoten bzw. zur Sammelschiene definiert. Bei einem Knoten ist dies immer der Mittelpunkt. Bei einer Sammelschiene ist dies ein beliebiger Punkt auf der Sammelschiene.

Über den Fremdschlüssel **GraphicText_ID** kann ein grafisches Textobjekt zugewiesen werden. D.h. es wird dann ein individuelles Textobjekt mit eigener Position und Grafikattributen im Grafikeditor angezeigt. Es ist auch zulässig, das Feld mit "NULL" zu initialisieren. Dann wird der Text mit Defaultattributen im Grafikeditor angezeigt, kann aber nicht manuell bearbeitet werden.

2.5.2 Zusatztabellen

GraphicText (Textobjekte)

Mit dieser Tabelle können individuelle Textobjekte für Knoten/Sammelschienen und Netzelemente definiert werden.

Attributname	Datentyp	Einheit	Beschreibung
GraphicText_ID	Long Integer		Primärschlüssel für Text
GraphicLayer_ID	Long Integer		Fremdschlüssel für Layer
Font	Text (20)		Schrift
FontStyle	Integer		Textstil 16: Standard 17: Fett 18: Kursiv
FontSize	Integer		Texthöhe
TextAlign	Integer		Textausrichtung 0: Links 1: Mitte 2: Rechts
TextOrient	Integer	0,1 °	Textrichtung
TextColor	Long Integer	RGB	Textfarbe
Visible	Integer		Sichtbar 0: Nein 1: Ja
AdjustAngle	Integer		Textwinkel ausrichten 0: Keine 1: Horizontal oder Vertikal 2: In Richtung des Elements
Angle	Double	°	Textwinkel
Pos1	Double	m	Abstand X-Richtung
Pos2	Double	m	Abstand Y-Richtung
Flag	Long Integer		Flag
RowTextNo	Integer	1	Anzahl Zeilen
AngleTermNo	Integer	1	Terminalsegment für Ausrichtung
Variant_ID	Long Integer		Fremdschlüssel für Variante
Flag_Variant	Integer		Element der aktuellen Variante 0: Nein 1: Ja

Achtung: ein Textobjekt darf nur einmal verwendet werden. Die Verwendung desselben Textobjektes ist bei verschiedenen Grafikelementen nicht zulässig!

GraphicBucklePoint (Knickstellen für Anschlüsse)

Mit dieser Tabelle können Knickstellen für die Anschlüsse der Netzelemente definiert werden.

Attributname	Datentyp	Einheit	Beschreibung
GraphicPoint_ID	Long Integer		Primärschlüssel für Knickpunkt
GraphicTerminal_ID	Long Integer		Fremdschlüssel für Grafikanschluss
NoPoint	Integer	1	Knickpunktnummer
PosX	Double	m	Knickpunkt X-Koordinate
PosY	Double	m	Knickpunkt Y-Koordinate
Variant_ID	Long Integer		Fremdschlüssel für Variante
Flag_Variant	Integer		Element der aktuellen Variante 0: Nein 1: Ja

Mit den Feldern **PosX** und **PosY** wird die grafische Position der Knickstelle definiert.

Die Reihenfolge der Knickstellen wird über das Feld **NoPoint** bestimmt. Hierbei werden die Knickstellen jeweils ausgehend vom Symbolpunkt des Netzelementes hin zu dem Anschlusspunkt am Knoten nummeriert.

2.5.3 Grundstrukturen

GraphicAreaTile (Zeichenblatteinstellungen)

Mit dieser Tabelle wird das Zeichenblatt beschrieben. Im Wesentlichen werden hier die Blattgröße sowie der Maßstab definiert.

Attributname	Datentyp	Einheit	Beschreibung
GraphicArea_ID	Long Integer		Primärschlüssel für Gebiet/Kachel
Name	Text (50)		Name der Ansicht
AreaWidth	Double	cm	Zeichenblattbreite
AreaHeight	Double	cm	Zeichenblatthöhe
VectorX	Double	m	Nullpunktverschiebung X-Richtung
VectorY	Double	m	Nullpunktverschiebung Y-Richtung
GridWidth	Integer	0.25mm	Hilfsgitterbreite
GridHeight	Integer	0.25mm	Hilfsgitterhöhe
Flag	Long Integer		Netzbearbeitungsmodus 1: Lageorientiert 2: Schematisch
Scale1	Integer		Maßstab vordefiniert 0: 1:100000000 1: 1:10000000 2: 1:1000000 3: 1:100000 4: 1:10000 5: 1:1000 6: 1:100 7: 1:10 8: 1:1

Scale2	Integer		Darstellungseinheit 0: mm 1: cm 2: m 3: km 4: Inch 5: Fuß 6: Yards 7: Meilen
ScalePaper	Double		Scale Paper
ScaleReal	Double		Scale Real
TileIndex	Text (8)		Kachelindex
Variant_ID	Long Integer		Fremdschlüssel für Variante

In einem PSS SINCAL Netz können auch mehrere verschiedene Zeichenblätter angelegt werden. Hierzu werden einfach mehrere Datensätze in dieser Tabelle erzeugt. In allen Grafiktabellen ist die **GraphicArea_ID** als Fremdschlüssel verfügbar. Somit kann definiert werden, welchem Zeichenblatt die jeweilige Grafik zugeordnet werden soll.

GraphicLayer (Grafikebene)

Mit dieser Tabelle wird eine Grafikebene definiert. Alle Grafikelemente werden einer Grafikebene zugeordnet. Mit Hilfe der Grafikebene kann die Sichtbarkeit der Netzelemente gesteuert werden.

Attributname	Datentyp	Einheit	Beschreibung
GraphicLayer_ID	Long Integer		Primärschlüssel für Ebene
Name	Text (50)		Ebenenname
Visible	Integer		Sichtbar 0: Nicht sichtbar 1: Sichtbar am Bildschirm 2: Sichtbar beim Drucken 3: Sichtbar am Bildschirm und beim Drucken
Locked	Integer		Gesperrt für Bearbeitung 0: Nein 1: Ja
Type	Integer		Plotterkopf 0: Nein 1: Ja
Flag	Long Integer		Reihenfolge im Dialog
Variant_ID	Long Integer		Fremdschlüssel für Variante
Flag_Variant	Integer		Element der aktuellen Variante 0: Nein 1: Ja
VisibleZF	Integer		Sichtbar ab Zoomfaktor
GraphicArea_ID	Long Integer		Fremdschlüssel für Gebiet/Kachel

GraphicObjectType (Objekttyp)

Allen grafischen Netzelementen wird ein Objekttyp zugeordnet. Mit Hilfe dieses Objekttyps kann der Beschriftungsumfang in der Netzgrafik gesteuert werden.

Attributname	Datentyp	Einheit	Beschreibung
GraphicType_ID	Long Integer		Primärschlüssel für Objekttyp
ParentType_ID	Long Integer		Fremdschlüssel für übergeordneten Objekttyp

Name	Text (50)		Objekttypname
Visible	Integer		Sichtbar 0: Nein 1: Ja
Locked	Integer		Gesperrt für Bearbeitung (nicht in Verwendung)
Type	Integer		Typ (nicht in Verwendung)
Flag	Long Integer		Flag (nicht in Verwendung)
Variant_ID	Long Integer		Fremdschlüssel für Variante
Flag_Variant	Integer		Element der aktuellen Variante 0: Nein 1: Ja
VisibleZF	Integer		Sichtbar ab Zoomfaktor

2.6 Ergebnisse in der Datenbank

In PSS SINCAL werden die Berechnungsergebnisse genauso wie die Eingabedaten in der Datenbank gespeichert. Das Speichern in die Datenbank erfolgt automatisch, sobald eine Berechnung erfolgreich durchgeführt wurde. Die in der Datenbank gespeicherten Ergebnisse können so auch jederzeit mit eigenen Anwendungen aus der Datenbank ausgelesen werden.

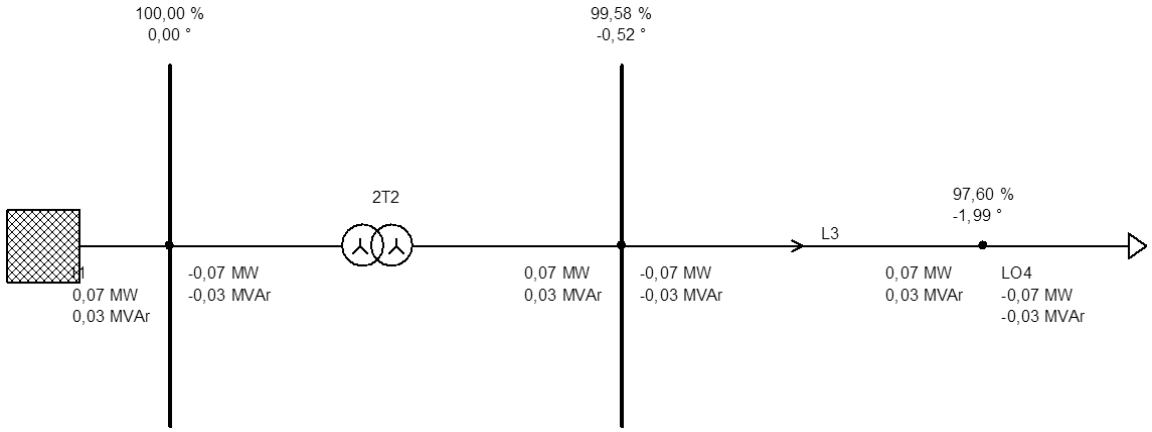


Bild: Beispielnetz mit Lastflussergebnissen

Die wichtigsten Ergebnistabellen für Elektronetze sind:

- **Lastfluss**
  - LFNodeResult: Knotenergebnisse Lastfluss
  - LFBranchResult: Zweigergebnisse Lastfluss
  - ULFNodeResult: Knotenergebnisse unsymmetrischer Lastfluss
  - ULFBranchResult: Zweigergebnisse unsymmetrischer Lastfluss
  - LFGroupResult: Lastflussergebnisse – Netzbereich
  - LFParNetLossesResult: Lastflussergebnisse – Teilnetzverluste
  - LFAccurResult: Lastflussergebnisse – Genauigkeit
- **Kurzschluss**
  - SC3NodeResult: Knotenergebnisse 3-poliger Kurzschluss
  - SC3BranchResult: Zweigergebnisse 3-poliger Kurzschluss

- SC1NodeResult: Knotenergebnisse Kurzschluss für unsymmetrische Fehler
- SC1BranchResult: Zweigergebnisse 1-poliger Erdschluss für unsymmetrische Fehler

- **Optimierungen**
  - SeparationResult: Trennstellensuche Ergebnisse
  - InstallCompResult: Kompensationsleistungsergebnisse
- **Oberschwingungen und Rundsteuerung**
  - HarBranchResult: Zweigergebnisse Oberschwingungen
  - HarNodeResult: Knotenergebnisse Oberschwingungen
  - RCBranchResult: Zweigergebnisse Rundsteuerung
  - RCNodeResult: Knotenergebnisse Rundsteuerung
  - RCTransmitterResult: Senderergebnisse Rundsteuerung
- **Zuverlässigkeit**
  - RelResult: Knotenergebnisse Zuverlässigkeit
  - RelNetResult: Netzergebnisse Zuverlässigkeit
  - RelGroupResult: Netzbereichsergebnisse Zuverlässigkeit

Der grundlegende Aufbau der Ergebnistabellen wird anhand der Lastflussergebnisse dargestellt. Eine detaillierte Beschreibung aller verfügbaren Ergebnistabellen mit deren Attributen ist im Handbuch **Datenbankbeschreibung** verfügbar.

LFNodeResult (Knotenergebnisse Lastfluss)

Diese Tabelle enthält die Knotenergebnisse der Lastflussberechnung. Die Zuordnung des Ergebnisses zum entsprechenden Knoten erfolgt über den Fremdschlüssel **Node_ID**.

Attributname	Datentyp	Einheit	Beschreibung
Result_ID	Long Integer		Primärschlüssel für Ergebnis
Node_ID	Long Integer		Fremdschlüssel für Knoten
Variant_ID	Long Integer		Fremdschlüssel für Variante
U	Double	kV	Leiterspannspannung
U_Un	Double	%	Leiterspannung zu Knotennennspannung
phi	Double	°	Winkel Leiterspannung zu Slackspannung
P	Double	MW	Wirkleistung
Q	Double	Mvar	Blindleistung
S	Double	MVA	Scheinleistung
t	Integer	h	Schlüssel für Zeit
tdiag	Double	s	Zeit für direkte Diagrammanbindung (22:00 = 22 * 3600)
Flag_Result	Integer		Ergebnisart 0: Lastfluss 1: Lastprofil 2: Lastentwicklung
ResDate			Datum
ResTime	Double	h	Zeit
Flag_State	Integer		Status 1: Ok 2: Grenzwertverletzung
Loading	Double	1	Faktor je nach erweiterter Berechnung
Uph	Double	kV	Phasenspannung am Knoten
Uph_Unph	Double	%	Phasenspannung zu Knotennennspannung

phi_ph	Double	°	Winkel Phasenspannung zu Slackspannung
U_Uref	Double	%	Leiterspannung zu Referenzspannung
Uph_Urefph	Double	%	Phasenspannung zu Referenzspannung

LFBranchResult (Zweigergebnisse Lastfluss)

Diese Tabelle enthält die Zweigergebnisse der Lastflussberechnung. Die Ergebnisse werden pro Anschluss (Terminal) bereitgestellt. Die Zuordnung des Ergebnisses zum entsprechenden Netzelement erfolgt über den Fremdschlüssel **Terminal1_ID**.

Attributname	Datentyp	Einheit	Beschreibung
Result_ID	Long Integer		Primärschlüssel für Ergebnis
Terminal1_ID	Long Integer		Fremdschlüssel für Anschluss
Terminal2_ID	Long Integer		Fremdschlüssel für Nachbarfeld
Variant_ID	Long Integer		Fremdschlüssel für Variante
P	Double	MW	Wirkleistung
Q	Double	Mvar	Blindleistung
S	Double	MVA	Scheinleistung
cos_phi	Double	pu	Leistungsfaktor
I	Double	kA	Strom
Inb	Double	%	Basisauslastung
Pl	Double	MW	Wirkleistungsverluste
Ql	Double	Mvar	Blindleistungsverluste
Sl	Double	MVA	Scheinleistungsverluste
dU	Double	kV	Längsspannungsabfall
deltaphi	Double	°	Phasendrehung
Sn	Double	MVA	Bezugsscheinleistung
S_Sn	Double	%	Scheinleistung/Bezugsscheinleistung
Inp	Double	kA	Bezugsstrom Seite 1 (primär)
I_Inp	Double	%	Strom/Bezugsstrom Seite 1 (primär)
Ins	Double	kA	Bezugsstrom Seite 2 (sekundär)
I_Ins	Double	%	Strom/Bezugsstrom Seite 2 (sekundär)
t	Integer	h	Schlüssel für Zeit
tdiag	Double	s	Zeit für direkte Diagrammanbindung (22:00 = 22 * 3600)
Flag_Result	Integer		Ergebnisart 0: Lastfluss 1: Lastprofil 2: Lastentwicklung
ResDate			Datum
ResTime	Double	h	Zeit
Flag_State	Integer		Status 1: Ok 2: Grenzwertverletzung
Inb1	Double	%	Erste Zusatzauslastung
Inb2	Double	%	Zweite Zusatzauslastung
Inb3	Double	%	Dritte Zusatzauslastung

3 Befüllen der PSS SINCAL Datenbank

In diesem Kapitel wird die Vorgangsweise für das manuelle Befüllen der PSS SINCAL Datenbank mit eigenen Applikationen erläutert.

3.1 Beispielprogramm zum Befüllen der Datenbank

Im Zuge der PSS SINCAL Installation wird ein Beispielprogramm bereitgestellt, welches die grundlegende Implementierung zum Befüllen der Datenbank zeigt. Das Beispielprogramm ist in der Sprache VBS (Visual Basic Script) erstellt. Dies kann mit dem Standard Windows-Scripting-Host ohne jede weitere Software auf allen aktuellen Windows Plattformen ausgeführt werden.

Das Beispielprogramm "ImportDB.vbs" ist im PSS SINCAL "Batch" Verzeichnis verfügbar.

Das Beispielprogramm kann in der Eingabeaufforderung gestartet werden. Beim Start ohne weitere Parameter werden Hinweise zur Benutzung ausgegeben:

```
>cscript.exe ImportDB.vbs
Usage: cscript.exe ImportDB.vbs ImportDB.mdb SincalDB.mdb MODE

MODE: E ... Import data for Electricity
MODE: W ... Import data for Water

This program reads data from ImportDB and writes the data into the SincalDB.
```

Zum Importieren von Daten muss eine vorbereitete Datenbank mit Importdaten **ImportDB.mdb** sowie eine passende PSS SINCAL Netzdatenbank **SincalDB.mdb** angegeben werden. Mit dem Parameter **MODE** wird zwischen Elektronetzen und Strömungsnetzen unterschieden.

Start von "ImportDB.vbs" mit korrekten Parametern:

```
>cscript.exe ImportDB.vbs EleData.mdb EleTest.mdb E
Init IDs...
Reading Nodes...
Reading Lines...
Reading Loads...
Reading Transformers...
Writing Data...
Node: 27/27
Element: 48/48
Terminal: 82/82
Line: 32/32
TwoWindingTransformer: 2/2
Load: 14/14
GraphicNode: 27/27
GraphicTerminal: 82/82
GraphicElement: 48/48

Successfully finished import to D:\Network\_Import\GIS\EleTest.mdb.

Inserted 362 records in 0.01 seconds.
```

3.1.1 Funktionsweise des Beispielprogramms

Der grundlegende Funktionsablauf im Beispielprogramm ist relativ einfach:

- Zuerst werden allgemeine Initialisierungen durchgeführt
- Anschließend werden die Daten aus der Quelldatenbank ausgelesen und in passende Datenstrukturen umgewandelt
- Schließlich werden die Daten mit SQL Anweisungen in die PSS SINCAL Netzdatenbank geschrieben

Im Quelltext sieht dieser Ablauf folgendermaßen aus:

```
' Execute the selected option
Select Case strParam
    Case "E"     bElectro = True
                 Call InitIDs()
                 Call ReadNodes( 1 )
                 Call ReadLines( 1 )
                 Call ReadLoads( 1 )
                 Call ReadTransformers( 1 )
    Case "W"     bElectro = False
                 Call InitIDs()
                 Call ReadFlowNodes( 1 )
                 Call ReadFlowLines( 1 )
    Case Else    Call Usage()
End Select

If ErrorCheck( "Error while reading input data!" ) Then WScript.Quit

' Write data from arrays to SINCAL database
Call WriteSINCAL()
If ErrorCheck( "Error while writing data!" ) Then WScript.Quit
```

Initialisierungen durchführen – InitIDs

Mit der Funktion **InitIDs** werden die Startwerte für die Primärschlüssel ermittelt. Dazu werden die entsprechenden Tabellen der PSS SINCAL Netzdatenbank geöffnet und jeweils das Maximum des Primärschlüssels ermittelt. Diese Werte werden dann in globalen Variablen gespeichert.

```
'-------------------------------------------------------------------------------
' Init startIDs for DB & Init base table names
'-------------------------------------------------------------------------------
Sub InitIDs

    WScript.Echo "Init IDs..."

    Call OpenDatabase( strSINCALdb )

    If bElectro = True Then
        strTableNode            = "Node"
        strTableElement         = "Element"
        strTableTerminal        = "Terminal"
        strTableGraphicText     = "GraphicText"
        strTableGraphicNode     = "GraphicNode"
        strTableGraphicElement  = "GraphicElement"
        strTableGraphicTerminal = "GraphicTerminal"
    Else
        strTableNode            = "FlowNode"
        strTableElement         = "FlowElement"
        strTableTerminal        = "FlowTerminal"
        strTableGraphicText     = "FlowGraphicText"
        strTableGraphicNode     = "FlowGraphicNode"
```

```
        strTableGraphicElement  = "FlowGraphicElement"
        strTableGraphicTerminal = "FlowGraphicTerminal"
    End If

    iNodeID             = 1 + ReadMaxID( strTableNode, "Node ID" )
    iElementID          = 1 + ReadMaxID( strTableElement, "Element ID" )
    iTerminalID         = 1 + ReadMaxID( strTableTerminal, "Terminal_ID" )

    iGraphicTextID      = 1 + ReadMaxID( strTableGraphicText, "GraphicText ID" )
    iGraphicNodeID      = 1 + ReadMaxID( strTableGraphicNode, "GraphicNode ID" )
    iGraphicElementID   = 1 + ReadMaxID( strTableGraphicElement, "GraphicElement ID" )
    iGraphicTerminalID  = 1 + ReadMaxID( strTableGraphicTerminal, "GraphicTerminal_ID" )

    Call CloseDatabase()

End Sub
```

Knotendaten aus der Datenbank auslesen – ReadNodes & AddNode

Mit der Funktion **ReadNodes** werden die Knotendaten aus der Quelldatenbank ausgelesen und in das PSS SINCAL Format konvertiert. Dabei werden direkt beim Umwandeln der Daten die passenden SQL Kommandos generiert.

```
'--------------------------------------------------------------------------------
' Read Node data from IMPORT DB
'--------------------------------------------------------------------------------
Sub ReadNodes( iMode ) ' iMode = 1 ... Normal Mode, 0 ... Only init
    Dim rsNode

    If iMode = 0 And iCntNode > 0 then Exit Sub

    WScript.Echo "Reading Nodes..."

    Call OpenDatabase(strIMPORTdb)
    Call OpenRecordset( "SELECT Name AS ID, Name, ShortName, NodeType, NetworkLevel, …
                      & "FROM Node", rsNode )

    If Not rsNode.EOF And Not rsNode.BOF Then

        Dim iRet
        rsNode.MoveFirst

        Dim pt
        Set pt = New Point

        Do While Not rsNode.EOF
            ' Names
            Dim strName, strShortName
            strName      = CStr( rsNode("Name")  )
            strShortName = Left( rsNode("ShortName"), 8 )

            ' VoltageLevel/NetworkLevel & NetworkGroup
            Dim iLevelID, iGroupID
            iLevelID = GetVoltageLevel( CDbl( rsNode("Un") ) )
            iGroupID = 1

            ' Type of Node
            Dim iType
            iType = GetNodeType( rsNode("NodeType") )

            ' Position & Height
            Dim dSh
            dSh = 1.0
            pt.SetXY CDbl( rsNode("hr") ), CDbl( rsNode("hh") )

            ' Add data of node to internal arrays
            iRet = InsertIntoNodeArray( CStr( rsNode("ID") ), iNodeID, pt, iLevelID )

            if iMode = 1 Then
```

```
                Dim iNID
                iNID = AddNode( strName, strShortName, iLevelID, iGroupID, iType, pt.x, …
                If Not pt.IsEmptyPoint Then
                    iRet = AddGraphicNode( iNID, 1, pt, pt )
                End If
            Else
                iNodeID = iNodeID + 1
            End If

            rsNode.MoveNext
        Loop

        Set pt = Nothing
    End If

    Call CloseRecordset( rsNode )
    Call CloseDatabase()
End Sub
```

Die Datensätze werden aus der Importdatenbank mit folgender SQL Anweisung ausgelesen:

```
Call OpenRecordset( "SELECT Name AS ID, Name, ShortName, NodeType, NetworkLevel, Un, hr, hh "
_
                  & "FROM Node", rsNode )
```

Anschließend werden diese Datensätze in einer Schleife verarbeitet und mittels **AddNode** in einer internen Liste gespeichert.

```
'-------------------------------------------------------------------------------
' Add Node
'-------------------------------------------------------------------------------
Function AddNode( strName , strShortName , iVoltLevelID , iGroupID , iType , dHr , dHh , dSh
)
    Dim iRet

    If bElectro Then
        iRet = InsertIntoArray( arrNode, iCntNode,
            "insert into " & strTableNode & "( Node_ID, VoltLevel_ID, Group_ID, Name, …
                & iNodeID & "," _
                & iVoltLevelID_ & "," _
                & iGroupID  & ","
                & "'" & strName  & "',"
                & "'" & strShortName  & "',"
                & iType_ & "," _
                & dHr_ & "," _
                & dHh_ & "," _
                & dSh  & ","
                & "1,1 )" )
    Else
        iRet = InsertIntoArray( arrNode, iCntNode, _
            "insert into " & strTableNode & "( Node_ID, NetworkLevel_ID, Group_ID, Name, …
                & iNodeID & "," _
                & iVoltLevelID  & ","
                & iGroupID  & ","
                & "'" & strName  & "',"
                & "'" & strShortName_ & "'," _
                & iType_ & "," _
                & dHr_ & "," _
                & dHh  & ","
                & dSh  & ","
                & "1,1 )" )
    End If

    AddNode = iNodeID
    iNodeID = iNodeID + 1
End Function
```

Die Funktion **AddNode** ist eigentlich sehr einfach. Bei jedem Funktionsaufruf wird aus den

Übergabeparametern ein SQL String generiert, welcher in das Array **arrNode** eingetragen wird. Die generierten SQL Kommandos sehen folgendermaßen aus:

```
insert into Node( Node_ID, VoltLevel_ID, Group_ID, Name, ShortName, Flag_Type,
                  hr, hh, sh, Variant_ID, Flag_Variant)
       values ( 1,2,1,'K1','K1',2,7750,21500,1,1,1 )

insert into GraphicNode( GraphicNode ID, GraphicLayer ID, GraphicType ID,
                         GraphicText ID1, Node ID, NodeStartX, NodeStartY, NodeEndX,
NodeEndY,
                         SymType, FrgndColor, BkgndColor, PenStyle, PenWidth, NodeSize, Flag,
                         Flag Variant, Variant ID )
       values ( 1,1,1,1,1,7750,21500,7750,21500,1,0,-1,0,2,4,0,1,1 )

insert into GraphicText( GraphicText ID, GraphicLayer ID, Font, FontStyle,
                         FontSize, TextAlign, TextOrient, TextColor, Visible,
                         AdjustAngle, Angle, Pos1, Pos2, Flag, RowTextNo, AngleTermNo,
                         Variant ID, Flag Variant )
       values ( 1,1,'Arial',17,11,3,0,0,1,0,0,0.25,0.25,0,0,0,1,1 )
```

Anhand dieser Kommandos wird deutlich, dass es nicht erforderlich ist, alle Attribute der Tabellen zu befüllen. Es reicht, wenn die Schlüsselattribute (durch Hervorhebung gekennzeichnet) befüllt werden. Im Wesentlichen sind dies Primärschlüssel, Fremdschlüssel und Variantenkodierung. Alle weiteren Attribute werden (sofern nicht befüllt) von PSS SINCAL automatisch mit den Defaultwerten vervollständigt.

Leitungsdaten aus der Datenbank auslesen – ReadLines

Mit der Funktion **ReadLines** werden die Leitungsdaten aus der Quelldatenbank ausgelesen und in das PSS SINCAL Format konvertiert. Dabei werden analog wie bei **ReadNodes** direkt beim Umwandeln der Daten die passenden SQL Kommandos generiert. Das Auslesen und Verarbeiten der Netzelementdaten ist allerdings etwas komplizierter, da hier mehr Tabellen als bei den Knoten befüllt werden müssen.

```
'------------------------------------------------------------------------------
' Read Line data from IMPORT db
'------------------------------------------------------------------------------
Sub ReadLines( iMode ) ' iMode = 1 ... Normal Mode, 0 ... Only init
    Dim rsLine

    Call ReadNodes( 0 )

    WScript.Echo "Reading Lines..."

    Call OpenDatabase(strIMPORTdb)
    Call OpenRecordset( "SELECT Node1 AS Node ID1, Node2 AS Node ID2, Name AS ID, Name, "
     & "'' As ShortName, "0 As Nr, l AS LineLength, r, x, c, Un, "
     & Ith, 'ON' AS Status, 'ON' AS Switch1, 'ON' AS Switch2 "
     & "FROM Line", rsLine )

    If Not rsLine.EOF and Not rsLine.BOF Then

        Dim iRet
        rsLine.MoveFirst

        Dim iTempNodeID
        iTempNodeID = 0

        Dim Node1, Node2
        Set Node1 = New Node
        Set Node2 = New Node

        Do While Not rsLine.EOF
            Dim bIsVaid
            bIsVaid = True
```

```
            Set Node1 = dctNodes.Item( CStr( rsLine("Node_ID1") ) )
            Set Node2 = dctNodes.Item( CStr( rsLine("Node_ID2") ) )

            ' Skip bad objects
            If IsEmpty( Node1 ) Or IsEmpty( Node2 ) Or Node1.iID = Node2.iID Then
                bIsVaid = False
            End If

            If IsNull( rsLine("LineLength") ) Then
                bIsVaid = False
            End If

            If bIsVaid Then
                ' VoltageLevel/NetworkLevel & NetworkGroup
                Dim iLevelID, iGroupID
                iLevelID = Node1.iLevel
                iGroupID = 1

                ' Check if there are multiple line segments - in this case we must add new
nodes
                Dim strName, strShortName, strFullName, iNr, iCntNr
                strName      = CStr( rsLine("Name") )
                strShortName = Left( rsLine("ShortName"), 8 )
                strFullName  = CStr( rsLine("Node_ID1") & "|" & rsLine("Node_ID2") & "|" …
                If dctLineSegments.Exists( strFullName ) Then iCntNr = dctLineSegments.Item(
…
                If Not IsNull( rsLine("Nr") ) Then iNr = CLng( rsLine("Nr") ) Else iNr = 0 …

                Dim iInternalID1, iInternalID2
                iInternalID1 = Node1.iID
                iInternalID2 = Node2.iID

                If iCntNr > 1 Then
                    If iNr = 1 Then
                       iTempNodeID = iNodeID
                       iInternalID2 = iTempNodeID
                    ElseIf iNr = iCntNr Then
                       iInternalID1 = iTempNodeID
                       iTempNodeID  = 0
                    Else
                       iInternalID1 = iTempNodeID
                       iTempNodeID  = iNodeID
                       iInternalID2 = iTempNodeID
                    End If
                Else
                    iTempNodeID = 0
                End If

                If iTempNodeID > 0 Then
                    Dim strTempName
                    strTempName = "K" & iNodeID
                    iRet = AddNode( strTempName, strTempName, iLevelID, iGroupID, 1, 0.0, …
                End If

                ' Process standard type mapping
                Dim iStandardType, iFlagStandardType
                iStandardType = 0
                iFlagStandardType = 0

                ' Map the input status of the element
                Dim iState
                iState = GetElementState( rsLine("Status") )

                ' Get Switches
                Dim iTermState1, iTermState2
                iTermState1 = GetSwitchState( CStr( rsLine("Switch1") ) )
                iTermState2 = GetSwitchState( CStr( rsLine("Switch2") ) )

                Dim iEleID, iTermID1, iTermID2
                iEleID   = AddElement( "Line", strName, strShortName, iLevelID, iGroupID, …
                iTermID1 = AddTerminal( iEleID, iInternalID1, 1, 7, iTermState1 )
                iTermID2 = AddTerminal( iEleID, iInternalID2, 2, 7, iTermState2 )
```

```
                iRet = InsertIntoArray( arrLine, iCntLine, _
                    "insert into Line ( Element_ID, Typ_ID, Flag_Typ_ID, l, r, x, c, " _
                        & "Un, Ith ) values ( "
                        & iEleID & ","
                        & iStandardType & ","
                        & iFlagStandardType & ","
                        & rsLine("LineLength") & ","  _
                        & rsLine("r") & ","  _
                        & rsLine("x") & ","
                        & rsLine("c") & ","
                        & rsLine("Un") & ","
                        & rsLine("Ith") _
                        & " )" )

                If Not Node1.IsPosEmpty() And Not Node2.IsPosEmpty() Then
                    Dim iGraEleID, iGraTermID1, iGraTermID2
                    iGraEleID   = AddGraphicElement( iEleID, 19, Node1.ptPos, Node2.ptPos )
                    iGraTermID1 = AddGraphicTerminal( iTermID1, iEleID, iGraEleID,
Node1.ptPos )
                    iGraTermID2 = AddGraphicTerminal( iTermID2, iEleID, iGraEleID,
Node2.ptPos )
                End If

          End If

          rsLine.MoveNext
        Loop

        Set Node1 = Nothing
        Set Node2 = Nothing
    End If

    Call CloseRecordset( rsLine )
    Call CloseDatabase()


End Sub
```

Mit der folgenden SQL Anweisung werden die Leitungsdaten aus der Importdatenbank ausgelesen:

```
Call OpenRecordset( "SELECT Node1 AS Node_ID1, Node2 AS Node_ID2, Name AS ID, Name, " _
     & "'' As ShortName, "0 As Nr, l AS LineLength, r, x, c, Un, "
     & Ith, 'ON' AS Status, 'ON' AS Switch1, 'ON' AS Switch2 " _
     & "FROM Line", rsLine )
```

Die so ausgelesenen Datensätze werden in einer Programmschleife verarbeitet. Für jeden Leitungsdatensatz werden dann die passenden SQL Insert-Kommandos erzeugt. Im Wesentlichen erfolgt dies mit den folgenden Anweisungen:

```
iEleID   = AddElement( "Line", strName, strShortName, iLevelID, iGroupID, 3, iState )
iTermID1 = AddTerminal( iEleID, iInternalID1, 1, 7, iTermState1 )
iTermID2 = AddTerminal( iEleID, iInternalID2, 2, 7, iTermState2 )

iRet = InsertIntoArray( arrLine, iCntLine,
       "insert into Line ( Element ID, Typ ID, Flag Typ ID, l, r, x, c, "
           & "Un, Ith ) values ( "
           & iEleID & "," _
           & iStandardType & "," _
           & iFlagStandardType & ","
           & rsLine("LineLength") & ","
           & rsLine("r") & ","
           & rsLine("x") & ","  _
           & rsLine("c") & ","  _
           & rsLine("Un") & ","  _
           & rsLine("Ith")
           & " )" )
```

Die Netzgrafik für die Leitung wird schließlich mit folgenden Anweisungen generiert:

```
iGraEleID   = AddGraphicElement( iEleID, 19, Node1.ptPos, Node2.ptPos )
iGraTermID1 = AddGraphicTerminal( iTermID1, iEleID, iGraEleID, Node1.ptPos )
iGraTermID2 = AddGraphicTerminal( iTermID2, iEleID, iGraEleID, Node2.ptPos )
```

Die generierten SQL Kommandos zum Erstellen der Leitungsdaten sehen folgendermaßen aus:

```
insert into Element( Element_ID, VoltLevel_ID, Group_ID,
                     Name, ShortName, Type, Flag Input, Flag State,
                     Variant_ID, Flag_Variant )
       values ( 1,1,1,'L10','','Line',3,1,1,1 )

insert into Line( Element_ID, Typ_ID, Flag_Typ_ID, l, r, x, c, Un, Ith )
       values ( 1,0,0,2.934134593,0.05,0.21,0,10,0.35 )

insert into Terminal( Terminal_ID, Element_ID, Node_ID,
                      TerminalNo, Flag_Terminal, Flag_State,
                      Variant ID, Flag Variant )
       values ( 1,1,2,1,7,1,1,1 )

insert into Terminal( Terminal ID, Element ID, Node ID,
                      TerminalNo, Flag_Terminal, Flag_State,
                      Variant_ID, Flag_Variant )
       values ( 2,1,4,2,7,1,1,1 )

insert into GraphicElement( GraphicElement ID,GraphicLayer ID,GraphicType ID,
                            GraphicText ID1,Element ID,SymbolDef,
                            FrgndColor,BkgndColor,PenStyle,PenWidth,SymbolSize,
                            SymCenterX,SymCenterY,SymbolType,SymbolNo,Flag,
                            Variant ID,Flag Variant)
       values ( 1,1,1,28,1,-1,0,-1,0,1,100,12250,22125,19,0,1,1,1 )

insert into GraphicTerminal( GraphicTerminal_ID, GraphicElement_ID,
                             GraphicText_ID, Terminal_ID, PosX, PosY,
                             FrgndColor, PenStyle, PenWidth, SwtType,
                             SwtAlign, SwtNodePos, SwtFactor, SwtFrgndColor,
                             SwtPenStyle, SwtPenWidth, SymbolType, SymbolAlign,
                             SymbolNodePos, SymbolFactor, SymbolFrgndColor,
                             SymbolPenStyle, SymbolPenWidth ,TextAlign, Flag,
                             Variant_ID, Flag_Variant )
       values ( 1,1,29,1,11250,21500,0,0,1,0,4,20,80,-1,0,1,0,4,40,80,-1,0,1,292,0,1,1 )

insert into GraphicTerminal( GraphicTerminal ID, GraphicElement ID,
                             GraphicText ID, Terminal ID, PosX, PosY,
                             FrgndColor, PenStyle, PenWidth, SwtType,
                             SwtAlign, SwtNodePos, SwtFactor, SwtFrgndColor,
                             SwtPenStyle, SwtPenWidth, SymbolType, SymbolAlign,
                             SymbolNodePos, SymbolFactor, SymbolFrgndColor,
                             SymbolPenStyle, SymbolPenWidth ,TextAlign, Flag,
                             Variant ID, Flag Variant )
       values ( 2,1,30,2,13250,22750,0,0,1,0,4,20,80,-1,0,1,0,4,40,80,-1,0,1,292,0,1,1 )

insert into GraphicText( GraphicText ID, GraphicLayer ID, Font, FontStyle,
                         FontSize, TextAlign, TextOrient, TextColor, Visible,
                         AdjustAngle, Angle, Pos1, Pos2, Flag, RowTextNo, AngleTermNo,
                         Variant_ID, Flag_Variant )
       values ( 28,1,'Arial',17,11,3,0,0,1,0,0,0.25,0.25,0,0,0,1,1 )

insert into GraphicText( GraphicText ID, GraphicLayer ID, Font, FontStyle,
                         FontSize, TextAlign, TextOrient, TextColor, Visible,
                         AdjustAngle, Angle, Pos1, Pos2, Flag, RowTextNo, AngleTermNo,
                         Variant_ID, Flag_Variant )
       values ( 29,1,'Arial',17,11,3,0,0,1,0,0,0.25,0.25,0,0,0,1,1 )

insert into GraphicText( GraphicText ID, GraphicLayer ID, Font, FontStyle,
                         FontSize, TextAlign, TextOrient, TextColor, Visible,
                         AdjustAngle, Angle, Pos1, Pos2, Flag, RowTextNo, AngleTermNo,
                         Variant_ID, Flag_Variant )

       values ( 30,1,'Arial',17,11,3,0,0,1,0,0,0.25,0.25,0,0,0,1,1 )
```

Speichern der ausgelesenen Daten – WriteSINCAL

Mit der Funktion **WriteSINCAL** werden die zuvor ausgelesenen und in SQL Insert-Kommandos umgewandelten Daten in die PSS SINCAL Netzdatenbank geschrieben.

```
'-------------------------------------------------------------------------------
' Insert data into SINCAL database
'-------------------------------------------------------------------------------
Sub WriteSINCAL()
    WScript.Echo "Writing Data..."

    Call OpenDatabase( strSINCALdb )

    ' Nodes
    Call InsertRecords( iCntNode, arrNode, "Node: " )

    ' Element & Terminal
    Call InsertRecords( iCntElement,  arrElement,  "Element: " )
    Call InsertRecords( iCntTerminal, arrTerminal, "Terminal: " )

   ' Lines
    Call InsertRecords( iCntLine, arrLine, "Line: " )

   ' Transfomer
    Call InsertRecords( iCntTransformer, arrTransformer, "TwoWindingTransformer: " )

    ' Load & Customer data
    Call InsertRecords( iCntLoad,         arrLoad,         "Load: " )

    ' Graphics
    Call InsertRecords( iCntGraphicNode,     arrGraphicNode,     "GraphicNode: " )
    Call InsertRecords( iCntGraphicText,     arrGraphicText,     "GraphicText: " )
    Call InsertRecords( iCntGraphicTerminal, arrGraphicTerminal, "GraphicTerminal: " )
    Call InsertRecords( iCntGraphicElement,  arrGraphicElement,  "GraphicElement: " )

    Call CloseDatabase()
End Sub
```

3.2 Hilfsprogramm zum Erzeugen der PSS SINCAL Netzdatenbank

Für eigene Kopplungslösungen wird eine leere PSS SINCAL Netzdatenbank benötigt, um diese dann mit den eigenen Daten zu füllen. Für diesen Zweck ist das in der PSS SINCAL Installation enthaltene Hilfsprogramm **SinDBCreate.exe** konzipiert. SinDBCreate bietet die Möglichkeit, PSS SINCAL Netzdatenbanken sowie Standard- und Schutzgerätedatenbanken ohne die PSS SINCAL Benutzeroberfläche anzulegen.

Das Programm wird in einer Eingabeaufforderung gestartet. Es verfügt über keine grafische Benutzerschnittstelle, d.h. die Steuerung erfolgt durch Startparameter. Zusätzlich werden diverse Einstellungen aus der PSS SINCAL Registry ausgelesen und verwendet (z.B. Datenbankkonfiguration Oracle), sofern diese nicht als Parameter angegeben werden.

Beim Start des Programms ohne Parameter werden folgende Informationen ausgegeben:

```
C:\> SinDBCreate.exe

Usage:
SinDBCreate /DBSYS:xxx /FILE:xxx /TYPE:xxx [Options]
    Create a new SINCAL-Database.

SinDBCreate /LIST /DBSYS:xxx /ADMIN:User/Password /SRV:xxx
    List all Databases on a server.

SinDBCreate /DELETE /DBSYS:xxx /FILE:xxx /ADMIN:User/Password /SRV:xxx
    Delete a SINCAL-Database on a database server.
```

```
    /DBSYS:{ACCESS|ORACLE|SQLSERVER|SQLEXPRESS}
                                Database-System

    /FILE:{Database}
        MS Access:              Path and FileName of the MDB-File
        SQL Server Express:     Path and Filename of the MDF-Datafile
        ORACLE:                 User/Password@Instance
        SQL Server:             Database@Instance

    /ADMIN:User/Password        Administrator-Login for Database-Servers
    /USER:User/Password         Login Information for Database-Servers
    /SRV:Instance               Database Service Name/Server Name

    /TYPE:{E|W|G|H}             Network-Type (E)lectro|(W)ater|(G)as|(H)eating
    [/DB:{NET|STD|PROT}]        Database-Type (Network-Database is default]
    [/DATA]                     Fills STD-DB and Prot-DB with default data
    [/LANG:{ENG|GER}]           Language for database (default is ENG)
    [/SIN:Filename]             Path and filename of the SIN-file.
```

PSS SINCAL Datenbank erstellen

Die Hauptfunktionalität von SinDBCreate ist die Erstellung von PSS SINCAL Datenbanken. Hierbei werden alle Einstellungen, die für das Erstellen notwendig sind, als Parameter angegeben.

```
SinDBCreate /DBSYS:xxx /FILE:xxx /TYPE:xxx [Options]
    Create a new SINCAL-Database.
```

Zwingende Parameter

```
/DBSYS:{ACCESS|ORACLE|SQLSERVER|SQLEXPRESS}
                            Database-System

/FILE:{Database}
    MS Access:              Path and FileName of the MDB-File
    SQL Server Express:     Path and Filename of the MDF-Datafile
    ORACLE:                 User/Password@Instance
    SQL Server:             Database@Instance

/ADMIN:User/Password        Administrator-Login for Database-Servers
/USER:User/Password         Login Information for Database-Servers
/SRV:Instance               Database Service Name/Server Name

/TYPE:{E|W|G|H}             Network-Type (E)lectro|(W)ater|(G)as|(H)eating
```

Optionale Parameter

```
[/DB:{NET|STD|PROT}]        Database-Type (Network-Database is default]
[/DATA]                     Fills STD-DB and Prot-DB with default data
[/LANG:{ENG|GER}]           Language for database (default is ENG)
[/SIN:Filename]             Path and filename of the SIN-file.
```

Der Parameter **DBSYS** legt das zu verwendende Datenbanksystem fest. Hierbei kann zwischen **ACCESS** (Microsoft Access), **ORACLE**, **SQLSERVER** (SQL Server) und **SQLEXPRESS** (SQL Server Express) gewählt werden.

Der Parameter **FILE** kennzeichnet den PSS SINCAL Datenbanknamen. Abhängig vom verwendeten Datenbanksystem ist dieser Parameter unterschiedlich anzugeben. Bei Microsoft Access und SQL Server Express werden der vollständige Pfad und Dateiname angegeben, bei Oracle werden Benutzername, Kennwort und Servername im Format "Benutzer/Kennwort@Server" angegeben. Bei

der Verwendung vom SQL Server wird der Datenbankname und Servername im Format "Datenbankname@Server" angegeben.

Der Parameter **ADMIN** ist bei der Verwendung der Datenbanksysteme Oracle und SQL Server erforderlich. Dieser ermöglicht es, den Hauptbenutzer zur Verwaltung von PSS SINCAL Netzen anzugeben. Dieser Parameter wird im Format "Benutzer/Kennwort" definiert. Wird dieser Parameter nicht angegeben, so werden die Einstellungen aus der PSS SINCAL Registry geladen.

Der Parameter **USER** wird bei der Verwendung von SQL Server als Datenbanksystem benötigt. Hierbei handelt es sich um die Anmeldeinformationen des Benutzers am SQL Server und wird im Format "Benutzername/Kennwort" angegeben.

Der Parameter **SRV** ermöglicht die explizite Angabe des Datenbankservers. Wird dieser Parameter nicht angegeben, so wird der Servername aus der PSS SINCAL Registry geladen.

Mittels Parameter **TYPE** wird der Netztyp angegeben. Hierbei kann zwischen **E** (Elektronetz), **W** (Wassernetz), **G** (Gasnetz) und **H** (Fernwärme-/Fernkältenetz) gewählt werden.

Alle weiteren Parameter sind optional und werden zur Steuerung des Erstellungsvorganges verwendet.

Der Parameter **DB** dient zur Festlegung des zu erstellenden Datenbanktyps. Dieser ist standardmäßig auf Netzdatenbank (**NET**) eingestellt. Weitere Werte für diesen Parameter sind **STD** für die Standardtypdatenbank und **PROT** für die Schutzgerätedatenbank.

Der Parameter **DATA** bewirkt, dass die Standarddatenbank bzw. Schutzgerätedatenbank mit den Standardtypen/-geräten befüllt wird.

Der Parameter **LANG** bietet die Möglichkeit, die Sprache (Englisch, Deutsch) der Netzdatenbank auszuwählen.

Der Parameter **SIN** ermöglicht die Angabe der PSS SINCAL Netzdatei. Dieser Parameter ist nur bei der Erstellung von Netzdatenbanken möglich.

Beispiel für die Erstellung einer Netzdatenbank

```
C:\> SinDBCreate.exe /DBSYS:ACCESS /FILE:C:\Temp\dbnet.mdb /TYPE:E
```

Das obige Beispiel erstellt die Access Datenbank "dbnet.mdb" für ein Elektronetz im Verzeichnis "C:\Temp" in englischer Sprache.

Beispiel für die Erstellung einer Standarddatenbank

```
C:\> SinDBCreate.exe /DBSYS:ORACLE /FILE:OraSTDFL/pwd123@ORA10 /TYPE:W /ADMIN:SINCAL/SINCAL
/DB:STD /DATA
```

Das obige Beispiel erstellt eine Oracle Standarddatenbank für Strömungsnetze in englischer Sprache. Diese Datenbank wird unter dem Oracle Benutzer "OraSTDFL" mit dem Kennwort "pwd123" angelegt. Zusätzlich wird die Datenbank mit den Standardtypen befüllt.

Beispiel für die Erstellung einer Schutzgerätedatenbank

```
C:\> SinDBCreate.exe /DBSYS:ACCESS FILE:C:\Temp\stdprot.mdb /TYPE:E /DB:PROT
```

Das obige Beispiel erstellt eine Access Schutzgerätedatenbank für Elektronetze in englischer Sprache. Die Datenbank "stdprot.mdb" wird im Verzeichnis "C:\Temp" angelegt. Es wird eine leere Datenbank erstellt.

PSS SINCAL Datenbanken auflisten

Das Hilfsprogramm SinDBCreate bietet neben der Erstellung von PSS SINCAL Datenbanken auch die Möglichkeit, alle Datenbanken auf einem Datenbankserver aufzulisten. Diese Funktion wird mit dem Parameter **LIST** aktiviert.

```
SinDBCreate /LIST /DBSYS:xxx /ADMIN:User/Password /SRV:xxx
    List all Databases on a server.
```

Zwingende Parameter

```
/DBSYS:{ORACLE|SQLSERVER}
                              Database-System

/ADMIN:User/Password          Administrator-Login for Database-Servers
/SRV:Instance                 Database Service Name/Server Name
```

Der Parameter **DBSYS** legt das zu verwendende Datenbanksystem fest. Hierbei kann zwischen **ORACLE** und **SQLSERVER** (SQL Server) gewählt werden.

Der Parameter **ADMIN** ist bei der Verwendung der Datenbanksysteme Oracle und SQL Server erforderlich. Dieser ermöglicht es, den Hauptbenutzer zur Verwaltung von PSS SINCAL Netzen anzugeben. Dieser Parameter wird im Format "Benutzer/Kennwort" definiert. Wird dieser Parameter nicht angegeben, so werden die Einstellungen aus der PSS SINCAL Registry geladen.

Der Parameter **SRV** ermöglicht die explizite Angabe des Datenbankservers. Wird dieser Parameter nicht angegeben, so wird der Servername aus der PSS SINCAL Registry geladen.

Beispiel

```
C:\> SinDBCreate /LIST /DBSYS:ORACLE /ADMIN:SINCAL/SINCAL /SRV:ORA10
```

Das obige Beispiel listet alle verfügbaren PSS SINCAL Datenbanken auf. Hierzu wird eine Verbindung zu der Oracle Instanz mit dem Namen "ORA10" und dem Benutzer "SINCAL" hergestellt. Bei erfolgreicher Verbindung werden die verfügbaren Datenbanken zeilenweise ausgegeben.

PSS SINCAL Datenbank löschen

Das Hilfsprogramm SinDBCreate bietet neben der Erstellung von PSS SINCAL Datenbanken auch die Möglichkeit, eine PSS SINCAL Datenbank auf einem Datenbankserver zu löschen. Diese Funktion wird mit dem Parameter **DELETE** aktiviert.

```
SinDBCreate /DELETE /DBSYS:xxx /FILE:xxx /ADMIN:User/Password /SRV:xxx
    Delete a SINCAL-Database on a database server.
```

Zwingende Parameter

```
/DBSYS:{ORACLE|SQLSERVER}
                            Database-System

/FILE:{Database}
    ORACLE:                 User
    SQL Server:             Database

/ADMIN:User/Password        Administrator-Login for Database-Servers
/SRV:Instance               Database Service Name/Server Name
```

Der Parameter **DBSYS** legt das zu verwendende Datenbanksystem fest. Hierbei kann zwischen **ACCESS** (Microsoft Access), **ORACLE**, **SQLSERVER** (SQL Server) und **SQLEXPRESS** (SQL Server Express) gewählt werden.

Der Parameter **FILE** kennzeichnet den PSS SINCAL Datenbanknamen. Abhängig vom verwendeten Datenbanksystem ist dieser Parameter unterschiedlich anzugeben. Bei Oracle wird der Benutzername angegeben. Bei der Verwendung vom SQL Server wird der Datenbankname angegeben.

Der Parameter **ADMIN** ist bei der Verwendung der Datenbanksysteme Oracle und SQL Server erforderlich. Dieser ermöglicht es, den Hauptbenutzer zur Verwaltung von PSS SINCAL Netzen anzugeben. Dieser Parameter wird im Format "Benutzer/Kennwort" definiert. Wird dieser Parameter nicht angegeben, so werden die Einstellungen aus der PSS SINCAL Registry geladen.

Der Parameter **SRV** ermöglicht die explizite Angabe des Datenbankservers. Wird dieser Parameter nicht angegeben, so wird der Servername aus der PSS SINCAL Registry geladen.

Beispiel

```
C:\> SinDBCreate /DELETE /DBSYS:ORACLE /FILE:SINCAL_TEST /ADMIN:SINCAL/SINCAL /SRV:ORA10
```

Das obige Beispiel löscht die PSS SINCAL Datenbank "SINCAL_TEST". Hierzu wird eine Verbindung zu der Oracle Instanz mit dem Namen "ORA10" und dem Benutzer "SINCAL" hergestellt. Das Löschen der PSS SINCAL Datenbank kann nicht rückgängig gemacht werden.

4 Automatisierung der Berechnungsmethoden

Die PSS SINCAL Architektur basiert auf einem System von verschiedenen Komponenten, die über COM-Funktionen kommunizieren (siehe Referenz – PSS SINCAL Architektur). In diesem Kapitel wird erläutert, wie die Berechnungsmethoden unter Verwendung der COM-Funktionen in eigene Anwendungen integriert werden können.

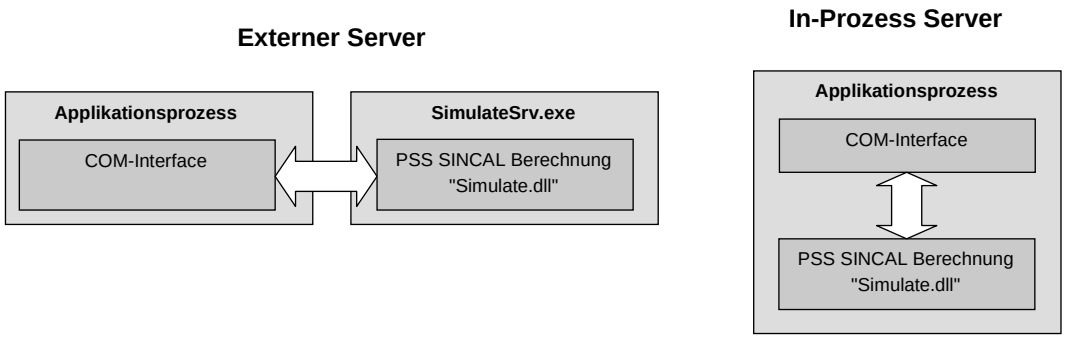
Die Erklärungen mit Codeauszügen und Beispielen erfolgen anhand des **Windows Scripting Host (WSH)**, da dieser die einfachste Syntax hat und direkt in den aktuellen Betriebssystemen standardmäßig verfügbar ist. Zur Nutzung der Automatisierungsfunktionen kann aber jede beliebige Programmiersprache verwendet werden, die die Verwendung von COM-Funktionen unterstützt (z.B. VisualBasic, VBA, C++, usw.).

Durch das offene Design der Berechnungsmethoden können diese für vielfältige Aufgabenstellungen genutzt werden. Im Wesentlichen kann aber zwischen der **Integration in externen Anwendungen** und der **Nutzung für eigene Lösungen** unterschieden werden.

Integration der Berechnungsmethoden in externen Anwendungen

Diese Form der Automatisierung wird vor allem für integrierte Lösungen in GIS, NIS oder SCADA Systemen genutzt. Hierbei wird die PSS SINCAL Berechnung direkt aus dem jeweiligen Quellsystem gestartet. Die vollständige Datenhaltung und -bearbeitung sowie die Visualisierung der Ergebnisse erfolgen direkt im Quellsystem.

Bei dieser Automatisierungslösung werden die Berechnungsmethoden direkt in das Quellsystem integriert. Die Anbindung erfolgt über COM-Interfaces, wobei die Berechnungsmethoden wahlweise als **Externer Server** (getrennte Prozesse) oder als **In-Prozess Server** (im selben Prozess) genutzt werden können.



Für hochperformante Lösungen können alle Netzdaten und natürlich auch die Ergebnisse in "virtuellen Tabellen" direkt in der PSS SINCAL Berechnung verwaltet werden. Der Aufbau dieser virtuellen Tabellen entspricht exakt dem PSS SINCAL Datenmodell. Für das Schreiben der Netzdaten, Abholen der Ergebnisse und zur Steuerung der Berechnungsmethoden sind COM-Interfaces verfügbar.

Ein C++ Beispielprogramm, welches die Integration der Berechnungsmethoden in eine externe Anwendung und auch die Verwendung von virtuellen Tabellen zeigt, ist in der PSS SINCAL Installationsstruktur verfügbar ("Batch\SimAuto.zip").

Nutzung der Berechnungsmethoden in eigenen Lösungen

Die Idee hierbei ist, dass die in PSS SINCAL verfügbaren Berechnungsverfahren (Lastfluss, Kurzschluss, usw.) als Grundlage für individuelle Problemlösungen und Analysen verwendet werden. Dabei wird im Normalfall ein bestehendes PSS SINCAL Netz genutzt, welches dann mit eigenen Lösungen analysiert wird.

Hierzu ein simples Beispiel: In einem Netz soll die Auswirkung einer Laststeigerung untersucht werden. Dazu wird ein Lastwert in Schritten gesteigert und gleichzeitig der Spannungspegel an den Knoten untersucht.

Für diese Problemstellung ist das kleine Beispielprogramm "VoltageDrop.vbs" in der PSS SINCAL Installationsstruktur verfügbar.

```
'------------------------------------------------------------------------------
' File:        VoltageDropBatch.vbs
' Description: Small sample for simulation automation.
'              A load at a node is constatly increased  until a specified
'              voltage drop occurs.
' Author:      SS, GM
' Modified:    14.03.2008
'------------------------------------------------------------------------------
Option Explicit

const siSimulationOK = 1101

Dim strDatabase ' Database of sincal network
strDatabase = "D:\Network\_Test\Example Ele.mdb"

Dim strProtDatabase ' Database with protection devices
strProtDatabase = "D:\Server-Setup\Database\ProtectionDB.mdb"

Dim strLoad ' Name of Load to be changed
strLoad = "LO8"

Dim strLF ' Load flow procedure
strLF = "LF NR"

' Set locale to US -> necessary because '.' is required for SQL commands!
SetLocale( "en-gb" )

'------------------------------------------------------------------------------
' Start of the script
'------------------------------------------------------------------------------
If Not UCase( Right(WScript.Fullname,11) ) = "CSCRIPT.EXE" Then
    Call Usage()
    WScript.Quit
End If

' Create an simulation object as "in process server"
Dim SimulateObj
Set SimulateObj = WScript.CreateObject( "Sincal.Simulation" )

If SimulateObj is Nothing Then
    WScript.Echo "Error: CreateObject Sincal.Simulation failed!"
    WScript.Quit
End If

' Setting databases and language
SimulateObj.DataSourceEx "DEFAULT", "JET", strDatabase, "Admin", ""
SimulateObj.DataSourceEx "PROT", "JET", strProtDatabase, "Admin", ""
SimulateObj.Language "US"

' Enable simulation batch mode: load from phys. database, store to virtual database
SimulateObj.BatchMode 1

' Load from database and generating calculation objects
SimulateObj.LoadDB CStr( strLF )
```

```
' Getting calculation object load for modifying
Dim LoadObj
Set LoadObj = SimulateObj.GetObj( "LOAD", CStr( strLoad ) )
If LoadObj is Nothing Then
    WScript.Echo "Error: Load " & strLoad & " not found!"
    WScript.Quit
End If

' Getting calculation object node of load
Dim NodeID
NodeID = LoadObj.Item( "TOPO.NODE1.DBID" )
Dim LoadNode
Set LoadNode = SimulateObj.GetObj( "NODE", NodeID )

' Getting virtual database object
Dim SimulateNetworkDataSource
Set SimulateNetworkDataSource = SimulateObj.DB_EL
If SimulateNetworkDataSource Is Nothing Then
    WScript.Echo "Error: getting virtual database object failed!"
    WScript.Quit
End If

'-------------------------------------------------------------------------------
' Perform special voltage drop analysis
'-------------------------------------------------------------------------------
Dim iLoop, iLoopErr
iLoop = 0
iLoopErr = 0

WScript.Echo vbCrLf & "Start load flow calculation (" & strLF & ")"

Do While iLoop < 1000
    WScript.Echo vbCrLf & "-------- " & CStr( iLoop ) & " --------"

    ' We modify the load by adding 0.1 MW in each loop
    Call ModifyLoad( LoadObj, 0.1 )

    ' Start loadflow simulation
    SimulateObj.Start strLF
    If SimulateObj.StatusID <> siSimulationOK Then
        WScript.Echo "Load flow failed!"
        Exit Do
    End If

    ' Getting load flow result for node
    If LoadNode Is Nothing Then
    Else
        Dim LFNodeResultLoad
        Set LFNodeResultLoad = LoadNode.Result( "LFNODERESULT", 0 )
        If LFNodeResultLoad Is Nothing Then
        Else
            Dim u_un
            u_un = LFNodeResultLoad.Item( "U_Un" )
            WScript.Echo "Node voltage at modified load U/Un = " & FormatNumber( u_un ) & "%"
            Set LFNodeResultLoad = Nothing
        End If
    End If

    ' Display some golbal result information
    Call OutputLFAccurResult( SimulateNetworkDataSource )

    iLoop = iLoop + 1
Loop

' Write calculation messages
Call WriteMessages( SimulateObj )

' Release used objects
Set SimulateNetworkDataSource = Nothing
Set LoadObj = Nothing
Set LoadNode = Nothing
Set SimulateObj = Nothing
Set SimulateObj = Nothing
```

```
'-------------------------------------------------------------------------------
' Modify load
'-------------------------------------------------------------------------------
Sub ModifyLoad( ByRef LoadObj , ValAdd )

    ' Modify load by increasing P and Q
    Dim P
    P = LoadObj .Item( "P" ) + ValAdd
    LoadObj .Item( "P" ) = P

    Dim Q
    Q = LoadObj_.Item( "Q" ) + ValAdd
    LoadObj .Item( "Q" ) = Q

    WScript.Echo "Set load " & strLoad & " to P = " & P & "MW, Q = " & Q  & "Mvar"

End Sub

'-------------------------------------------------------------------------------
' Output some data of LFAccurResult to console
'-------------------------------------------------------------------------------
Sub OutputLFAccurResult( ByRef SimulateNetworkDataSource )

    ' Get datbase object LFAccurResult from virtual database
    Dim LFAccurResult
    Set LFAccurResult = SimulateNetworkDataSource.GetRowObj( CStr( "LFAccurResult" ) )
    If LFAccurResult Is Nothing Then
        WScript.Echo "Error: cant get objects in LFAccurResult!"
        WScript.Quit
    End If

    ' Open table LFAccurResult
    Dim hr
    hr = LFAccurResult.Open
    If hr <> 0 Then
        WScript.Echo "Error: cant open LFAccurResult!"
        WScript.Quit
    End If

    ' Move cursor to first row
    Dim bRead_next_data
    bRead next data = LFAccurResult.MoveFirst

    If bRead next data = 0 Then
        ' Get attribut Iteration Number
        Dim IterCnt
        IterCnt = LFAccurResult.Item( "IT" )

        ' Get attribute Power Node Balance
        Dim PNB
        PNB = LFAccurResult.Item( "PNB" )

        ' Get attribute Power Node Balance
        Dim PNBre
        PNBre = LFAccurResult.Item( "PNBre" )

        'Get attribute Voltage Mesh Balance
        Dim VLB
        VLB = LFAccurResult.Item( "VLB" )

        'Get attribut Voltage Mesh Balance
        Dim VLBre
        VLBre = LFAccurResult.Item( "VLBre" )

        'Output to console
        WScript.Echo "IT = " & IterCnt & ", Power Accuracy PNBre = " & FormatNumber( PNBre /
1000.0 ) & "kW"
    End If

    ' Release datbase object LFAccurResult
    Set LFAccurResult = nothing

End Sub
```

```
'-------------------------------------------------------------------------------
' Write simulation messages
'-------------------------------------------------------------------------------
Sub WriteMessages( ByRef SimulateObj )

    WScript.Echo vbCrLf & "Simulation Messages:" & vbCrLf


    Dim objMessages
    Set objMessages = SimulateObj.Messages

    Dim strType
    Dim intMsgIdx
    For intMsgIdx = 1 To objMessages.Count
        Dim Msg
        Set Msg = objMessages.Item( intMsgIdx )

        Select Case Msg.Type
            case 1 ' STATUS
            case 2 ' INFO
            case 3 ' WARNING
                WScript.Echo Msg.Text
            case 4 ' ERROR
                WScript.Echo Msg.Text
        End Select

        Set Msg = Nothing
    Next
    Set objMessages = Nothing
End Sub

'-------------------------------------------------------------------------------
' Show usage
'-------------------------------------------------------------------------------
Sub Usage()
    Dim strUsage
    strUsage = "Usage: cscript.exe VoltageDropBatch.vbs"
             & vbCrLf & vbCrLf
             & "A load at a node is constatly increased until a specified " _
             & "voltage drop occurs." _
             & vbCrLf
    WScript.Echo strUsage
End Sub
```

Das Beispielprogramm kann in der Eingabeaufforderung wie folgt gestartet werden:

```
> cscript.exe VoltageDrop.vbs
```

Nach dem Start wird im Normalfall eine Fehlermeldung ausgegeben. Der Grund dafür ist auch einfach. Im Beispielprogramm ist statisch hinterlegt, welche PSS SINCAL Netzdatenbank zur Berechnung verwendet werden soll. Für eigene Experimente müssen diese globalen Voreinstellungen angepasst werden.

```
Dim strDatabase ' Database of sincal network
strDatabase = "D:\Network\_Test\Example Ele.mdb"

Dim strProtDatabase ' Database with protection devices
strProtDatabase = "D:\Server-Setup\Database\ProtectionDB.mdb"

Dim strLoad ' Name of Load to be changed
strLoad = "LO8"
```

Zuerst sollte der Inhalt der Variable **strDatabase** geändert werden. Hier wird das Netz angegeben, welches berechnet werden soll.

Die Variable **strProtDatabase** muss ebenfalls angepasst werden. Hiermit wird die globale Schutzgerätedatenbank definiert. Diese ist im "Database" Verzeichnis der PSS SINCAL Installation verfügbar.

Mit der Variable **strLoad** wird jene Last spezifiziert, die gesteigert werden soll. Im vorliegenden Beispiel ist dies die Last mit dem Namen "L08".

Wenn das Beispielprogramm nach dem Anpassen der Datenbankenpfade erneut gestartet wird, dann wird folgender Text ausgeben.

```
>cscript.exe VoltageDrop.vbs

-------- 0 --------
Set load LO8 to P = 0.5MW, Q = 0.4Mvar
Node voltage at modified load U/Un = 92.37%
IT = 9, Power Accuracy PNBre = 0.00kW

-------- 1 --------
Set load LO8 to P = 0.6MW, Q = 0.5Mvar
Node voltage at modified load U/Un = 91.75%
IT = 9, Power Accuracy PNBre = 0.00kW

-------- 2 --------
Set load LO8 to P = 0.7MW, Q = 0.6Mvar
Node voltage at modified load U/Un = 91.13%
IT = 10, Power Accuracy PNBre = 0.00kW

...

-------- 33 --------
Set load LO8 to P = 3.8MW, Q = 3.7Mvar
Node voltage at modified load U/Un = 55.96%
IT = 10, Power Accuracy PNBre = 0.00kW

-------- 34 --------
Set load LO8 to P = 3.9MW, Q = 3.8Mvar
Load flow failed!

Simulation Messages:

W 2714: Element data not physically meaningful
E 3101: Load flow: no convergence break after 200 iterations
E 1070: Please have a look at System Manual - Technical Reference - Messages from
Calculations - Errors for further error information
```

Nun konnte das Programm ohne Fehler ausgeführt werden. Dabei wurden insgesamt 34 Lastflussberechnungen durchgeführt. Vor jeder Berechnung wurde der Lastwert von "L08" jeweils um 0,1 MW erhöht. Beim 33. Umlauf war noch eine Konvergenz mit einer Knotenspannung von 55,96 % möglich. Beim 34. Umlauf ist keine Konvergenz mehr möglich. D.h. der maximal zulässige Belastungswert ist hier P = 3,8 MW und Q = 3,7 Mvar.

4.1 Verfügbare Automatisierungsfunktionen

Die Automatisierungsobjekte in den Berechnungsmethoden sind hierarchisch strukturiert. Ausgehend vom übergeordneten Objekt können die jeweils untergeordneten Objekte instanziiert werden. Die Objekte selbst stellen dann die verschiedenen Methoden und Funktionen zur Verfügung.

Bild: Hierarchische COM-Objektstruktur der Berechnungsmethoden

Übersicht der verfügbaren Automatisierungsfunktionen

Simulationsobjekt:

- BatchMode: Virtuelle Datenbank aktivieren
- DataSourceEx: Voreinstellen der Datenbanken
- Database: Voreinstellen der Datenbanken
- SQLUser: Voreinstellen des SQL Benutzers
- DataFile: Voreinstellen der Datendatei
- MacroPath: Pfade für Modelle
- Language: Voreinstellen der Sprache
- Currency: Voreinstellen der Währung
- SetInputState: Setzen des Eingabestatus
- LoadDB: Laden der Eingabedaten aus der Datenbank
- SaveDB: Speichern der Ergebnisse in die Datenbank
- AddObjID: Objekte hinzufügen
- Parameter: Setzen und Abfragen von globalen Parametern
- DoCommand: Anweisungen ausführen
- Start: Starten der Berechnung
- StatusID: Statuscode des Berechnungsvorganges
- GetObj: Zugriff auf Berechnungsobjekte
- GetObjById, GetObjByGUID: Zugriff auf Berechnungsobjekte über ID
- DB_EL, DB_FLOW: Zugriff auf die Datenbankobjekte

- Messages: Zugriff auf die Meldungsobjekte

Berechnungsobjekt:

- Count: Anzahl der möglichen Attribute
- Name: Attributnamen ermitteln
- Item: Zugriff auf Attribute
- Result: Zugriff auf Berechnungsergebnisobjekt

Berechnungsergebnisobjekt:

- Count: Anzahl der möglichen Attribute
- Name: Attributnamen ermitteln
- Item: Zugriff auf Attribute

Datenbankobjekt:

- GetRowObj: Instanz eines Tabellenobjektes ermitteln

Tabellenobjekt:

- Open: Öffnen des Tabellenobjektes
- Close: Schließen eines Tabellenobjektes
- CountRows: Ermitteln der Datensatzanzahl
- MoveFirst, MoveLast, MoveNext, MovePrev: Positionierung in der Datenmenge
- Count: Anzahl der möglichen Attribute
- Name: Attributnamen ermitteln
- Item: Zugriff auf Attribute

Meldungsobjekt:

- Count: Anzahl der verfügbaren Meldungen
- Item: Zugriff auf ein Meldungsdatenobjekt

Meldungsdatenobjekt:

- Text: Meldungstext
- Type: Meldungstyp
- CountObjectIds: Anzahl der Netzelemente
- ObjectIdAt, ObjectTypeAt: Netzelementdaten

Attribute der Berechnungsobjekte:

- Attribute für Elektronetze
- Attribute für Strömungsnetze

4.1.1 Simulationsobjekt

Dieses Objekt bildet die Basis für alle weiteren Automatisierungsfunktionen. Es ist ein Abbild des PSS SINCAL Berechnungsmoduls und somit das Hauptobjekt jeglicher Automatisierung. Die Instanziierung des Simulationsobjektes kann wahlweise als **Local Server** oder als **In-Process Server** erfolgen.

Local Server

Local Server sind ausführbare Programme, die COM-Komponenten implementieren. Bei Instanziierung einer COM-Komponente wird dieses Programm als eigener Hintergrundprozess gestartet.

Zur Kommunikation zwischen den Prozessen wird ein spezielles RPC-Protokoll (Remote Procedure Call) genutzt, wodurch die Geschwindigkeit beim Datenaustausch verlangsamt wird. Der Vorteil ist allerdings, dass hier vollständig getrennte Prozess- und Speichermodelle verwendet werden. D.h. selbst schwerwiegende Programmfehler beeinflussen den jeweils anderen Prozess überhaupt nicht.

Mit den folgenden Anweisungen wird die Berechnung als neuer Prozess gestartet.

```
Set SincalSimSrv = WScript.CreateObject( "Sincal.SimulationSrv" )
If SincalSimSrv Is Nothing Then
    WScript.Echo "Error: CreateObject Sincal. SincalSimSrv failed!"
    WScript.Quit
End If
```

Der Zugriff auf das Simulationsobjekt erfolgt via COM-Interface.

```
Set SincalSim = SincalSimSrv.GetSimulation
If SincalSim Is Nothing Then
    WScript.Echo "Error: GetSimulation failed!"
    WScript.Quit
End If
```

In-Process Server

Im Falle des In-Process Servers werden die Schnittstellen in einer DLL zur Verfügung gestellt. Wird eine COM-Komponente eines In-Process Servers instanziiert, so wird der zugehörige Server in den aktuellen Prozess geladen. In-Process Server sind besonders schnell, da der Zugriff auf die Funktionen der Schnittstellen innerhalb der Prozessgrenzen erfolgt.

Mit der folgenden Anweisung wird die Berechnung im "aktuellen" Prozess als zusätzliche COM-Komponente instanziiert.

```
Set SincalSim = WScript.CreateObject( "Sincal.Simulation" )
If SincalSim Is Nothing Then
    WScript.Echo "Error: CreateObject Sincal.Simulation failed!"
    WScript.Quit
```

```
End If
```

BatchMode – Virtuelle Datenbank aktivieren

Ändert den Datenbankmodus der Berechnung.

```
SimulateObj.BatchMode iMode
```

Parameter

iMode (Integer)
Datenbankmodus der Berechnung. Der Modus entspricht einem Zahlenwert von 0 bis 2.

Kennziffer	Beschreibung
0	Laden aus reeller Datenbank, Speichern in reelle Datenbank
1	Laden aus reeller Datenbank, Speichern in virtuelle Datenbank
2	Laden aus virtueller Datenbank, Speichern in virtuelle Datenbank
4	Laden aus reeller in virtuelle Datenbank, Speichern in virtuelle Datenbank

Anmerkungen

Im normalen Simulationsfall werden die Ergebnisdaten direkt in der Datenbank gespeichert. Durch die Funktion **Batchmode** kann das Schreiben der Ergebnisdaten allerdings in eine virtuelle Datenbank umgeleitet werden. Damit wird die Geschwindigkeit um ein Vielfaches gesteigert, da das zeitaufwendige Eintragen in die Datenbank komplett entfällt. Die Ergebnisse werden dann nur in der virtuellen Datenbank im Hauptspeicher der Berechnung vorgehalten.

Beispiel

```
' Enable virtual database.
SimulateObj.BatchMode 1
```

DataSourceEx – Voreinstellen der Datenbanken

Definiert die zur Berechnung verwendeten Datenbanken.

```
SimulateObj.DataSourceEx strDBType, strDBSystem, strDatabase, strUser, strPassword
```

Parameter

strDBType (String)
Vordefiniertes Kennzeichen des Datenbanktyps.

Datenbanktyp	Beschreibung
"DEFAULT"	Netzdatenbank
"PROT"	Globale Schutzgerätedatenbank
"PROT_USR"	Lokale Schutzgerätedatenbank

strDBSystem (String)
Vordefiniertes Kennzeichen des Datenbanksystems.

Datenbanksystem	Beschreibung
"JET"	Microsoft Access
"ORACLE"	Oracle

strDatabase (String)
Vollständiger Pfad und Dateiname der Datenbank.

strUser (String)
Benutzername der Datenbank.

strPassword (String)
Passwort der Datenbank.

Anmerkungen

Diese Funktion ist veraltert und es wird empfohlen, sie nicht mehr zu verwenden. Sie steht nur mehr aus Kompatibilitätsgründen zur Verfügung und sollte durch die Funktion **Database – Voreinstellen der Datenbanken** ersetzt werden.

Beispiel

```
' Set database filename and path.
SimulateObj.DataSourceEx "DEFAULT", "JET", strDatabase, "Admin", ""
SimulateObj.DataSourceEx "PROT", "JET", strProtDatabase, "Admin", ""
```

Database – Voreinstellen der Datenbanken

Definiert die zur Berechnung verwendeten Datenbanken.

```
SimulateObj.Database strConnection
```

Parameter

strConnection (String)
Datenbankverbindung.

Kennzeichen	Beschreibung
"TYP"	Datenbanktyp
"MODE"	Datenbanksystem
"INSTANCE"	Datenbankserver
"NAME"	Datenbankname
"USR"	Benutzername
"PWD"	Passwort
"SYSUSR"	PSS SINCAL Administrationsbenutzer
"SYSPWD"	Passwort vom PSS SINCAL Administrationsbenutzer

"FILE"	Dateiname der Datenbank
"SINFILE"	Dateiname der PSS SINCAL Datei

Für das Kennzeichen "TYP" sind folgende Werte möglich.

Datenbanktyp	Beschreibung
"NET"	Netzdatenbank
"PROT"	Globale Schutzgerätedatenbank
"PROT_USR"	Lokale Schutzgerätedatenbank
"STD"	Globale Standarddatenbank
"STD_USR"	Lokale Standarddatenbank

Für das Kennzeichen "MODE" sind folgende Werte möglich.

Datenbanksystem	Beschreibung
"JET"	Microsoft Access
"ORACLE"	Oracle
"SQLSERVER"	SQL Server
"SQLEXPRESS"	SQL Server Express

Anmerkungen

Die Datenbankverbindung kann **paarweise** mit Feld=Wert und ";" getrennt bzw. in **Kurzform** in der vollständigen Reihenfolge angegeben werden.

Paarweise:

```
TYP=NET;MODE=JET;...
```

Kurzform:

```
TYP;MODE;FILE;INSTANCE;NAME;USR;PWD;SINFILE;SYSUSR;SYSPWD;
```

Beispiel

Abhängig vom Datenbanksystem sind unterschiedliche Verbindungskennzeichen anzugeben.

```
' Set database connection string depending on database system.
' ACCESS:
SimulateObj.Database "TYP=NET;MODE=JET;FILE=C:\Temp\Example
Ele_files\database.mdb;USR=Admin;PWD=;SINFILE=C:\Temp\Example Ele.sin;"

SimulateObj.Database "NET;JET;C:\Temp\Example Ele_files\database.mdb;;;Admin;;C:\Temp\Example
Ele.sin;;;"

' ORACLE:
SimulateObj.Database
"TYP=NET;MODE=ORACLE;USR=ORA_ELE1;PWD=ORA_ELE1;INSTANCE=ORA11;SINFILE=C:\Temp\Example
Ele.sin;SYSUSR=sincal;SYSPWD=sincal;"

SimulateObj.Database "NET;ORACLE;;ORA11;;ORA ELE1;ORA ELE1;C:\Temp\Example
Ele.sin;sincal;sincal;"

' SQLEXPRESS:
```

```
SimulateObj.Database "TYP=NET;MODE=SQLEXPRESS;FILE=C:\Temp\Example
Ele_files\database.mdf;NAME=Example Ele;SINFILE=C:\Temp\Example Ele.sin;"

SimulateObj.Database "NET;SQLEXPRESS;C:\Temp\Example Ele files\database.mdf;;Example Ele;;;
C:\Temp\Example Ele.sin;;;"

' SQLSERVER:
SimulateObj.Database
"TYP=NET;MODE=SQLSERVER;NAME=SQLSRV ELE1;INSTANCE=SQLSRV;USR=username;PWD=password;SINFILE=C:
\Temp\Example Ele.sin;SYSUSR=sincal;SYSPWD=sincal;"

SimulateObj.Database "NET;SQLSERVER;;SQLSRV;SQLSRV_ELE1;username;password;C:\Temp\Example
Ele.sin;sincal;sincal;"
```

SQLUser – Voreinstellen des SQL Benutzers

Legt den Benutzernamen und das Passwort für den SQL Server fest.

```
SimulateObj.SQLUser strUser, strPassword
```

Parameter

strUser (String)
SQL Benutzername.

strPassword (String)
SQL Passwort.

Beispiel

```
' Set the SQL user.
SimulateObj.SQLUser "User", "Password"
```

DataFile – Voreinstellen der Datendatei

Definiert die vom Import oder Export notwendige Datendatei.

```
SimulateObj.DataFile strDataFile
```

Parameter

strDataFile (String)
Vollständiger Pfad und Dateiname der Datendatei.

Beispiel

```
' Set the datafile for imports or exports.
SimulateObj.DataFile "C:\Test\CIM\CIMExample.xml"
```

MacroPath – Pfade für Modelle

Definiert den lokalen und globalen Pfad, von dem die Modelle verwendet werden.

```
SimulateObj.MacoPath strGlobalPath, strLocalPath
```

Parameter

strGlobalPath (String)
Vollständiger Pfad des Verzeichnisses, in dem die globalen Modelle gespeichert sind.

strLocalPath (String)
Vollständiger Pfad des Verzeichnisses, in dem die lokalen Modelle gespeichert sind.

Beispiel

```
' Set global and local path for models.
SimulateObj.MacroPath "C:\GlobalMacros", "C:\LocalMacros"
```

Language – Voreinstellen der Sprache

Legt die Sprache für die Berechnung und Berechnungsmeldungen fest.

```
SimulateObj.Language strLanguage
```

Parameter

strLanguage (String)
Vordefiniertes Kennzeichen der einzustellenden Sprache.

Kennzeichen	Beschreibung
"DE"	Deutsche Sprachausgabe
"US"	Englische Sprachausgabe

Beispiel

```
' Select language for messages.
SimulateObj.Language "DE"
```

Currency – Voreinstellen der Währung

Legt das Währungszeichen für die Berechnung und Berechnungsausgaben fest.

```
SimulateObj.Currency strCurrency
```

Parameter

strCurrency (String)
Einzustellendes Währungssymbol.

Beispiel

```
' Set currency.
SimulateObj.Currency "EUR"
SimulateObj.Currency "€"
```

SetInputState – Setzen des Eingabestatus

Setzt den Eingabestatus für die von der Berechnung zu berücksichtigenden Daten.

```
SimulateObj.SetInputState lInputMask
```

Parameter

lInputMask (Long Integer)
Bitweise Maske vordefinierter Kennzeichen der Eingabestati.

Kennzeichen	Beschreibung
Elektronetze	
0x00000001	Lastfluss
0x00000002	Kurzschluss
0x00000004	Oberschwingungen
0x00000008	Motoranlauf
0x00000010	Dimensionierung Niederspannung
0x00000020	Mehrfachfehler
0x00000040	Schutz
0x00000080	Distanzschutz
0x00000100	Optimierung
0x00000200	Dynamik
0x00000400	Unsymmetrischer Lastfluss
0x00000800	Zuverlässigkeit
0x00001000	Wirtschaftlichkeit
0x00002000	Lastermittlung
0x00004000	Arc Flash
Strömungsnetze	
0x00080000	Stationäre Berechnung
0x00100000	Dynamische Berechnung
0x00200000	Geostationäre Berechnung

Beispiel

```
const CalcMethod LF = &H00000001
const CalcMethod_SC = &H00000002

' Set input data mask.
SimulateObj.SetInputState CalcMethod_LF Or CalcMethod_SC
```

LoadDB – Laden der Eingabedaten aus der Datenbank

Lädt die Datenbank in den Hauptspeicher und erzeugt das Netzmodell für den Berechnungsprozess.

```
SimulateObj.LoadDB strMethod
```

Parameter

strMethod(String)
Vordefiniertes Kennzeichen der Berechnungsmethode. Eine vollständige Liste der zulässigen Werte ist bei der Funktion **Start – Starten der Berechnung** vorhanden.

Anmerkungen

Durch die Angabe der Berechnungsmethode werden nur jene Daten aus der Datenbank geladen, die für diese Berechnungsmethode notwendig sind.

Beispiel

```
' Load input data for load flow calculations from database.
SimulateObj.LoadDB "LF_NR"
```

SaveDB – Speichern der Ergebnisse in die Datenbank

Speichert die virtuellen Ergebnisse in die physikalische Datenbank, damit diese nach der Automatisierungslösung für weitere Auswertungen in der Datenbank verfügbar sind.

```
SimulateObj.SaveDB strMethod
```

Parameter

strMethod (String)
Vordefiniertes Kennzeichen der Berechnungsmethode. Eine vollständige Liste der zulässigen Werte ist bei der Funktion **Start – Starten der Berechnung** vorhanden.

Beispiel

```
' Save results to database.
SimulateObj.SaveDB "LF_NR"
```

AddObjID – Objekte hinzufügen

Definiert ein Objekt für die Berechnungsmethode zur weiteren speziellen Bearbeitung. Die Art der Bearbeitung ist abhängig von der jeweiligen Berechnungsmethode sowie dem übergebenen Steuerparameter.

```
SimulateObj.AddObjID( lRowType, lDBID, eMode )
```

Parameter

lRowType (Long Integer)
Datenbanktyp des Objektes.

lDBID (Long Integer)
Datenbank ID des Objektes.

eMode (Enum)
Vordefinierte Kennziffer für die Verwendung des Objektes in der Berechnung.

Kennzeichen	Kennziffer	Beschreibung
ADDOBJ_LF_NONE	0	
ADDOBJ_LF_MALF	1	Ausfallanalyse
ADDOBJ_OBJ_SC	2	Kurzschluss am Objekt
ADDOBJ_OBJ_EXP	3	Netomac Export – Ermittlung der Maschinen
ADDOBJ_OBJ_MOT_SIMPLE	4	Vereinfachter Motoranlauf
ADDOBJ_LF_ALLOC	5	Last anschließen
ADDOBJ_LF_RESUP	6	Wiederversorgung
ADDOBJ_FLOW_H2O_MALF	7	Ausfallanalyse Wasser
ADDOBJ_FLOW_GAS_MALF	8	Ausfallanalyse Gas
ADDOBJ_FLOW_HEAT_MALF	9	Ausfallanalyse Wärme/Kälte
ADDOBJ_OPT_CAP	10	Kondensatorplatzierung
ADDOBJ_GEN_PV	11	PV Kurven
ADDOBJ_FLOW_H2O_LEAK	12	Löschwasser Wasser
ADDOBJ_LF_MALF_RECON	13	Ausfallanalyse – Wiederanschluss
ADDOBJ_OPT_NET	14	Optimale Netzstruktur
ADDOBJ_ECO	15	Wirtschaftlichkeit
ADDOBJ_NETRED_INCLUDE	16	Wird derzeit nicht verwendet
ADDOBJ_NETRED_EXCLUDE	17	Elemente, die nicht reduziert werden sollen
ADDOBJ_NETRED_BOUNDARY	18	Grenzleitungen für Netzreduktion

Beispiel

```
' Set object for further usage.
const ADDOBJ OBJ SC = 2
Simulation.AddObjID( 4, 1, ADDOBJ_OBJ_SC );
```

Parameter – Setzen und Abfragen von globalen Parametern

Setzt einen globalen Parameter für die Berechnungsmethode.

```
SimulateObj.Parameter( strParameter ) = Value
Value = SimulateObj.Parameter( strParameter )
```

Eigenschaften

Parameter (Variant)
Wert des Parameters.

Parameter

strParameter (String)
Vordefinierter Name des Parameters. Je nach Berechnungsmethode stehen unterschiedliche Parameter zur Verfügung.

Globale Parameter

Parameter	Datentyp	Beschreibung
"Sim.Identification"	String	Namensidentifikation der Objekte "Name" = Identifikation anhand des Namens "ShortName" = Identifikation anhand des Kurznamens Legt fest, ob die Identifikation der Objekte anhand des Namen ("Name") oder mit dem Kurznamen ("ShortName") erfolgt.
"GRAPHIC_VIEWID"	Integer	Die von der Berechnung zu verwendende Grafikansicht (GraphicAreaTile_ID)

Ausfallanalyse

Parameter	Datentyp	Beschreibung
"CA_MODE"	String	Gibt die Berechnungsart an "NORMAL" = Komplette Berechnung "REDUCED" = Reduzierte Berechnung "PRE_ANALYSE" = Voranalyse
"CA_PRE_ANALYSE_MODE"	String	Gibt die Bewertungsmethode für die Voranalyse an "VOLT" = Spannungsänderung "ISOL_POWER" = Nicht gelieferte Leistung "ISOL_ELEMENTS" = Unversorgte Elemente "ISOL_CONSUMERS" = Unversorgte Verbraucher
"CA_PRE_ANALYSE_COUNT"	Integer	Anzahl der zu berechnenden Ausfälle

Wiederversorgung

Parameter	Datentyp	Beschreibung
"LF_RESUP_MODE"	Integer	Gibt den Modus an 0 = Standard 1 = Abgangsbasierend
"LF_RESUP_RESUPPLYCNT_ACT"	Integer	Aktivierung der maximalen Anzahl der Wiederversorgungen 0 = Nicht aktiviert 1 = Aktiviert
"LF_RESUP_RESUPPLYCNT"	Integer	Anzahl der Wiederversorgungen
"LF_RESUP_SWITCHCNT_ACT"	Integer	Aktivierung der maximalen Anzahl der Schalthandlungen 0 = Nicht aktiviert 1 = Aktiviert
"LF_RESUP_SWITCHCNT"	Integer	Anzahl der Schalthandlungen
"LF_RESUP_LOADSHEDDING_ACT"	Integer	Aktivierung von Lastabwurf 0 = Nicht aktiviert 1 = Aktiviert
"LF_RESUP_VIOLATION_ACT"	Integer	Aktivierung von Grenzwertverletzungen

		0 = Nicht aktiviert 1 = Aktiviert
"LF_RESUP_SWITCHCNT_FACTOR"	Double	Gewichtung für Schalthandlungen
"LF_RESUP_LOADSHEDDING_FACTOR"	Double	Gewichtung für Lastabwurf
"LF_RESUP_VIOLATION_FACTOR"	Double	Gewichtung für Grenzwertverletzung

VoltVar

Parameter	Datentyp	Beschreibung
"OPT_VOLTVAR_CAP_SN"	Double	Nennscheinleistung für den Kondensator
"OPT_VOLTVAR_TRAFO"	Integer	Aktivierung des Wandlers 0 = Nicht aktiviert 1 = Aktiviert
"OPT_VOLTVAR_TRAFO_SN"	Double	Nennscheinleistung für den Transformator
"OPT_VOLTVAR_TRAFO_UK"	Double	Bezogene Kurzschlussspannung für den Transformator
"OPT_VOLTVAR_LIMIT_LOWER"	Double	Untergrenze Spannung in %
"OPT_VOLTVAR_LIMIT_UPPER"	Double	Obergrenze Spannung in %
"OPT_VOLTVAR_MINMAX_MODE"	Integer	Gibt den Berechnungsmodus an 0 = Faktor 1 = Arbeitspunkt
"OPT_VOLTVAR_MIN_FACTOR"	Double	Faktor für Minimum
"OPT_VOLTVAR_MAX_FACTOR"	Double	Faktor für Maximum
"OPT_VOLTVAR_MIN_OPID"	Integer	Arbeitspunkt für Minimum
"OPT_VOLTVAR_MAX_OPID"	Integer	Arbeitspunkt für Maximum

Statische Netzreduktion

Parameter	Datentyp	Beschreibung
"STATNETRED_USESOURCEDB"	Integer	Definiert, ob die Originaldatenbank geändert werden soll oder aber ob das reduzierte Netz in eine zweite Datenbank geschrieben wird. 0 = Zweite Datenbank mit reduziertem Netz befüllen 1 = Änderung der Originaldatenbank durchführen
"STATNETRED_SINFILE"	String	Kompletter Dateiname der SIN Datei der zweiten Datenbank. z.B.: "D:\Network\Red-RS.sin"
"STATNETRED_DATABASE"	String	Datenbankdefinition für die zweite Datenbank. z.B.: "TYP=NET;MODE=JET;FILE=D:\Network\Red-RS_files\database.mdb;USR=Admin;SINFILE=D:\Network\Red-RS.sin;"
"STATNETRED_CREATEGRAPHIC"	Integer	Aktiviert die grafische Generierung der Randknoten bei Verwendung von zwei getrennten Datenbanken. 0 = Keine Grafik erzeugen 1 = Grafik erzeugen
"STATNETRED_EXTWARD"	Integer	Aktiviert die Bestimmung von Ersatzeinspeisungen mit dem Extended Ward Verfahren. 0 = Keine Extended Wards erzeugen 1 = Extended Wards erzeugen
"STATNETRED_SC1"	Integer	Aktiviert die Bestimmung von Nullsystemdaten im reduzierten Netz für unsymmetrische Kurzschlussberechnungen. 0 = Keine Nullsystemdaten bestimmen 1 = Nullsystemdaten bestimmen
"STATNETRED_SC3"	Integer	Aktiviert die Bestimmung von Kurzschlussdaten im reduzierten Netz. 0 = Keine Kurzschlussdaten bestimmen 1 = Kurzschlussdaten bestimmen

Dynamische Netzreduktion

Parameter	Datentyp	Beschreibung
"DYNNETRED_USESOURCEDB"	Integer	Definiert, ob die Originaldatenbank geändert werden soll oder aber ob das reduzierte Netz in eine zweite Datenbank geschrieben wird. 0 = Zweite Datenbank mit reduziertem Netz befüllen 1 = Änderung der Originaldatenbank durchführen
"DYNNETRED_SINFILE"	String	Kompletter Dateiname der SIN Datei der zweiten Datenbank. z.B.: "D:\Network\Red-RS.sin"
"DYNNETRED_DATABASE"	String	Datenbankdefinition für die zweite Datenbank. z.B.: "TYP=NET;MODE=JET;FILE=D:\Network\Red-RS_files\database.mdb;USR=Admin;SINFILE=D:\Network\Red-RS.sin;"
"DYNNETRED_CREATEGRAPHIC"	Integer	Aktiviert die grafische Generierung der Randknoten bei Verwendung von zwei getrennten Datenbanken: 0 = Keine Grafik erzeugen 1 = Grafik erzeugen
"DYNNETRED_TIMESTART"	Double	Startzeit in Sekunden für die Korrelationsfunktionen
"DYNNETRED_TIMEEND"	Double	Endzeit in Sekunden für die Korrelationsfunktionen
"DYNNETRED_LOWERLIMIT"	Double	Unterer Grenzwert für den Korrelationsfaktor
"DYNNETRED_MACHINES"	Integer	Anzahl der kohärenten Maschinen, die im reduzierten Netz generiert werden sollen. Bei Angabe von "0" wird die Anzahl automatisch bestimmt.
"DYNNETRED_FUNCTION"	Integer	Bestimmt, welches Signal für die Korrelationsfunktionen verwendet wird. 1 = Schlupf 2 = Polradwinkel 3 = Wirkleistung 4 = Blindleistung 5 = Spannung Sinnvollerweise sollte hier immer Schlupf gewählt werden, da damit die besten Ergebnisse erreicht werden.
"DYNNETRED_REFNODE"	Integer	KnotenID des Referenzknotens im zu reduzierenden Netzteil
"DYNNETRED_REFVOLTAGE"	Double	Bezugsspannung für Netzäquivalent in kV
"DYNNETRED_MAXPOWER"	Double	Max. Leistung über Ersatzleitung in MW
"DYNNETRED_POWERIGNORE"	Double	Leistung von zu ignorierende Maschinen in MW
"DYNNETRED_PREFIX"	String	Beliebiger Namenszusatz für reduzierte Elemente
"DYNNETRED_NODEMODELNET"	Integer	Interne Darstellung der Knoten in der Netzreduktion 1 = PQ Typ 2 = I Typ 5 = PQ Typ (neg. Maschinen) 6 = I Typ (neg. Maschinen)
"DYNNETRED_NODEMODELMACHINES"	Integer	Interne Darstellung der Maschinen in der Netzreduktion 1 = PQ Typ 2 = I Typ 3 = PV Typ
"DYNNETRED_NODEMODELCOUPLING"	Integer	Interne Darstellung der Kuppelknoten in der Netzreduktion 1 = PQ Typ 2 = I Typ 3 = PV Typ 4 = S Typ
"DYNNETRED_KEEPNAMES"	Integer	Namen von Einzelmaschinen erhalten 0 = Nein 1 = Ja
"DYNNETRED_REDCONTROLLER"	Integer	Maschinen im zu reduzierenden Netz ohne Regler 0 = Nein 1 = Ja

"DYNNETRED_NOTREDCONTROLLER"	Integer	Maschinen im nicht zu reduzierenden Netz ohne Regler 0 = Nein 1 = Ja
"DYNNETRED_KEEPCONTROLLER"	Integer	Regler von Einzelmaschinen erhalten 0 = Nein 1 = Ja
"DYNNETRED_PSSCONTROLLER"	String	Name des PSS Reglers
"DYNNETRED_EXCITER"	String	Name des Spannungsreglers
"DYNNETRED_GOVERNOR"	String	Name des Turbinenreglers

CIM Export

Parameter	Datentyp	Beschreibung
"CIM_FORMAT"	String	CIM-Version "CIM_V10" "CIM_V11" "CIM_V12" "CIM_V14" "CIM_V15"
"CIM_PROFILE"	String	CIM-Profil "CIM_STANDARD" = CIM Standard "CIM_PLANNING" = CIM for Planning "CIM_ENTSOE" = CIM for ENTSO-E
"FILENAME" "FILENAME_###"	String	Vollständiger Pfad und Dateiname der ersten Datei sowie der Folgedateien, ### steht für die Dateinummer beginnend bei 2 bis maximal der unter FILENAME_CNT angegebenen Anzahl.
"FILENAME_FLAG" "FILENAME_FLAG_###"	String	Dateiart der ersten Datendatei sowie der Folgedateien, ### steht für die Dateinummer beginnend bei 2 bis maximal der unter FILENAME_CNT angegebenen Anzahl. "DATA" = CIM Eingabedatei "BOUNDARY" = CIM Boundary Datei "CONFIG" = CIM Konfigurationsdatei
"CIM_NAME"	Integer	Legt fest, welches CIM Attribut als Name verwendet wird. Gültige Kennzahlen sind: 0 = None 1 = cim:IdentifiedObject.Name 2 = cim:IdentifiedObject.AliasName 3 = cim:IdentifiedObject.Description
"CIM_SHORTNAME"	Integer	Legt fest, welches CIM Attribut als Kurzname verwendet wird. Gültige Kennzahlen sind: 0 = None 1 = cim:IdentifiedObject.Name 2 = cim:IdentifiedObject.AliasName 3 = cim:IdentifiedObject.Description
"CIM_SPLITFILES"	Integer	Export in mehrere XML Dateien 0 = Export in mehrere Dateien 1 = Export in eine Datei
"CIM_CREATEZIP"	Integer	ZIP Archiv erzeugen 0 = Nein 1 = Ja
"CIM_MRID"	String	Art der verwendeten ID "SINCALID" = PSS SINCAL generierte ID "UUID" = Universal Unique ID "GUID" = Global Unique ID
"CIM_LFRESULTS"	Integer	Lastflussergebnisse exportieren 0 = keine Ergebnisse exportieren 1 = Lastflussergebnisse exportieren

"EXPORT_GRAPHIC"	Integer, String	Grafik exportieren 0, "NONE" = kein Grafikexport 1, "AUTOMATIC" = automatischer Grafikexport 2, "SINCAL" = vereinfachter Grafikexport 3, "EXTENDED" = erweiterter Grafikexport

CIM Import

Parameter	Datentyp	Beschreibung
"FILENAME_CNT"	Integer	Anzahl der zu importierenden Dateinamen
"FILENAME" "FILENAME_###"	String	Vollständiger Pfad und Dateiname der ersten Datei sowie der Folgedateien, ### steht für die Dateinummer beginnend bei 2 bis maximal der unter FILENAME_CNT angegebenen Anzahl.
"FILENAME_FLAG" "FILENAME_FLAG_###"	String	Dateiart der ersten Datendatei sowie der Folgedateien, ### steht für die Dateinummer beginnend bei 2 bis maximal der unter FILENAME_CNT angegebenen Anzahl. "DATA" = CIM Eingabedatei "BOUNDARY" = CIM Boundary Datei "CONFIG" = CIM Konfigurationsdatei
"CIM_FORMAT"	String	CIM-Version "CIM_V10" "CIM_V11" "CIM_V12" "CIM_V14" "CIM_V15"
"CIM_PROFILE"	String	CIM-Profil "CIM_STANDARD" = CIM Standard "CIM_PLANNING" = CIM for Planning "CIM_ENTSOE" = CIM for ENTSO-E
"BASE_FREQUENCY"	Double	Basis-Frequenz
"LENGTH_FACTOR"	Double	Umrechnungsfaktor für Längenangaben
"CIM_NAME"	Integer	Legt fest, welches CIM Attribut als Name verwendet wird. Gültige Kennzahlen sind: 0 = None 1 = cim:IdentifiedObject.Name 2 = cim:IdentifiedObject.AliasName 3 = cim:IdentifiedObject.Description
"CIM_SHORTNAME"	Integer	Legt fest, welches CIM Attribut als Kurzname verwendet wird. Gültige Kennzahlen sind: 0 = None 1 = cim:IdentifiedObject.Name 2 = cim:IdentifiedObject.AliasName 3 = cim:IdentifiedObject.Description
"IMPORT_GRAPHIC"	Integer	Grafik importieren 0 = Nein 1 = Ja
"GRAPHIC_MODE"	Integer	Grafikmodus 0 = Schematisch 1 = Geografisch
"GRAPHIC_INDIVIDUAL_TEXT"	Integer	Individueller Text für Netzelemente und Knoten 1 = Individueller Text 0 = Kein individueller Text
"GRAPHIC_SCALE_FACTOR"	Double	Skalierungsfaktor der Grafik
"GRAPHIC_SYMBOLSIZE"	Integer	Symbolgröße der Netzelemente
"GRAPHIC_OFFSETX"	Double	X-Offset der Grafik
"GRAPHIC_OFFSETY"	Double	Y-Offset der Grafik

PSS E Export

Parameter	Datentyp	Beschreibung
"EXPORT_NAME"	Integer	Export des Namens oder Kurznamens 0 = Name 1 = Kurzname
"EXPORT_NAME_KEY"	Integer	Kurznamen der Knoten als "BUS Number" verwenden 0 = Ja 1 = Nein
"PSSE_VERSION"	Integer	Versionsnummer Zulässig sind: 32 und 33

PSS E Import

Parameter	Datentyp	Beschreibung
"FILENAME_CNT"	Integer	Anzahl der Datendateien
"FILENAME" "FILENAME_###"	String	Vollständiger Pfad und Dateiname für die erste Datendatei sowie für alle Folgedateien. ### kennzeichnet die Nummer der Datei, beginnend bei 2 bis max. der unter FILENAME_CNT angegebenen Anzahl.
"PSSE_VERSION"	Integer	Versionsnummer Zulässig sind: 29, 30, 31, 32, 33 und 0 (Auto)
"PSSE_SEQ_FILENAME"	String	Vollständiger Pfad und Dateiname der Sequencedatei
"PSSE_MODE"	Integer	Importmodus 0 = Standardmodus 1 = Erweiterter Import
"BASE_FREQUENCY"	Double	Basisfrequenz
"LENGTH_FACTOR"	Double	Skalierungsfaktor für Längen
"ZERO_IMPEDANCE"	Integer	Import von impedanzlosen Leitungen 0 = Nein 1 = Ja
"ZERO_IMPEDANCE_MIN_VALUE"	Double	Minimalimpedanz, ab der Leitungen als impedanzlose Verbindungen betrachtet werden
"PSSE_REFVOLTAGE"	Double	Bezugsnennspannung (für Knoten und Elemente mit 0,0 kV)
"GRAPHIC_FILENAME_CNT"	Integer	Anzahl der Grafikdateien
"GRAPHIC_FILENAME" "GRAPHIC_FILENAME_###"	String	Vollständiger Pfad und Dateiname für die erste Grafikdatei sowie für alle Folgedateien. ### kennzeichnet die Nummer der Datei beginnend bei 2 bis max. der unter GRAPHIC_FILENAME_CNT angegebenen Anzahl.

UCTE Export

Parameter	Datentyp	Beschreibung
"EXPORT_NAME"	Integer	Export des Namens oder Kurznamens 0 = Name 1 = Kurzname
"EXPORT_NAME_KEY"	Integer	Kurznamen der Knoten als "BUS Number" verwenden 0 = Ja 1 = Nein

DVG Export

Parameter	Datentyp	Beschreibung
"EXPORT_NAME"	Integer	Export des Namens oder Kurznamens 0 = Name

		1 = Kurzname

DVG Import

Parameter	Datentyp	Beschreibung
"FILENAME"	String	Vollständiger Pfad und Dateiname der Datendatei
"DVG_IMPORT_MODE"	Integer	Importmodus (Verhalten bei Problemen und Fehler) 0 = Strikter Modus, Abbruch bei Fehlern 1 = Fehlertoleranter Import, Probleme und Fehler werden protokolliert
"GRAPHIC_FILENAME_CNT"	Integer	Anzahl der Grafikdateien
"GRAPHIC_FILENAME" "GRAPHIC_FILENAME_###"	String	Vollständiger Pfad und Dateiname für die erste Grafikdatei sowie für alle Folgedateien. ### kennzeichnet die Nummer der Datei beginnend bei 2 bis max. der unter GRAPHIC_FILENAME_CNT angegebenen Anzahl.

Beispiel

```
' Set ShortName as default for object access.
SimulateObj.Parameter( "Sim.Identification" ) = "ShortName"
' Set Name as default for object access.
SimulateObj.Parameter( "Sim.Identification" ) = "Name"
```

DoCommand – Anweisungen ausführen

Führt eine vordefinierte Anweisung in der Berechnung durch.

```
SimulateObj.DoCommand strCommand, vtParameter1, vtParameter2
```

Parameter

strCommand (String)
Vordefinierte Kennzeichen der auszuführenden Anweisung.

Anweisung	Beschreibung
"CHANGEVARIANT"	Wechseln der Variante
"DELETERESULTS"	Löschen aller Ergebnisse in der Datenbank

vtParameter1, vtParameter2 (Variant)
Zusätzliche Parameter, abhängig von der Anweisung.

"CHANGEVARIANT" – Wechseln der Variante

Parameter	Datentyp	Beschreibung
vtParameter1	String oder Long Integer	Name der Variante bzw. DB-ID der Variante
vtParameter2	Boolean	Variante der Include-Netze wechseln

"DELETERESULTS" – Löschen aller Ergebnisse in der Datenbank

Parameter	Datentyp	Beschreibung
vtParameter1	String	Kennzeichen des Netztyps ("EL" oder "FLOW")

vtParameter2		wird nicht verwendet

Beispiel

```
' Change variant to Base.
SimulateObj.DoCommand "CHANGEVARIANT", "Base", False
```

Start – Starten der Berechnung

Startet die Berechnung.

```
SimulateObj.Start strMethod
```

Parameter

strMethod (String)
Vordefiniertes Kennzeichen der Berechnungsmethode.
Die folgende Tabelle zeigt die zulässigen Parameter zur Spezifikation der Berechnungsmethode.

Ber. Methode	Beschreibung
Elektronetze	
LF	Lastfluss lt. Einstellung in den Berechnungsparametern
LF_NR	Lastfluss Newton Raphson
LF_YMAT	Lastfluss Admittanzmatrix
LF_CI	Lastfluss Stromiteration
LF_USYM	Unsymmetrischer Lastfluss
LF_NETO	Lastfluss Netomac
LF_PSSE	Lastfluss PSS E
LF_MALF	Ausfall von ausgewählten Netzelementen
LF_TRIM	Lastermittlung
LF_ALLOC	Last anschließen
LF_BAL	Lastsymmetrierung
LF_RESUP	Wiederversorgung
LF_TAP	Tap-Zone Ermittlung
LF_INC	Lastentwicklung
LC	Lastprofil
GEN_PV	PV Kurven
OPT_LF	Optimierung Lastfluss
OPT_BR	Optimierung Trennstellen
OPT_COMP	Kompensationsleistung
OPT_CAP	Kondensatorplatzierung
OPT_NET	Optimale Netzstruktur
COND	Ausfallanalyse
SC1	1-poliger Erdschluss
SC2	2-poliger Kurzschluss
SC3	3-poliger Kurzschluss
SC1 [NodeID]	1-poliger Erdschluss am Knoten
SC2 [NodeID]	2-poliger Kurzschluss am Knoten

SC3 [NodeID]	3-poliger Kurzschluss am Knoten
IC[x][R][E] [NodeID]	Individueller Kurzschluss (Anzahl Phasen, Rückleiterschluss, Erdschluss) am Knoten
GC2	2-poliger Erdschluss
MF	Mehrfachfehler
DIM	Sicherungsüberprüfung
HAR	Oberschwingungen
RC	Rundsteuerung
ECO_SUM	Wirtschaftlichkeit
MOT	Motoranlauf
MOT_SIMPLE	Vereinfachter Motoranlauf
NETO_STAB	Stabilität
NETO_TSTAB	Transiente Stabilität
NETO_EW	Eigenwerte
REL	Zuverlässigkeit
REL_EVAL	Auswertung Zuverlässigkeit
PROT SC1 [FaultID]	Schutzkoordination – 1-poliger Erdschluss
PROT SC2 [FaultID]	Schutzkoordination – 2-poliger Kurzschluss
PROT GC2 [FaultID]	Schutzkoordination – 2-poliger Erdschluss
PROT SC3 [FaultID]	Schutzkoordination – 3-poliger Kurzschluss
PROT IC[x][R][E] [FaultID]	Schutzkoordination – individueller Kurzschluss (Anzahl Phasen, Rückleiterschluss, Erdschluss)
PROT MF	Mehrfachfehler Fehlerpaket
PROT_DET	Fehlerortung
PROT_SET	Ermittlung Einstellwerte Distanzschutz
PROT_SET_CHART	Einstellwertdiagramme Distanzschutz
PROT_ROUTE SC1	Schutzstrecken – 1-poliger Erdschluss
PROT_ROUTE SC2	Schutzstrecken – 2-poliger Erdschluss
PROT_ROUTE GC2	Schutzstrecken – 2-poliger Kurzschluss
PROT_ROUTE SC3	Schutzstrecken – 3-poliger Kurzschluss
PROT_ROUTE IC[x][R][E]	Schutzstrecken – individueller Kurzschluss (Anzahl Phasen, Rückleiterschluss, Erdschluss)
ARCFL	Arc Flash
ZUBER	Zuverlässigkeit
ZUBER_EVAL	Auswertung Zuverlässigkeit
Wassernetze	
FLOW_H2O	Stationär
FLOW_H2O_TM	Zeitreihe
FLOW_H2O_OP	Arbeitspunktreihe
FLOW_H2O_COND	Ausfallanalyse
FLOW_H2O_MALF	Ausfallanalyse (ausgewählte Netzelemente)
FLOW_H2O_FWP	Löschwasserdruck
FLOW_H2O_FWQ	Löschwassermenge
FLOW_H2O_LEAKP	Löschwasserdruck (ausgewählte Netzelemente)
FLOW_H2O_LEAKQ	Löschwassermenge (ausgewählte Netzelemente)
Gasnetze	
FLOW_GAS	Stationär
FLOW_GAS_TM	Zeitreihe
FLOW_GAS_OP	Arbeitspunktreihe
FLOW_GAS_COND	Ausfallanalyse
FLOW_GAS_MALF	Ausfallanalyse (ausgewählte Netzelemente)

Wärme-/Kältenetze	
FLOW_HEAT	Stationär
FLOW_ HEAT _TM	Zeitreihe
FLOW_ HEAT _OP	Arbeitspunktreihe
FLOW_HEAT_COND	Ausfallanalyse
FLOW_HEAT_MALF	Ausfallanalyse (ausgewählte Netzelemente)

Die folgende Tabelle zeigt die zulässigen Parameter zur Spezifikation der Import- und Exportfunktionen.

Import/Export	Beschreibung
Elektronetze	
CIM_IMP	CIM Import
CIM_EXP	CIM Export
PSSE_IMP	PSS E Import
PSSE_EXP	PSS E Export
UCTE_IMP	UCTE Import
UCTE_EXP	UCTE Export
DVG_IMP	DVG Import
DVG_EXP	DVG Export

Beispiel

```
' Start load flow simulation.
SimulateObj.Start "LF_NR"
If SimulateObj.StatusID = 1101 Then
    WScript.Echo "Simulation finished without errors!"
Else
    WScript.Echo "Error: Load flow failed!"
    Exit Do
End If
```

StatusID – Statuscode des Berechnungsvorganges

Fragt den Status des Berechnungsvorganges ab.

```
lStatus = SimulateObj.StatusID
```

Eigenschaften

StatusID (Long Integer)
Statuscode des Berechnungsvorganges. Der Wert "1101" kennzeichnet, dass der Berechnungsvorgang ohne Fehler durchgeführt wurde.

Beispiel

```
' Check if the simulation was finished without any errors.
If Not ( SimulateObj.StatusID = 1101 ) Then
    WScript.Echo "Error: Simulation failed!"
```

```
End If
```

GetObj – Zugriff auf Berechnungsobjekte

Liefert eine Instanz eines Berechnungsobjektes, die den direkten Zugriff auf die "internen" in der Berechnung aufgebauten Objekte ermöglicht.

```
Set LoadObj = SimulateObj.GetObj( strObjectType, strName )
Set LoadObj = SimulateObj.GetObj( strObjectType, lDBID )
```

Parameter

strObjectType (String)
Objekttyp des Netzelementes. Dieser entspricht dem Namen der Datenbanktabelle, in der die entsprechenden Elementdaten gespeichert werden.

lDBID (String)
Datenbank-ID des Netzelementes.

strName (String)
Name des Netzelementes. Die Identifikation über den Namen oder den Kurznamen kann mit der Parameteranweisung global voreingestellt werden.

Rückgabewert

Object (Object)
Automatisierungsobjekt eines in der Berechnung aufgebauten Netzelementes.

Beispiel

```
' Get SimulationObject of type "LOAD" with name "LO8".
Dim LoadObj
Set LoadObj = SimulateObj.GetObj( "LOAD", CStr( "LO8" ) )
If LoadObj is Nothing Then
    WScript.Echo "Error: Load not found!"
    WScript.Quit
End If
```

GetObjById, GetObjByGUID – Zugriff auf Berechnungsobjekte über ID

Liefert eine Instanz auf ein Berechnungsobjekt, die den direkten Zugriff auf die "internen" in der Berechnung aufgebauten Objekte ermöglicht.

```
Set SimObj = SimulateObj.GetObjByID( lID )
Set SimObj = SimulateObj.GetObjByGUID( strGUID )
```

Parameter

lID (Long Integer)
Interne Nummer des Berechnungsobjektes. Diese "interne ID" ist in den Topologiedaten der

Berechnungsobjekte verfügbar.

strGUID (String)
Master Ressource des Berechnungsobjektes.

Rückgabewert

SimObj (Object)
Automatisierungsobjekt eines in der Berechnung aufgebauten Netzelementes.

Beispiel

```
' Get SimulationObject with internal ID 8.
Dim SimObj
Set SimObj = SimulateObj.GetObjByID( 8 )
If SimObj Is Nothing Then
    WScript.Echo "Error: Object not found!"
    WScript.Quit
End If

' Get SimulationObject with Master Resource ID.
Dim SimObj
Set SimObj = SimulateObj.GetObjByGUID( "289DAFC1-8541-4abb-AE9F-1C47E6A2D32B" )
If SimObj Is Nothing Then
    WScript.Echo "Error: Object not found!"
    WScript.Quit
End If
```

DB_EL, DB_FLOW – Zugriff auf die Datenbankobjekte

Ermöglicht den Zugriff auf die Eingabedaten und Ergebnisse der Simulation.

```
Set SimulateDatabase = SimulateObj.DB_EL
Set SimulateDatabase = SimulateObj.DB_FLOW
```

Eigenschaften

DB_EL (Object)
Datenbankobjekt für Elektronetze.

DB_FLOW (Object)
Datenbankobjekt für Strömungsnetze.

Anmerkungen

Diese Eigenschaft kann nur gelesen werden.

Beispiel

```
' Get the database object.
Dim SimulateDatabase
Set SimulateDatabase = SimulateObj.DB_EL
If SimulateDatabase Is Nothing Then
    WScript.Echo "Error: Getting database object failed!"
    WScript.Quit
```

```
End If
' ...
Set SimulateDatabase = Nothing
```

Messages – Zugriff auf die Meldungsobjekte

Ermöglicht den Zugriff auf die von der Berechnung generierten Meldungen.

```
Set objMessages = SimulateObj.Messages
```

Eigenschaften

Messages (Object)
Automatisierungsobjekt für die Berechnungsmeldungen.

Anmerkungen

Diese Eigenschaft kann nur gelesen werden.

Beispiel

```
' Get message object.
Dim objMessages
Set objMessages = SimulateObj.Messages

' Release message object.
Set objMessages = Nothing
```

4.1.2 Berechnungsobjekt

Das Berechnungsobjekt ermöglicht einen direkten Zugriff auf die "internen" in der Berechnung aufgebauten Objekte, die das Netzmodell beschreiben. Dieses Objekt stellt also die Abbildung eines Netzelementes im Hauptspeicher der Berechnung dar.

Das Berechnungsobjekt ermöglicht die direkte Manipulation von Eingabedaten in der Berechnung. So können damit beispielsweise bei einer Last die Wirk- und Blindleistungswerte beliebig abgeändert werden, ohne dass dabei Daten von der Datenbank geladen werden müssen. Darüber hinaus stellt das Berechnungsobjekt Funktionen zur Verfügung, mit denen sehr einfach auf die Ergebnisse des Netzelementes zugegriffen werden kann.

Eine Auflistung der verfügbaren Attribute, die mit dem Berechnungsobjekt angesprochen werden können, ist im Kapitel **Attribute der Berechnungsobjekte** verfügbar.

Eine Instanz eines Berechnungsobjektes wird über das Simulationsobjekt mit der Funktion **GetObj** erzeugt.

Beispiel

```
' Get simulation object of type "Load" with name "LO8".
Dim LoadObj
Set LoadObj = SimulateObj.GetObj( "LOAD", "LO8" )
```

```
If LoadObj Is Nothing Then
    WScript.Echo "Error: Load not found!"
    WScript.Quit
End If

' Release the simulation object.
Set LoadObj = Nothing
```

Count – Anzahl der mögliche Attribute

Liefert die Anzahl der möglichen Attribute für das jeweilige Objekt zurück.

```
lCnt = LoadObj.Count()
```

Rückgabewert

lCnt (Long Integer)
Anzahl der möglichen Attribute.

Beispiel

```
' Get the number of available attributes.
Dim Cnt
Cnt = LoadObj.Count()
```

Name – Attributnamen ermitteln

Liefert den Namen eines Attributes zurück.

```
strName = SimObj.Name( iAttribute )
```

Eigenschaften

Name (String)
Name des Attributes.

Parameter

iAttribute (Long Integer)
Nummer des Attributes.

Beispiel

```
' Display the names of all available attributes for the object.
Dim iAttr, lCnt
lCnt = SimObj.Count()
For iAttr = 1 To lCnt
    Dim strName
    strName = SimObjObj.Name( iAttr )
    WScript.Echo iAttr & ": " & strName
Next
```

Item – Zugriff auf Attribute

Stellt den Zugriff auf die verschiedenen Eingabe- und Ergebnisdaten eines Berechnungsobjektes zur Verfügung.

```
Value = SimObj.Item( lAttribute )
Value = SimObj.Item( strAttribute )

SimObj.Item( lAttribute ) = Value
SimObj.Item( strAttribute ) = Value
```

Eigenschaften

Item (Variant)
Wert des Attributes.

Parameter

iAttribute (Long Integer)
Numerischer Index des Attributes.

strAttribute (Long Integer)
Name des Attributes.

Anmerkungen

Die verfügbaren Attribute sind abhängig vom jeweiligen Objekt. Alle Objekte besitzen identische Topologieattribute. Mit diesen Topologieattributen können die Knoten und Netzelemente eindeutig indenitifiziert werden und auch das Ein- bzw. Ausschalten ist damit möglich.

Eine detaillierte Darstellung aller Objekte und deren verfügbaren Attribute finden Sie im Kapitel **Berechnungsobjekte und deren Attribute**.

Beispiel

```
' Get P from the object and set a new value for this attribute.
Dim Val
Val = SimObj.Item( "P" )
SimObj.Item( "P" ) = P * 2

' Get the value with the index 3 and assign a new value.
Dim Val
Val = SimObj.Item( 3 )
SimObj.Item( 3 ) = Val * 2
```

Result – Zugriff auf Berechnungsergebnisobjekt

Ermöglicht den Zugriff auf ein Ergebnisobjekt eines Berechnungsobjektes.

```
Set ResultObj = SimObj.Result( strResult, iTerminalNo )
```

Parameter

strResult (String)
SQL Name der gewünschten Ergebnistabelle.

iTerminalNo (Integer)
Anschlussnummer des gewünschten Ergebnisses.

Rückgabewert

ResultObj (Object)
Automatisierungsobjekt eines Ergebnisses.

Anmerkungen

Die Netzelementergebnisse werden pro Anschluss bereitgestellt. Knotenelemente (z.B. Generatoren, Asynchronmaschinen, Lasten) haben einen Anschluss und Zweigelemente (z.B. Leitungen, Transformatoren, Längsdrosseln) haben zwei Anschlüsse.

Beispiel

```
' Get load flow results on the first terminal of object.
Dim ResultObj
Set ResultObj = SimObj.Result( "LFBRANCHRESULT", 1 )

If ResultObj Is Nothing Then
    WScript.Echo "Error: No result available!"
End If
```

4.1.3 Berechnungsergebnisobjekt

Berechnungsergebnisobjekte sind virtuelle Objekte, die einen sehr komfortablen Zugriff auf die individuellen Ergebnisse für einzelne Netzelemente ermöglichen.

Eine Instanz eines Berechnungsergebnisobjekts wird über das Berechnungsobjekt mit der Funktion **Result** erzeugt.

Beispiel

```
' Get the load flow result object for a load.
Dim LFBranchResultLoad
Set LFBranchResultLoad = LoadObj.Result( "LFBRANCHRESULT", 1 )

If LFBranchResultLoad Is Nothing Then
    WScript.Echo "Error: Cant get result object!"
End If
```

Zu beachten ist, dass das Berechnungsergebnisobjekt nach Verwendung ordnungsgemäß freigegeben werden muss.

```
' Relase the result object.
Set LFBranchResultLoad = Nothing
```

Count – Anzahl der möglichen Attribute

Liefert die Anzahl der möglichen Attribute für das jeweilige Ergebnisobjekt zurück.

```
lCnt = ResultObj.Count()
```

Rückgabewert

lCnt (Long Integer)
Anzahl der möglichen Attribute.

Beispiel

```
' Get the number of available attributes.
Dim Cnt
Cnt = ResultObj.Count()
```

Name – Attributnamen ermitteln

Liefert den Namen eines Attributes zurück.

```
strName = ResultObj.Name( lAttribute )
```

Eigenschaften

Name (String)
Name des Attributes.

Parameter

lAttribute (Long Integer)
Nummer des Attributes.

Beispiel

```
' Display the names of all available attributes for the object.
Dim iAttr, lCnt
lCnt = ResultObj.Count()
For iAttr = 1 To lCnt
    Dim strName
    strName = ResultObj.Name( iAttr )
    WScript.Echo iAttr & ": " & strName
Next
```

Item – Zugriff auf Attribute

Stellt den Zugriff auf die Ergebnisdaten eines Ergebnisobjektes zur Verfügung.

```
Value = ResultObj.Item( lAttribute )
Value = ResultObj.Item( strAttribute )
```

Eigenschaften

Item (Variant)
Wert des Attributes.

Parameter

iAttribute (Long Integer)
Numerischer Index des Attributes.

strAttribute (Long Integer)
Name des Attributes.

Anmerkungen

Auf die Eigenschaft kann nur lesend zugegriffen werden.

Das Berechnungsergebnisobjekt entspricht exakt den Ergebnistabellen der PSS SINCAL Datenbank. D.h. die Attributnamen sind identisch mit den Feldbezeichnungen der Ergebnistabelle. Eine detaillierte Dokumentation aller Ergebnistabellen ist in der **PSS SINCAL Datenbankbeschreibung** verfügbar.

Beispiel

```
' Getting load flow result for node.
Dim LFNodeResult
Set LFNodeResult = NodeObj.Result( "LFNodeResult", 0 )
If LFNodeResult Is Nothing Then
Else
    Dim u_un
    u_un = LFNodeResult.Item( "U Un" )
    WScript.Echo "Node voltage at node U/Un = " & u_un & "%"
    Set LFNodeResult = Nothing
End If
```

4.1.4 Datenbankobjekt

Über dieses Objekt ist es möglich, auf die einzelnen Tabellen der Datenbank zuzugreifen.

Beispiel

```
' Database object.
Dim SimulateDatabase
Set SimulateDatabase = SimulateObj.DB EL
If SimulateDatabase Is Nothing Then
    WScript.Echo "Error: Getting database object failed!"
    WScript.Quit
End If
```

Nach der Verwendung muss die Instanz des Datenbankobjektes freigegeben werden. Dies erfolgt mit der folgenden Anweisung.

```
' Release the tabular object.
Set SimulateDataBase = Nothing
```

GetRowObj – Instanz eines Tabellenobjektes ermitteln

Stellt eine Instanz auf ein Tabellenobjekt zur Verfügung.

```
Set TableObj = SimulateDatabase.GetRowObj( strTable )
```

Parameter

strTable (String)
Name der Datenbanktabelle.

Rückgabewert

TableObj (Object)
Automatisierungsobjekt der Datenbanktabelle.

Beispiel

```
' Get the load flow node result tabular object.
Dim LFNodeResult
Set LFNodeResult = SimulateDatabase.GetRowObj( "LFNodeResult" )
If LFNodeResult Is Nothing Then
    WScript.Echo "Error: Getting LFNodeResult object failed!"
    WScript.Quit
End If
```

4.1.5 Tabellenobjekt

In diesem Objekt sind sämtliche in einer Datenbanktabelle enthaltenen Daten verfügbar. Die Daten im Tabellenobjekt sind, ähnlich wie in einer Kalkulationstabelle, in Zeilen und Spalten angeordnet. Jede Zeile stellt einen eindeutigen Datensatz und jede Spalte ein Feld bzw. Attribut innerhalb des Datensatzes dar.

Eine Instanz eines Tabellenobjektes kann über das Datenbankobjekt erzeugt werden.

Beispiel

```
' Get the load flow node result tabular object.
Dim LFNodeResult
Set LFNodeResult = SimulateDataBase.GetRowObj( "LFNodeResult" )
If LFNodeResult Is Nothing Then
    WScript.Echo "Error: Getting LFNodeResult object failed!"
    WScript.Quit
End If
```

Nach der Verwendung muss die Instanz des Tabellenobjektes freigegeben werden. Dies erfolgt mit der folgenden Anweisung.

```
' Release the tabular object.
Set LFNodeResult = Nothing
```

Open – Öffnen eines Tabellenobjektes

Öffnet die Datenbanktabelle eines Tabellenobjektes und ermöglicht den Zugriff auf die Datensätze.

```
hr = TableObj.Open()
```

Rückgabewert

hr (HRESULT)
Statuscode der Anweisung.

Beispiel

```
' Open a database table.
Dim hr
hr = TableObj.Open()
If hr <> 0 Then
    WScript.Echo "Error: Open failed!"
    WScript.Quit
End If
```

Close – Schließen eines Tabellenobjektes

Schließt eine Datenbanktabelle eines Tabellenobjektes.

```
TableObj.Close
```

Anmerkungen

Nach dem Schließen der Datenbanktabelle darf kein Zugriff mehr auf das Tabellenobjekt durchgeführt werden. Die Instanz des Tabellenobjektes muss ebenfalls freigegeben werden.

Beispiel

```
' Close a database table and release it.
TableObj.Close
Set LFNodeResult = Nothing
```

CountRows – Ermitteln der Datensatzanzahl

Stellt die Anzahl der Datensätze in einem Tabellenobjekt zur Verfügung.

```
lCnt = TableObj.CountRows
```

Eigenschaften

CountRows (Long Integer)
Anzahl der Datensätze in einem Tabellenobjekt.

Beispiel

```
' Open a database table and determine the count of records in it.
Dim hr
hr = TableObj.Open()
If hr <> 0 Then
    WScript.Echo "RecordCount: " & TableObj.CountRows
End If
```

MoveFirst, MoveLast, MoveNext, MovePrev – Positionierung in der Datenmenge

Positioniert den Datencursor innerhalb einer Datenmenge.

```
hr = TableObj.MoveFirst()
hr = TableObj.MoveNext()
hr = TableObj.MoveLast()
hr = TableObj.MovePrev()
```

Rückgabewert

hr (HRESULT)
Statuscode der Anweisung.

Anmerkungen

MoveFirst positioniert den Datencursor an den Anfang der Tabelle, **MoveLast** auf den letzten Datensatz einer Tabelle. **MoveNext** und **MovePrev** ermöglichen es, den Datencursor auf den nächsten bzw. vorigen Datensatz zu verschieben.

Beispiel

```
' Move to the first record and start reading.
Dim hr
hr = LFNodeResult.MoveFirst()
Do While hr = 0

    ' Your own code is here ...

    ' Move to next records.
    hr = LFNodeResult.MoveNext()
Loop
```

Count – Anzahl der möglichen Attribute

Liefert die Anzahl der Attribute für das Tabellenobjekt zurück.

```
lCnt = TableObj.Count
```

Rückgabewert

lCnt (Long Integer)
Anzahl der möglichen Attribute.

Anmerkungen

Diese Eigenschaft kann nur gelesen werden.

Beispiel

```
' Get the number of available attributes.
Dim lCnt
lCnt = TableObj.Count
```

Name – Attributnamen ermitteln

Liefert den Namen eines Attributes zurück.

```
strName = TableObj.Name( iAttribute )
```

Eigenschaften

Name (String)
Name des Attributes.

Parameter

iAttribute (Long Integer)
Nummer des Attributes.

Beispiel

```
' Display the names of all available attributes for the object.
Dim iAttr, lCnt
lCnt = TableObj.Count
For iAttr = 1 To lCnt
    Dim strName
    strName = TableObj.Name( iAttr )
    WScript.Echo iAttr & ": " & strName
Next
```

Item – Zugriff auf Attribute

Stellt den Zugriff auf die einzelnen Attribute (Felder) der aktuellen Datenzeile zur Verfügung.

```
Value = TableObj.Item( lAttribute )
Value = TableObj.Item( strAttribute )
```

Eigenschaften

Item (Variant)
Wert des Attributes.

Parameter

iAttribute (Long Integer)
Numerischer Index des Attributes.

strAttribute (Long Integer)
Name des Attributes.

Beispiel

```
' Get U_Un from the table.
Dim Val
Val = TableObj.Item( "U_Un" )
```

4.1.6 Meldungsobjekt

Mit dem Meldungsobjekt ist ein Zugriff auf die Meldungen, die während des Berechnungsvorganges generiert wurden, möglich.

Der Zugriff auf das Meldungsobjekt erfolgt mit Hilfe des Simulationsobjekts. Das folgende Beispiel zeigt, wie alle Meldungen der Berechnung mit Hilfe einer Programmschleife ausgegeben werden können.

Beispiel

```
' Get messages from simulation.
Dim objMessages
Set objMessages = SimulateObj.Messages

Dim strType
Dim intMsgIdx
For intMsgIdx = 1 To objMessages.Count
    Dim Msg
    Set Msg = objMessages.Item( intMsgIdx )

    Select Case Msg.Type
        case 1 ' STATUS
        case 2 ' INFO
        case 3 ' WARNING
            WScript.Echo Msg.Text
        case 4 ' ERROR
            WScript.Echo Msg.Text
    End Select

    Set Msg = Nothing
Next

' Release message object if not longer needed.
Set objMessages = Nothing
```

Count – Anzahl der verfügbaren Meldungen

Liefert die Anzahl der Meldungen zurück.

```
Cnt = objMessages.Count
```

Eigenschaften

Count (Long Integer)
Anzahl der verfügbaren Meldungen.

Anmerkungen

Diese Eigenschaft steht nur lesend zur Verfügung.

Beispiel

```
' Get the number of available messages.
Dim Cnt
Cnt = objMessages.Count
WScript.Echo "Number of Messages: " & intMsgCnt
```

Item – Zugriff auf ein Meldungsdatenobjekt

Stellt den Zugriff auf ein Meldungsdatenobjekt zur Verfügung.

```
Set Msg = objMessages.Item( iMsgIndex )
```

Eigenschaften

Item (Object)
Automatisierungsobjekt einer Meldung.

Parameter

iMsgIndex (Long Integer)
Numerischer Index der Meldung. Der zulässige Index beginnt bei 1 und endet bei der Anzahl der Meldungen.

Anmerkungen

Diese Eigenschaft steht nur lesend zur Verfügung.

Beispiel

```
' Get a message.
Dim Msg
Set Msg = objMessages.Item( 1 )
```

4.1.7 Meldungsdatenobjekt

Dieses Objekt repräsentiert eine Meldung. Von dieser können der Meldungstext und der Meldungstyp sowie andere Daten einer Meldung abgerufen werden.

Beispiel

```
' Get the first message and display the message type and text.
Dim Msg
Set Msg = objMessages.Item( 1 )

Select Case Msg.Type
    case 1 ' STATUS
       WScript.Echo "Status: " & Msg.Text
    case 2 ' INFO
       WScript.Echo "Info: " & Msg.Text
    case 3 ' WARNING
       WScript.Echo "Warning: " & Msg.Text
    case 4 ' ERROR
       WScript.Echo "Error: " & Msg.Text
End Select
```

Text – Meldungstext

Ermöglicht den Zugriff auf den Meldungstext einer Berechnungsmeldung.

```
strText = Msg.Text
```

Eigenschaften

Text (String)
Meldungstext.

Anmerkungen

Diese Eigenschaft kann nur gelesen werden.

Beispiel

```
' Get the first message and display the message text.
Dim Msg
Set Msg = objMessages.Item( 1 )

WScript.Echo Msg.Text
```

Type – Meldungstyp

Kennzeichnet die Art der Meldung.

```
iType = Msg.Type
```

Eigenschaften

Type (Integer)
Vordefinierte Kennziffer des Meldungstyps.

Kennziffer	Meldungstyp	Beschreibung
1	Statusmeldung	Statusmeldungen werden von den Berechnungsmethoden beim Durchlaufen der verschiedenen Funktionen ausgegeben.
2	Informationsmeldung	Informationsmeldungen beinhalten allgemeine Information (Anzahl isolierter Knoten usw.).
3	Warnungen	Warnungen stellen tolerierbare Fehler während des Berechnungsvorganges dar.
4	Fehlermeldungen	Fehlermeldungen kennzeichnen, dass schwerwiegende Fehler während des Berechnungsvorganges aufgetreten sind und die Berechnung nicht ordnungsgemäß beendet werden konnte.

Beispiel

```
' Get the first message and display the message type.
Dim Msg
Set Msg = objMessages.Item( 1 )

Select Case Msg.Type
    case 1 ' STATUS
       WScript.Echo "Status"
    case 2 ' INFO
       WScript.Echo "Info"
    case 3 ' WARNING
       WScript.Echo "Warning"
    case 4 ' ERROR
       WScript.Echo "Error"
End Select
```

CountObjectIds – Anzahl der Netzelemente

Kennzeichnet die Anzahl der Netzelemente, auf die sich die Meldung bezieht.

```
lCntElements = Msg.CountObjectIds
```

Eigenschaften

CountObjectIds (Long Integer)
Anzahl der Netzelemente.

Anmerkungen

Diese Eigenschaft kann nur gelesen werden.

Beispiel

```
' Loop over all objects of the current message.
If Msg.CountObjectIds > 0 Then
    Dim i
    For i = 1 To Msg.CountObjectIds
        ' ...
    Next
```

```
End If
```

ObjectIdAt, ObjectTypeAt – Netzelementdaten

Stellt die Objekt-ID und den Objekttyp eines Netzelementes zur Verfügung.

```
lObjID   = Msg.ObjectIdAt( lIndex )
iObjType = Msg.ObjectTypeAt( lIndex )
```

Parameter

lIndex (Long Integer)
Nummerischer Index beginnend bei 1.

Eigenschaften

ObjectIdAt (Long Integer)
ID des Netzelementes.

ObjectTypeAt (Integer)
Datenbank und Typ des Netzelementes in einer Maske.

Anmerkungen

Der Objekttyp wird in den ersten 12 Bits gespeichert und der Datenbanktyp in den höchsten 4 Bits. Um den Objekttyp zu ermitteln, muss die Bitmaske mit 0FFF kombiniert werden. Den Datenbanktyp erhält man durch Verschieben der Bits um 12 Stellen nach rechts.

Diese Eigenschaften stehen nur im Lesezugriff zur Verfügung.

Beispiel

```
' Loop over all objects of the current message.
If Msg.CountObjectIds > 0 Then
    Dim i
    Dim strObjects
    For i = 1 To Msg.CountObjectIds
        If strObjects <> "" Then
            strObjects = strObjects & ", "
        End If

       ' Determine database and object type.
        Dim sDBType
        Dim sRowType
        sDBType = Msg.ObjectTypeAt( i ) \ &H1000
        sRowType = Msg.ObjectTypeAt( i ) And &H0FFF

        strObjects = strObjects & Msg.ObjectIdAt( i ) & "(" & sRowType & ")"
    Next
    WScript.Echo strObjects
End If
```

4.2 Berechnungsobjekte und deren Attribute

4.2.1 Verfügbare Berechnungsobjekte

Die folgenden Tabellen zeigen die verfügbaren Berechnungsobjekte für die verschiedenen Netztypen.

Elektronetze

Objekttyp	Beschreibung
Allgemeine Daten	
CalcParameter	Berechnungsparameter
VoltageLevel	Netzebene
NetworkGroup	Netzbereich
Knoten/Sammelschienen	
Node	Knoten
Knotenelemente	
SynchronousMachine	Synchronmaschine
PowerUnit	Kraftwerksblock
Infeeder	Netzeinspeisung
DCInfeeder	DC-Einspeisung
AsynchronousMachine	Asynchronmaschine
Load	Allgemeine Last
ShuntImpedance	Querimpedanz
ShuntReactor	Querdrossel
ShuntCondensator	Querkondensator
VarShuntElement	Variables Querelement
HarResNet	Quer Oberschwingungs-Resonanznetz
Zweigelemente	
TwoWindingTransformer	Zweiwicklungstransformator
ThreeWindingTransformer	Dreiwicklungstransformator
Line	Leitung
VarSerialElement	Variables Längselement
SerialReactor	Längsdrossel
SerialCondensator	Längskondensator
HarBranchResNet	Längs Oberschwingungs-Resonanznetz
Zusatzelemente	
ProtLocation	Einbauort des Schutzgerätes
ProtOCFault	Fehleruntersuchung

Wassernetze

Objekttyp	Beschreibung
Allgemeine Daten	
FlowCalcParameter	Berechnungsparameter
FlowNetworkLevel	Netzebene
FlowNetworkGroup	Netzbereich
Knoten/Sammelschienen	
FlowNode	Knoten
Knotenelemente	
FlowWaterTower	Hochbehälter
FlowPump	Pumpeinspeisung
FlowConsumer	Verbraucher
FlowPressureBuffer	Druckbuffer
FlowLeakage	Leck
Zweigelemente	
FlowLine	Leitung
FlowValve	Schieber/Rückschlagventil
FlowPumpLine	Druckverstärkerpumpe
FlowConstLine	Konst. Druckabfall/Konst. Fluss
FlowPressureReg	Druckregler

Gasnetze

Objekttyp	Beschreibung
Allgemeine Daten	
FlowCalcParameter	Berechnungsparameter
FlowNetworkLevel	Netzebene
FlowNetworkGroup	Netzbereich
Knoten/Sammelschienen	
FlowNode	Knoten
Knotenelemente	
FlowInfeederG	Einspeisung Gas
FlowConsumer	Verbraucher
FlowPressureBuffer	Druckbuffer
FlowLeakage	Leck
Zweigelemente	
FlowLine	Leitung
FlowValve	Schieber/Rückschlagventil
FlowConstLine	Konst. Druckabfall/Konst. Fluss
FlowPressureReg	Druckregler
FlowCompressor	Kompressor

Wärme-/Kältenetze

Objekttyp	Beschreibung
Allgemeine Daten	
FlowCalcParameter	Berechnungsparameter
FlowNetworkLevel	Netzebene
FlowNetworkGroup	Netzbereich
Knoten/Sammelschienen	
FlowNode	Knoten
Knotenelemente	
FlowInfeederH	Einspeisung Wärme/Kälte
FlowPump	Pumpeinspeisung
FlowConsumer	Verbraucher
FlowPressureBuffer	Druckbuffer
FlowLeakage	Leck
Zweigelemente	
FlowLine	Leitung
FlowValve	Schieber/Rückschlagventil
FlowPumpLine	Druckverstärkerpumpe
FlowConstLine	Konst. Druckabfall/Konst. Fluss
FlowPressureReg	Druckregler
FlowThermoReg	Temperaturregler
FlowHeatExchanger	Wärmetauscher

4.2.2 Allgemeine Topologieattribute

Die allgemeinen Topologieattribute sind sowohl bei den Knoten als auch bei den Netzelementen verfügbar. Damit können wichtige Basisinformationen wie Name, Anschlussphasen sowie Errichtung- und Stilllegungszeitpunkt abgefragt werden.

Attribut	Status	Beschreibung
Knoten		
TOPO.ID	Read	Interne ID des Knotens
TOPO.DBID	Read	Datenbank ID des Knotens (Node_ID)
TOPO.Name	Read	Name des Knotens
TOPO.ShortName	Read	Kurzname des Knotens
TOPO.Phase	Read	Anschlussphasen (wird dynamisch durch die angeschlossenen Elemente bestimmt) 1: L1 2: L2 3: L3 4: L12 5: L23 6: L31 7: L123

TOPO.TI	Read / Write	Errichtungszeitpunkt
TOPO.TS	Read / Write	Stilllegungszeitpunkt
Netzelemente		
TOPO.ID	Read	Interne ID des Netzelementes
TOPO.DBID	Read	Datenbank ID des Netzelementes (Element_ID)
TOPO.Name	Read	Name des Netzelementes
TOPO.ShortName	Read	Kurzname des Netzelementes
TOPO.State	Read / Write	Betriebszustand des Netzelementes 0: außer Betrieb 1: in Betrieb
TOPO.TI	Read / Write	Errichtungszeitpunkt
TOPO.TS	Read / Write	Stilllegungszeitpunkt
TOPO.Node1.ID	Read	Interne ID des 1. Knotens (bis zu 3 Knoten sind möglich)
TOPO.Node1.DBID	Read	Datenbank ID des 1. Knotens
TOPO.Terminal1.ID	Read	Interne ID des 1. Anschlusses (bis zu 3 Anschlüsse sind möglich)
TOPO.Terminal1.DBID	Read	Datenbank ID des 1. Anschlusses
TOPO.Terminal1.State	Read / Write	Schaltzustand des 1. Anschlusses 0 : Schalter geöffnet 1: Schalter geschlossen
TOPO.Terminal1.Phase	Read	Anschlussphase 1: L1 2: L2 3: L3 4: L12 5: L23 6: L31 7: L123
Zusatzelemente		
TOPO.ID	Read	Interne ID des Zusatzelementes
TOPO.DBID	Read	Datenbank ID des Zusatzelementes (Element_ID)
TOPO.Name	Read	Name des Zusatzelementes

Das Attribut **ID** enthält einen eindeutigen Schlüssel, der jedes Objekt in den Berechnungsmethoden eindeutig identifiziert.

Im Attribut **DBID** ist die Datenbank ID des jeweiligen Knoten, Netzelementes oder Anschlusses verfügbar.

Eine Besonderheit stellt das Attribut **State** dar. Dies ist beim Netzelement und dessen Anschlüssen verfügbar. Dieses Attribut kennzeichnet den Betriebszustand des Netzelementes bzw. den Schaltzustand des jeweiligen Anschlusses. Durch simples Ändern dieses Attributes kann das Netzelement ein- bzw. ausgeschaltet werden.

4.2.3 Attribute der Berechnungsobjekte für Elektronetze

Berechnungsparameter (CalcParameter)

Attributname	Datentyp	Einheit	Beschreibung
ViewDate	Double		View Date
LoadDataDate	Double		Load Data Date

Sref	Double	MVA	Reference Power
FrqNet	Double	Hz	Frequency
Uref	Double	kV	Reference Voltage
ExportForm	Integer		Export Format for Names 0: Name 1: Short name
IncreaseLoads	Integer		Use Increased Loads 0: No 1: Yes
ContrAdjustment	Integer		Controller Adjustment 1: Discrete 2: Continuous
FlatStart	Integer		Flat Start 0: No 1: Yes
ChangeLFMethode	Integer		Change Load Flow Method at Convergence Problems 0: Off 1: On
LFPreCalc	Integer		Pre-Calculate 0: No 1: Yes
LFMethod	Integer		Load Flow Procedure 1: Current iteration 2: Newton-Raphson 3: Admittance matrix 5: Unbalanced
StoreRes	Integer		Include Load Curve Result in Database 0: Due to method 1: All 2: Restricted elements only 3: All elements in case of restrictions
ImpLoad	Integer		Impedance Load Conversion 0: No 1: Normal 2: Extended
LFControl	Integer		Enable Automatic Controller Change 0: No 1: Normal 2: Extended
Island	Integer		Island Operation 0: No 1: Yes
StartTime	Double		Start Time Load Curve
Duration	Double		Duration Load Curve
TimeStep	Double		Time Step Load Curve
IncrStartDate	Double		Start Date Load Increase
IncrEndDate	Double		End Date Load Increase
PeakCurrentCalc	Integer		Peak Short Circuit Current Calculation Type 1: Meshed network 2: Non-meshed network 3: Equivalent frequency procedure
TrippCurrentCalc	Integer		Tripping Current Calculation Type 1: IANEU VDE0102/1.90 – IEC 909 2: IAALT VDE0102/10.71
EcoInflation	Double	%	Cost increase

Netzebene (VoltageLevel)

Attributname	Datentyp	Einheit	Beschreibung
Un	Double	kV	Nominal Voltage

Uop	Double	kV	Network Operating Voltage
f	Double	Hz	Frequency
fRD	Double	Hz	Ripple Control Frequency
Temp_Line	Double	°C	Overhead Line Conductor Temperature
Temp_Cabel	Double	°C	Cable Conductor Temperature
CalcSC	Integer		Calculate Short Circuit 0: No 1: Yes
CalcNpt	Integer		Calculate Current through Neutral Points 0: No 1: Yes
FlagUsc	Integer		Voltage Data due to VDE/IEC 1: c-value 2: Source voltage
Uk	Double	kV	Source Voltage
c	Double	1	c Value
ts	Double	s	Switch Delay
Ipmax	Double	kA	Maximum Admissible Surge Current
Ibrkmax	Double	kA	Maximum Admissible Tripping Current
Upre	Double	pu	Pre-Fault Voltage due to ANSI/IEEE
Flag_Toleranz	Integer		Voltage Tolerance – Low Voltage Networks 1: 6 % 2: 10 %

Netzbereich (NetworkGroup)

Attributname	Datentyp	Einheit	Beschreibung
Flag_IC	Integer		Transfer Active 0: No 1: Yes
Pdes	Double	MW	Interchange Leaving the Area
Ptol	Double	MW	Interchange Tolerance Bandwidth
Flag_Malfunc	Integer		Malfunction 0: None 1: All elements 2: Loaded elements 3: All lines 4: Loaded lines 5: All lines and transformers 6: Loaded lines and transformers
Flag_Connectors	Integer		Consider Connectors in Malfunction and Caused Malfunction 0: No 1: Yes
Util_BaseLimit	Double	%	Base Utilization Limit
Flag_CausedMalfunc	Integer		Caused Malfunction 0: None 1: Marked areas 2: Own area
Flag_CausedElem	Integer		Caused Elements 1: Loaded elements 2: Loaded lines 3: Loaded lines and transformers
Util_CausedLimit	Double	%	Caused Utilization Limit
Flag_CausedForeign	Integer		Marked for Caused Malfunction 0: No 1: Yes

Flag_Util	Integer		Show Elements outside Limits 0: None 1: Elements and nodes 2: Elements 3: Lines, transformers and nodes 4: Lines and transformers 5: Lines and nodes 6: Lines

Knoten (Node)

Attributname	Datentyp	Einheit	Beschreibung
Uref	Double	kV	Voltage Target Value
uul	Double	%	Voltage Upper Limit
ull	Double	%	Voltage Lower Limit
StartU	Double	kV	Initial Voltage
StartPhi	Double	°	Angle – Initial Voltage
Ik2	Double	kA	Maximum Admissible Short Circuit Current
Ip	Double	kA	Maximum Admissible Peak Short Circuit Current
uul1	Double	%	Additional Voltage Upper Limit
ull1	Double	%	Additional Voltage Lower Limit
FlagPhase	Integer		Preferred Fault Phase 1: L1 2: L2 3: L3 4: L12 5: L23 6: L31 7: L123

Synchronmaschine (SynchronousMachine)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Machine	Integer		Type of Machine 1: Turbo generator 2: Hydro gen. (amort.) 3: Hydro generator 4: Condenser 5: Non-interconnected equivalent 6: Power station equivalent 7: Transmission system equivalent 8: Distribution system equivalent
Sn	Double	MVA	Rated Apparent Power
P	Double	MW	Active Power
Q	Double	Mvar	Reactive Power
u	Double	%	Generator Voltage Percentage
Un	Double	kV	Rated Voltage
Ug	Double	kV	Generator Voltage Absolute
R_X	Double	pu	Ratio R/X – Positive-Phase Sequence
xd2sat	Double	%	Saturated Subtransient Reactance
xd1sat	Double	%	Saturated Transient Reactance
xi	Double	%	Internal Reactance
Ugmax	Double	%	Maximum Generator Voltage
Ikp	Double	kA	Sustained Short Circuit Current of Compound Machines

Flag_LF	Integer		Load Flow Type 1: |I| and phi 2: P and Q 3: |usrc| and delta 4: |S| and cosphi 5: |Usrc| and delta 6: P and |u| 7: P and |U| 8: |uterm| and delta 9: |Uterm| and delta
I	Double	kA	Basic Current Source
delta	Double	°	Voltage Angle
S	Double	MVA	Apparent Power
cosphi	Double	1	Power Factor
Null- u. Gegensystem			
Flag_Z0	Integer		Grounding 0: Not grounded 1: Fixed grounded 2: Grounded w. impedances
Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: Z0/Z1 and R0/X0 2: R0 and X0
R0_X0	Double	pu	Ratio R/X – Zero-Phase Sequence
Z0_Z1	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence
X22	Double	%	Reactance – Negative-Phase Sequence
R2_X2	Double	pu	Ratio R/X – Negative-Phase Sequence
Reglerdaten			
Unode	Double	%	Controlled Voltage at Controller Node
Regelbanddaten			
Pmin	Double	MW	Active Power – Lower Limit
Pmax	Double	MW	Active Power – Upper Limit
Qmin	Double	Mvar	Reactive Power – Lower Limit
Qmax	Double	Mvar	Reactive Power – Upper Limit
Umin	Double	%	Voltage Lower Limit
Umax	Double	%	Voltage Upper Limit
cosphi_lim	Double		Limit Power Factor
Dynamische Daten			
xd2	Double	pu	Subtransient Reactance
Zuverlässigkeitsdaten			
Flag_LP	Integer		Load Priority 1: High 2: Medium 3: Normal 4: Small 5: Low
CustCnt	Long Integer	1	Number of Supplied Customers

Kraftwerksblock (PowerUnit)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Machine	Integer		Type of Machine 1: Turbo generator 2: Hydro gen. (amort.) 3: Hydro generator 4: Condenser 5: Non-interconnected equivalent 6: Power station equivalent 7: Transmission system equivalent 8: Distribution system equivalent
Sn	Double	MVA	Rated Apparent Power
Un	Double	kV	Rated Voltage
R_X	Double	pu	Ratio R/X – Positive-Phase Sequence
xd2	Double	%	Subtransient Reactance
xi	Double	%	Internal Reactance
Ugmax	Double	%	Maximum Generator Voltage
Ug	Double	kV	Rated Voltage Generator
cosphin	Double	1	Rated Power Factor
Ikp	Double	kA	Sustained Short Circuit Current of Compound Machines
xd1sat	Double	%	Saturated Transient Reactance
xd2sat	Double	%	Saturated Subtransient Reactance
Un2	Double	kV	Rated Voltage Transformer – Network Side
Un1	Double	kV	Rated Voltage Transformer – Generator Side
Snt	Double	MVA	Rated Apparent Power Transformer
Smax	Double	MVA	Full Load Power
ur	Double	%	Short Circuit Voltage – Ohmic Part
Flag_LF	Integer		Load Flow Type 1: |I| and phi 2: P and Q 3: |usrc| and delta 4: |S| and cosphi 5: P and |u| 6: |Usrc| and delta 7: P and |U| 8: |uterm| and delta 9: |Uterm| and delta
phi	Double	°	Phase Angle
I	Double	kA	Basic Current Source
P	Double	MW	Active Power
Q	Double	Mvar	Reactive Power
S	Double	MVA	Apparent Power
cosphi	Double	1	Power Factor
u	Double	%	Generator Voltage Percentage
Null- u. Gegensystem			
Flag_Z0	Integer		Grounding 0: Not grounded 1: Fixed grounded 2: Grounded w. impedances
Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: Z0/Z1 and R0/X0 2: R0 and X0
R0_X0	Double	pu	Ratio R/X – Zero-Phase Sequence
Z0_Z1	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence

X22	Double	%	Saturated Reactance – Negative-Phase Sequence
R2_X2	Double	pu	Ratio R/X – Negative-Phase Sequence
Reglerdaten			
Flag_Roh	Integer		State – Tap Position 1: Fixed 2: Variable
roh	Double	1	Present Tap Position
rohl	Double	1	Minimum Tap Position
rohu	Double	1	Maximum Tap Position
alpha	Double	°	Surplus Voltage Angle
Unode	Double	%	Controlled Voltage at Controller Node
Regelbanddaten			
Pmin	Double	MW	Active Power – Lower Limit
Pmax	Double	MW	Active Power – Upper Limit
Qmin	Double	Mvar	Reactive Power – Lower Limit
Qmax	Double	Mvar	Reactive Power – Upper Limit
Umin	Double	%	Voltage Lower Limit
Umax	Double	%	Voltage Upper Limit
cosphi_lim	Double		Limit Power Factor

Netzeinspeisung (Infeeder)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Typ	Integer		State – Input Values 1: R and X 2: R/X and Sk2
R_X	Double	pu	Resistance/Reactance
xi	Double	%	Internal Reactance
Flag_LF	Integer		Load Flow Type 1: |I| and phi 2: P and Q 3: |usrc| and delta 4: |S| and cosphi 5: P and |u| 6: |Usrc| and delta 7: P and |U| 8: |uterm| and delta 9: |Uterm| and delta
I	Double	kA	Basic Current Source
P	Double	MW	Active Power
Q	Double	Mvar	Reactive Power
u	Double	%	Voltage
S	Double	MVA	Apparent Power
cosphi	Double	1	Power Factor
Ug	Double	kV	Generator Voltage
Null- u. Gegensystem			
Flag_Z0	Integer		Grounding 0: Not grounded 1: Fixed grounded 2: Grounded w. impedances

Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: Z0/Z1 and R0/X0 2: R0 and X0
Z0_Z1	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
R0_X0	Double	pu	Ratio R/X – Zero-Phase Sequence
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence
Reglerdaten			
Unode	Double	%	Controlled Voltage at Controller Node
Regelbanddaten			
Pmin	Double	MW	Active Power – Lower Limit
Pmax	Double	MW	Active Power – Upper Limit
Qmin	Double	Mvar	Reactive Power – Lower Limit
Qmax	Double	Mvar	Reactive Power – Upper Limit
Umin	Double	%	Voltage Lower Limit
Umax	Double	%	Voltage Upper Limit
cosphi_lim	Double		Limit Power Factor

DC-Einspeisung (DCInfeeder)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Input_Type	Integer		DC Input 1: P and Q 2: P and cosphi 3: Inverter
P	Double	MW	Active Power
Q	Double	Mvar	Reactive Power
cosphi	Double	1	Power Factor
DC_power	Double	kW	Installed DC-Power
fDC_power	Double	1	Factor Installed DC-Power
fP	Double	1	Multiplication Factor – Active Power
fQ	Double	1	Multiplication Factor – Reactive Power
DC_losses	Double	%	Losses until Inverter
Eta_Inverter	Double	%	Efficiency – Inverter
Q_Inverter	Double	%	Reactive Power Demand – Inverter
Ctrl_power	Double	W	Controller Power
Tr_UrNet	Double	kV	Rated Voltage Netside – Transformer
Tr_Sr	Double	kVA	Rated Apparent Power – Transformer
Tr_uk	Double	%	Reference Short Circuit Voltage – Transformer
Tr_rx	Double	pu	Ratio R/X – Transformer
Umin_Inverter	Double	%	Minimum Voltage – Inverter
Umax_Inverter	Double	%	Maximum Voltage – Inverter
t_off	Double	s	Switch Off Time
Flag_Connect	Integer		Type of Connecting 1: Directly 2: Transformer
Flag_I	Flag_I		Control Current State 0: Not active at load flow 1: Active at load flow
Ireg	Ireg	kA	Control Current
pk	pk	1	Reference Compensation Power

Asynchronmaschine (AsynchronousMachine)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Typ	Integer		Input Type of Asynchronous Machine 1: Pn 2: In 3: NEMA
Pn	Double	MW	Rated Active Power
Un	Double	kV	Rated Voltage
Speedn	Double	1/min	Rated Speed
pol	Double	1	Pole Pair Number
cosphin	Double	pu	Rated Power Factor
etan	Double	pu	Rated Efficiency
IaIn	Double	pu	Current Ratio At Start-Up
R_X	Double	pu	Ratio R/X – Positive-Phase Sequence
Inm	Double	kA	Rated Current
Flag_LF	Integer		Load Flow Type 1: P and Q 2: P and cosphi 3: P/Pn and cosphi 4: U, I und cosphi 5: DFIG (P, Q and Slip)
P	Double	MW	Active Power
Q	Double	Mvar	Reactive Power
cosphi	Double	1	Power Factor
ppn	Double	pu	Utilization
I	Double	kA	Basic Current Source
Slip	Double	%	Slip
Flag_SC	Integer		Short Circuit Behavior 1: Ik" + ip / I1c + Iint 2: ip / I1c 3: Ignore
Null- u. Gegensystem			
Flag_Z0	Integer		Grounding 0: Not grounded 1: Fixed grounded 2: Grounded w. impedances
Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: Z0/Z1 and R0/X0 2: R0 and X0
Z0_Z1	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
R0_X0	Double	pu	Ratio R/X – Zero-Phase Sequence
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence
Ia2In	Double	pu	Current Ratio At Start-Up
R2_X2	Double	pu	Ratio R/X – Negative-Phase Sequence
Kennlinien			
TA	Double	s	Starting Time Power Unit Data
GD2	Double	Mpm²	Momentum Power Unit Data
Motoranlauf			

Flag_StartUpCtrl	Integer		Start Up Control 0: None 1: Current 2: Auto transformer 3: Current and auto transformer 4: Capacitor 5: Current and capacitor
ConStart	Integer		Start-Up Circuitry 1: Star 2: Delta 3: Star/delta
ConRun	Integer		Characteristic Data Circuitry 1: Star 2: Delta
Zuverlässigkeitsdaten			
CustCnt	Long Integer	1	Number of Supplied Customers

Allgemeine Last (Load)

Attributname	Datentyp	Einheit	Beschreibung
Flag_LoadTyp	Integer		Load Type 1: Load 2: Customer load
Flag_LF	Integer		Load Input 1: P, Q and (u) 2: P, Q and (U) 3: S, cosphi and u 4: S, cosphi and U 5: I, cosphi and u 6: I, cosphi and U 7: P and I 8: E, cosphi and t 9: Eap and Eaq 10: Pi and Qi 11: P, cosphi and u 12: P, cosphi and U 13: Pi, Qi and (u) – star 14: Pij, Qij and (u) – delta 15: P, Q and (u) – delta
P	Double	MW	Active Power
Q	Double	Mvar	Reactive Power
u	Double	%	Voltage
Ul	Double	kV	Voltage
S	Double	MVA	Apparent Power
cosphi	Double	pu	Power Factor
I	Double	kA	Current
P1	Double	MW	Active Power Phase 1
Q1	Double	Mvar	Reactive Power Phase 1
P2	Double	MW	Active Power Phase 2
Q2	Double	Mvar	Reactive Power Phase 2
P3	Double	MW	Active Power Phase 3
Q3	Double	Mvar	Reactive Power Phase 3
P12	Double	MW	Active Power Phase 12
Q12	Double	Mvar	Reactive Power Phase 12
P23	Double	MW	Active Power Phase 23
Q23	Double	Mvar	Reactive Power Phase 23
P31	Double	MW	Active Power Phase 31

Q31	Double	Mvar	Reactive Power Phase 31
Null- u. Gegensystem			
Flag_Z0	Integer		Grounding 0: Not grounded 1: Fixed grounded 2: Grounded w. impedances
Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: Z0/Z1 and R0/X0 2: R0 and X0
Z0_Z1	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
R0_X0	Double	pu	Ratio R/X – Zero-Phase Sequence
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence
Pneg	Double	MW	Active Power – Negative-Phase Sequence
Qneg	Double	Mvar	Reactive Power – Negative-Phase Sequence
Dynamische Daten			
ResFlux1	Double	pu	Residual Flux Phase L1
ResFlux2	Double	pu	Residual Flux Phase L2
ResFlux3	Double	pu	Residual Flux Phase L3

Querimpedanz (ShuntImpedance)

Attributname	Datentyp	Einheit	Beschreibung
Un	Double	kV	Rated Voltage
R	Double	Ohm	Resistance
X	Double	Ohm	Reactance
Nullsystem			
Flag_Z0	Integer		Grounding 0: Not grounded 1: Fixed grounded 2: Grounded w. impedances
Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: Z0/Z1 and R0/X0 2: R0 and X0
R0_X0	Double	pu	Ratio R/X – Zero-Phase Sequence
Z0_Z1	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence
Dynamische Daten			
ResFlux1	Double	pu	Residual Flux Phase L1
ResFlux2	Double	pu	Residual Flux Phase L2
ResFlux3	Double	pu	Residual Flux Phase L3

Querdrossel (ShuntReactor)

Attributname	Datentyp	Einheit	Beschreibung
Sn	Double	MVA	Rated Apparent Power

Un	Double	kV	Rated Voltage
Vfe	Double	kW	Iron Losses
Vcu	Double	kW	Copper Losses
Nullsystem			
Flag_Z0	Integer		Grounding 0: Not grounded 1: Fixed grounded 2: Grounded w. impedances
Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: Z0/Z1 and R0/X0 2: R0 and X0
R0_X0	Double	pu	Ratio R/X – Zero-Phase Sequence
Z0_Z1	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence
Reglerdaten			
Flag_Roh	Integer		Controller State 1: Fix 2: Variable – node 3: Variable – terminal
roh	Integer	1	
rohl	Integer	1	Present Tap Position
rohu	Integer	1	Minimum Tap Position
deltaS	Double		Maximum Tap Position
uul	Double	%	Voltage Upper Limit
ull	Double	%	Voltage Lower Limit
Qmin	Double	Mvar	Minimum Total Reactive Power
Qmax	Double	Mvar	Maximum Total Reactive Power
CosPhiMin	Double	1	Cosinus Phi Minimum
CosPhiMax	Double	1	Cosinus Phi Minimum
Dynamische Daten			
ResFlux1	Double	pu	Residual Flux Phase L1
ResFlux2	Double	pu	Residual Flux Phase L2
ResFlux3	Double	pu	Residual Flux Phase L3

Querkondensator (ShuntCondensator)

Attributname	Datentyp	Einheit	Beschreibung
Sn	Double	MVA	Rated Apparent Power
Un	Double	kV	Rated Voltage
Vdi	Double	kW	Dielectric Losses
Nullsystem			
Flag_Z0	Integer		Grounding 0: Not grounded 1: Fixed grounded 2: Grounded w. impedances
Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: Z0/Z1 and R0/X0 2: R0 and X0
R0_X0	Double	pu	Ratio R/X – Zero-Phase Sequence

Z0_Z1	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence
Reglerdaten			
Flag_Roh	Integer		Controller State 1: Fix 2: Variable – node 3: Variable – terminal
roh	Integer	1	
rohl	Integer	1	Present Tap Position
rohu	Integer	1	Minimum Tap Position
deltaS	Double		Maximum Tap Position
uul	Double	%	Voltage Upper Limit
ull	Double	%	Voltage Lower Limit
Qmin	Double	Mvar	Minimum Total Reactive Power
Qmax	Double	Mvar	Maximum Total Reactive Power
CosPhiMin	Double	1	Cosinus Phi Minimum
CosPhiMax	Double	1	Cosinus Phi Minimum

Variables Querelement (VarShuntElement)

Attributname	Datentyp	Einheit	Beschreibung
Flag_LF	Integer		Load Flow Input 1: Power 2: Impedance 3: Model 4: Mixed power 5: Function
Flag_LoadType	Integer		Load Flow Type 1: Z constant 2: P and Q constant 3: I constant
Flag_Macro_LF	Integer		Model Type Load Flow 0: None 1: Controller 2: Equivalent circuit
Plf	Double	MW	Active Power Load Flow
Qlf	Double	Mvar	Reactive Power Load Flow
Ulf	Double	kV	Voltage Load Flow
Rlf	Double	Ohm	Resistance Load Flow
Xlf	Double	Ohm	Reactance Load Flow
fPk	Double	pu	Factor Constant Active Power
fPi	Double	pu	Factor Current Dependent Active Power
fPu	Double	pu	Factor Voltage Dependent Active Power
fQk	Double	pu	Factor Constant Reactive Power
fQi	Double	pu	Factor Current Dependent Reactive Power
fQu	Double	pu	Factor Voltage Dependent Reactive Power
f_p_1	Double	pu	Factor 1 Active Power
f_q_1	Double	pu	Factor 1 Reactive Power
e_p_1	Double	pu	Exponent 1 Active Power
e_q_1	Double	pu	Exponent 1 Reactive Power
f_p_2	Double	pu	Factor 2 Active Power
f_q_2	Double	pu	Factor 2 Reactive Power

e_p_2	Double	pu	Exponent 2 Active Power
e_q_2	Double	pu	Exponent 2 Reactive Power
f_p_3	Double	pu	Factor 3 Active Power
f_q_3	Double	pu	Factor 3 Reactive Power
e_p_3	Double	pu	Exponent 3 Active Power
e_q_3	Double	pu	Exponent 3 Reactive Power
Rsc	Double	Ohm	Resistance Circuit Input
Xsc	Double	Ohm	Reactance Circuit Input
Null- u. Gegensystem			
Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: Z0/Z1 and R0/X0 2: R0 and X0
Z0_Z1	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
R0_X0	Double	pu	Ratio R/X – Zero-Phase Sequence
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence
Pneg	Double	MW	Active Power – Negative-Phase Sequence
Qneg	Double	Mvar	Reactive Power – Negative-Phase Sequence

Quer Oberschwingungs-Resonanznetz (HarResNet)

Attributname	Datentyp	Einheit	Beschreibung
Un	Double	kV	Rated Voltage
R	Double		Resistance at Network Frequency
X	Double		Reactance at Network Frequency
Faktor	Double		Initial Value Factor
Impedance	Integer		Determine Impedance 1: Vmax 2: Imax
FlagZ0	Integer		Zero Sequence Data 0: Not grounded 1: Fixed grounded

Zweiwicklungstransformator (TwoWindingTransformer)

Attributname	Datentyp	Einheit	Beschreibung
Un1	Double	kV	Rated Voltage (Side 1)
Un2	Double	kV	Rated Voltage (Side 2)
Sn	Double	MVA	Rated Apparent Power
Smax	Double	MVA	Full Load Power
Smax1	Double	MVA	First Additional Full Load Power
Smax2	Double	MVA	Second Additional Full Load Power
Smax3	Double	MVA	Third Additional Full Load Power
uk	Double	%	Reference Short Circuit Voltage
ur	Double	%	Short Circuit Voltage – Ohmic Part
Vfe	Double	kW	Iron Losses
i0	Double	%	No Load Current

VecGrp	Integer		Vector Group 1: DD0, 2: DZ0, 3: DZN0, 4: YNY0, 5: YNYN0, 6: YY0, 7: YYN0, 8: ZD0, 9: ZND0, 10: DYN1, 11: DZ1, 12: DZN1, 13: YD1, 14: YND1, 15: YNZN1, 16: YZ1, 17: YZN1, 18: ZD1, 19: ZND1, 20: ZNYN1, 21: ZY1, 22: ZYN1, 23: DY5, 24: DYN5, 25: YD5, 26: YND5, 27: YNZ5, 28: YNZN5, 29: YZ5, 30: YZN5, 31: ZNY5, 32: ZNYN5, 33: ZY5, 34: ZYN5, 35: DD6, 36: DZ6, 37: DZN6, 38: YNY6, 39: YNYN6, 40: YY6, 41: YYN6, 42: ZD6, 43: ZND6, 44: DY7, 45: DYN7, 46: DZ7, 47: DZN7, 48: YD7, 49: YND7, 50: YNZN7, 51: YZ7, 52: YZN7, 53: ZD7, 54: ZND7, 55: ZNYN7, 56: ZY7, 57: ZYN7, 58: DY11, 59: DYN11, 60: YD11, 61: YND11, 62: YNZ11, 63: YNZN11, 64: YZ11, 65: YZN11, 66: ZNY11, 67: ZNYN11, 68: ZY11, 69: ZYN11, 70: DY1, 71: Y0, 72: YN0, 73: D0, 74: ZNY1, 75: ZNY7, 76: DDN0, 77: DND0, 78: DNYN1, 79: DNYN11, 80: YNDN1, 81: YNDN11
uk_ct	Double	%	Ref. Short Circuit Voltage Half Winding
ur_ct	Double	%	SC Voltage – Ohmic Part Half Winding
Nullsystem			
FlagZ0Input	Integer		Zero Data Input 1: Z0/Z1 and R0/X0 2: R0 and X0 3: R0/R1 and X0/X1 4: ZABNL, ZBANL and ZABSC
Z0_Z1	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
R0_R1	Double	pu	Ratio Zero-Phase to Positive-Phase Resistance
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence
X0_X1	Double	pu	Ratio Zero-Phase to Positive-Phase Reactance
R0_X0	Double	pu	Ratio R/X – Zero-Phase Sequence
ZABNL	Double	Ohm	Impedance between A and B in No Load
ZBANL	Double	Ohm	Impedance between B and A in No Load
ZABSC	Double	Ohm	Impedance between B and A in Short Circuit
Reglerdaten			
FlagConNode	Integer		Controller Node 1: Side 1 2: Side
Flag_Roh	Integer		State – Tap Position 1: Fixed 2: Node 3: Impedance 4: Active power 5: Reactive power 6: Control Charact.
Flag_Tap	Integer		Individual Tap Positions 0: No 1: Yes
roh	Double		Present Tap Position
roh1	Double		Present Tap Position Winding 1
roh2	Double		Present Tap Position Winding 2
roh3	Double		Present Tap Position Winding 3
rohl	Double		Minimum Tap Position
rohm	Double		Main Tap Position
rohu	Double		Maximum Tap Position
alpha	Double	°	Additional Voltage Angle
ukr	Double	%	Additional Voltage per Tap Position
phi	Double	°	Voltage Phase Shift per Tap Position
ukl	Double	%	Short Circuit Voltage at Minimum Tap Position

uku	Double	%	Short Circuit Voltage at Maximum Tap Position
ull	Double	%	Voltage Lower Limit
uul	Double	%	Voltage Upper Limit
Plp	Double	MW	Active Power Lower Limit for Controller
Pup	Double	MW	Active Power Upper Limit for Controller
Qlp	Double	Mvar	Reactive Power Lower Limit for Controller
Qup	Double	Mvar	Reactive Power Upper Limit for Controller
Dynamische Daten			
ResFlux1	Double	pu	Residual Flux Phase L1
ResFlux2	Double	pu	Residual Flux Phase L2
ResFlux3	Double	pu	Residual Flux Phase L3

Dreiwicklungstransformator (ThreeWindingTransformer)

Attributname	Datentyp	Einheit	Beschreibung
Un1	Double	kV	Rated Voltage (Side 1)
Un2	Double	kV	Rated Voltage (Side 2)
Un3	Double	kV	Rated Voltage (Side 3)
Sn12	Double	MVA	Rated Apparent Through Power (Side 1-2)
Sn23	Double	MVA	Rated Apparent Through Power (Side 2-3)
Sn31	Double	MVA	Rated Apparent Through Power (Side 3-1)
Smax1	Double	MVA	Full Load Power (Side 1)
Smax2	Double	MVA	Full Load Power (Side 2)
Smax3	Double	MVA	Full Load Power (Side 3)
Smax1_1	Double	MVA	First Additional Full Load Power (Side 1)
Smax1_2	Double	MVA	Second Additional Full Load Power (Side 1)
Smax1_3	Double	MVA	Third Additional Full Load Power (Side 1)
Smax2_1	Double	MVA	First Additional Full Load Power (Side 2)
Smax2_2	Double	MVA	Second Additional Full Load Power (Side 2)
Smax2_3	Double	MVA	Third Additional Full Load Power (Side 2)
Smax3_1	Double	MVA	First Additional Full Load Power (Side 3)
Smax3_2	Double	MVA	Second Additional Full Load Power (Side 3)
Smax3_3	Double	MVA	Third Additional Full Load Power (Side 3)
uk1	Double	%	Reference Short Circuit Voltage (Side 1-2)
uk2	Double	%	Reference Short Circuit Voltage (Side 2-3)
uk3	Double	%	Reference Short Circuit Voltage (Side 3-1)
ur1	Double	%	Ohmic Short Circuit Voltage (Side 1-2)
ur2	Double	%	Ohmic Short Circuit Voltage (Side 2-3)
ur3	Double	%	Ohmic Short Circuit Voltage (Side 3-1)
i0	Double	%	No Load Current
Vfe	Double	kW	Iron Losses
phi1	Double	°	Additional Phase Rotation (Side 1)
phi2	Double	°	Additional Phase Rotation (Side 2)
phi3	Double	°	Additional Phase Rotation (Side 3)
VecGrp1	Integer		Vector Group (Side 1) 1: Y0, 2: YN0, 3: Y6, 4: YN6, 5: D1, 6: D5, 7: D7, 8: D11, 9: Z1, 10: ZN1, 11: Z5, 12: ZN5, 13: Z7, 14: ZN7, 15: Z11, 16: ZN11, 17: ATN, 18: AT

VecGrp2	Integer		Vector Group (Side 2) 1: Y0, 2: YN0, 3: Y6, 4: YN6, 5: D1, 6: D5, 7: D7, 8: D11, 9: Z1, 10: ZN1, 11: Z5, 12: ZN5, 13: Z7, 14: ZN7, 15: Z11, 16: ZN11, 17: ATN, 18: AT
VecGrp3	Integer		Vector Group (Side 3) 1: Y0, 2: YN0, 3: Y6, 4: YN6, 5: D1, 6: D5, 7: D7, 8: D11, 9: Z1, 10: ZN1, 11: Z5, 12: ZN5, 13: Z7, 14: ZN7, 15: Z11, 16: ZN11
Nullsystem			
FlagZ0Input	Integer		Input 1: Z0/Z1 and R0/X0 2: R0 and X0 3: R0/R1 and X0/X1
Z0_Z1_12	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
Z0_Z1_23	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
Z0_Z1_31	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
R0_R1_12	Double	pu	Ratio Zero-Phase to Positive-Phase Resistance
R0_R1_23	Double	pu	Ratio Zero-Phase to Positive-Phase Resistance
R0_R1_31	Double	pu	Ratio Zero-Phase to Positive-Phase Resistance
R0_12	Double	Ohm	Resistance – Zero-Phase Sequence
R0_23	Double	Ohm	Resistance – Zero-Phase Sequence
R0_31	Double	Ohm	Resistance – Zero-Phase Sequence
X0_12	Double	Ohm	Reactance – Zero-Phase Sequence
X0_23	Double	Ohm	Reactance – Zero-Phase Sequence
X0_31	Double	Ohm	Reactance – Zero-Phase Sequence
X0_X1_12	Double	pu	Ratio Zero-Phase to Positive-Phase Reactance
X0_X1_23	Double	pu	Ratio Zero-Phase to Positive-Phase Reactance
X0_X1_31	Double	pu	Ratio Zero-Phase to Positive-Phase Reactance
R0_X0_12	Double	pu	Ratio R/X – Zero-Phase Sequence
R0_X0_23	Double	pu	Ratio R/X – Zero-Phase Sequence
R0_X0_31	Double	pu	Ratio R/X – Zero-Phase Sequence
Reglerdaten			
Flag_Roh1	Integer		State – Tap Position (Side 1) 0: None 1: Fixed 2: Node 3: Impedance 4: Active power 5: Reactive power
Flag_Roh2	Integer		State – Tap Position (Side 2) 0: None 1: Fixed 2: Node 3: Impedance 4: Active power 5: Reactive power
Flag_Roh3	Integer		State – Tap Position (Side 3) 0: None 1: Fixed 2: Node 3: Impedance 4: Active power 5: Reactive power
roh1	Double		Present Tap Position (Side 1)
roh2	Double		Present Tap Position (Side 2)
roh3	Double		Present Tap Position (Side 3)
rohu1	Double		Maximum Tap Position (Side 1)
rohu2	Double		Maximum Tap Position (Side 2)

rohu3	Double		Maximum Tap Position (Side 3)
rohm1	Double		Main Tap Position (Side 1)
rohm2	Double		Main Tap Position (Side 2)
rohm3	Double		Main Tap Position (Side 3)
rohl1	Double		Minimum Tap Position (Side 1)
rohl2	Double		Minimum Tap Position (Side 2)
rohl3	Double		Minimum Tap Position (Side 3)
Dynamische Daten			
ResFlux1	Double	pu	Residual Flux Phase L1
ResFlux2	Double	pu	Residual Flux Phase L2
ResFlux3	Double	pu	Residual Flux Phase L3

Leitung (Line)

Attributname	Datentyp	Einheit	Beschreibung
Flag_LineTyp	Integer		Line Type 1: Cable 2: Overhead line 3: Connector
FlagMat	Integer		Line Material 1: Al 2: Cu
Len	Double	km	Length
ParSys	Double	1	Number of Parallel Systems
R	Double	Ohm/km	Resistance
X	Double	Ohm/km	Reactance
C	Double	nF/km	Capacitance
va	Double	kW/km	Leakage Losses to Ground
Ith	Double	kA	Thermal Limit Current
Ith1	Double	kA	First Additional Limit Current
Ith2	Double	kA	Second Additional Limit Current
Ith3	Double	kA	Third Additional Limit Current
FrqNenn	Double	Hz	Rated Frequency
alpha	Double	1/°C	Temperature Coefficient for Temperature Dependent Resistance Change
Nullsystem			
Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: X0/X1 and R0/R1 2: r0 and x0
R0_R1	Double	pu	Ratio Zero-Phase to Positive-Phase Resistance
X0_X1	Double	pu	Ratio Zero-Phase to Positive-Phase Reactance
R0	Double	Ohm/km	Resistance – Zero-Phase Sequence
X0	Double	Ohm/km	Reactance – Zero-Phase Sequence
C0	Double	nF/km	Capacitance in Zero-Phase Sequence
rR	Double	Ohm/km	Resistance – Return Conductor
xR	Double	Ohm/km	Reactance – Return Conductor

Variables Längselement (VarSerialElement)

Attributname	Datentyp	Einheit	Beschreibung
Flag_LF	Integer		Load Flow Input 1: Impedance 2: Model
Flag_Macro_LF	Integer		Model Type Load Flow 0: None 1: Controller 2: Equivalent circuit
Ur1	Double	kV	Rated Voltage Side 1
Ur2	Double	kV	Rated Voltage Side 2
R12lf	Double	Ohm	Resistance Load Flow
X12lf	Double	Ohm	Reactance Load Flow
R21lf	Double	Ohm	Resistance Load Flow
X21lf	Double	Ohm	Reactance Load Flow
R12sc	Double	Ohm	Resistance Short Circuit
X12sc	Double	Ohm	Reactance Short Circuit
R21sc	Double	Ohm	Resistance Short Circuit
X21sc	Double	Ohm	Reactance Short Circuit
Nullsystem			
Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: Z0/Z1 and R0/X0 2: R0 and X0
R0_X0	Double	pu	Ratio R/X – Zero-Phase Sequence
Z0_Z1	Double	pu	Ratio Zero-Phase to Positive-Phase Sequence Impedance
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence
Dynamische Daten			
Flag_Macro_SC	Integer		Impedances for Dynamics 1: Load flow 2: Short circuit
Oberschwingungen			
Flag_Har	Integer		State – Harmonics 0: No frequency dependency 1: Quality – R constant 2: Quality – X/R constant 3: Impedance characteristic
qr	Double	1	Quality – R Constant
ql	Double	1	Quality – X/R Constant

Längsdrossel (SerialReactor)

Attributname	Datentyp	Einheit	Beschreibung
Flag_CoInput	Integer		Input Data 1: Reference coil voltage 2: Inductance
uD	Double	%	Reference Coil Voltage
L	Double	mH	Inductance
Un	Double	kV	Rated Voltage
InD	Double	kA	Rated Current

Ith1	Double	kA	First Additional Limit Current
Ith2	Double	kA	Second Additional Limit Current
Ith3	Double	kA	Third Additional Limit Current
Nullsystem			
R_X	Double		Ratio R/X – Positive-Phase Sequence
Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: R0/R1 and X0/X1 2: R0 and X0 3: R0 and L0
X0_X1	Double		Ratio Zero-Phase to Positive-Phase Reactance
R0_R1	Double		Ratio Zero-Phase to Positive-Phase Resistance
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence
L0	Double	mH	Inductance in Zero-Phase Sequence
Dynamische Daten			
ResFlux1	Double	pu	Residual Flux Phase L1
ResFlux2	Double	pu	Residual Flux Phase L2
ResFlux3	Double	pu	Residual Flux Phase L3

Längskondensator (SerialCondensator)

Attributname	Datentyp	Einheit	Beschreibung
C	Double	nF/km	Capacitance
XC	Double	Ohm	Capacitive Reactance
Un	Double	kV	Rated Voltage
R_X	Double	pu	Ratio R/X – Positive-Phase Sequence
Smax	Double	MVA	Full Load Power
Smax1	Double	MVA	First Additional Full Load Power
Smax2	Double	MVA	Second Additional Full Load Power
Smax3	Double	MVA	Third Additional Full Load Power
Nullsystem			
Flag_Z0Input	Integer		Zero-Phase Sequence Input Data 1: R0/R1 and X0/X1 2: R0 and X0 3: R0 and C0
X0_X1	Double		Ratio Zero-Phase to Positive-Phase Reactance
R0_R1	Double		Ratio Zero-Phase to Positive-Phase Resistance
R0	Double	Ohm	Resistance – Zero-Phase Sequence
X0	Double	Ohm	Reactance – Zero-Phase Sequence

Längs Oberschwingungs-Resonanznetz (HarBranchResNet)

Attributname	Datentyp	Einheit	Beschreibung
Un	Double	kV	Rated Voltage
R1	Double	Ohm	Resistance at Network Frequency
X1	Double	Ohm	Reactance at Network Frequency

Impedance	Integer		Determine Impedance 1: Vmax 2: Imax
RCData	Integer		Ripple Control Impedance 0: No 1: Yes
R1rc	Double	Ohm	Resistance at Ripple Control Frequency
X1rc	Double	Ohm	Reactance at Ripple Control Frequency
FlagZ0	Integer		Input Data Zero-Phase Sequence System 1: Blocking 2: Z0 identical Z1 3: R0/R1 and X0/X1 4: R0 and X0
R0	Double	Ohm	Resistance Zero-Phase Sequence System
X0	Double	Ohm	Reactance Zero-Phase Sequence System
R0_R1	Double	1	Ratio Zero-Phase to Positive-Phase Resistance
X0_X1	Double	1	Ratio Zero-Phase to Positive-Phase Reactance

Einbauort des Schutzgerätes (ProtLocation)

Attributname	Datentyp	Einheit	Beschreibung
Flag_State	Integer		Active 0: Off 1: On

Fehleruntersuchung (ProtOCFault)

Attributname	Datentyp	Einheit	Beschreibung
Node_ID	Integer		Fault Location Node
Element_ID	Integer		Fault Location Branch element
Flag_FaultPhase	Integer		Faulty Phases 1: L1 2: L2 3: L3 4: L23 5: L31 6: L12 7: L123

Flag_FaultType	Integer		Fault Type – Node: 1: GC1 2: GC2 3: GC3 4: SC2 5: SC3 Fault Type – Element: 1: I1 2: I2 3: I3 4: I1GC1 5: I2GC2 6: I3GC3 7: I3GC1 8: I3GC2 9: I3SC2 10: I3SC3 11: I2SC2 12: GC1 13: GC2 14: GC3 15: SC2 16: SC3
len	Double	%	Distance
Flag_State	Integer		Operating State 0: Off 1: On

4.2.4 Attribute der Berechnungsobjekte für Strömungsnetze

Wassernetze

Berechnungsparameter (FlowCalcParameter)

Attributname	Datentyp	Einheit	Beschreibung
ITmax	Integer		Maximum Number of Iterations (non-linear)
ITmax2	Integer		Maximum Number of Iterations (linear)
MeshAccuracy	Double	bar	Mesh Accuracy
NodeAccuracy	Double	l/s	Node Accuracy
FlowStep	Double	l/s	Maximum Step for Flow
Flag_Operate	Integer		Check Operating Conditions 0: Warning 1: Error
fCharCurve	Double	pu	Characteristic Curve Factor
SpecDensity	Double	kg/m³	Specific Density
KinematicVis	Double	mm²/s	Kinematic Viscosity
Flag_Pump	Integer		Parallel Pumps 0: No 1: Yes
Flag_Result	Integer		Store Results in Database 0: None 1: All 2: Restricted elements only 3: All elements in case of restrictions
StartTime	Double	s	Starting Time
Duration	Double	s	Duration
TimeStepGeo	Double	s	Time Step Geo-stationary

Netzebene (FlowVoltageLevel)

Attributname	Datentyp	Einheit	Beschreibung
pRated	Double	bar	Rated Pressure
vMax	Double	m/s	Maximum Flow Velocity
pMin	Double	bar	Minimum Operating Pressure
pMax	Double	bar	Maximum Operating Pressure

Netzbereich (FlowNetworkGroup)

Attributname	Datentyp	Einheit	Beschreibung
Flag_MarkedForCaused	Integer		Marked for Caused Malfunction 0: No 1: Yes
Flag_Malfunc	Integer		Malfunction 0: None 1: All elements 2: All lines 3: All restricted elements 4: All restricted lines
Speed_BaseLimit	Double	m/s	Base Speed Limit
Flag_CausedMalfunc	Integer		Caused Malfunction 0: None 1: Marked areas 2: Own area
Flag_CausedElem	Integer		Caused Elements 1: Restricted elements 2: Restricted lines
Speed_CausedLimit	Double	m/s	Caused Speed Limit
Flag_Report	Integer		Reporting 0: None 1: Elements and nodes 2: Lines and nodes 3: Elements 4: Lines 5: Nodes
Flag_FireWater	Integer		Join Fire Water Simulation 0: No 1: Yes
ConLineLength	Double	m	Length
ConLineDiameter	Double	mm	Diameter
ConLineRoughness	Double	mm	Sand Roughness
ConLineZeta	Double		Loss Factor Zeta Value
dsh	Double	m	Delta Elevation
QFireWater	Double	l/s	Fire Water Flow
pFireWater	Double	bar	Fire Water Pressure
tFireWater	Double	h	Fire Water Time
pRelMinLimit	Double	bar	Minimum Pressure – Relative

Konst. Druckabfall/Konst. Fluss (FlowConstLine)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Typ	Integer		Line Type 1: Constant pressure drop 2: Constant flow
PressureDecr	Double	bar	Pressure Drop

Verbraucher (FlowConsumer)

Attributname	Datentyp	Einheit	Beschreibung
Q	Double	l/s	Const. Consumption
Flag_ConControl	Integer		Pressure Dependent Consumption Decrease 0: No 1: Yes
pDiffMin	Double	bar	Minimum Pressure Difference
pRelMin	Double	bar	Minimum Relative Pressure

Druckregler (FlowPressureReg)

Attributname	Datentyp	Einheit	Beschreibung
pInlet	Double	bar	Max. Pressure Deviation
pOutlet	Double	bar	Pressure at Outlet Node
pDevation	Double	bar	Pressure at Inlet Node
Flag_PessInc	Integer		Function 1: Pressure increase 2: Pressure drop 3: Pressure increase and drop

Druckverstärkerpumpe (FlowPumpLine)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Type	Integer		Pump Type 1: Centrifugal pump 2: Reciprocating pump
QOutput	Double	l/s	Output Flow
uPump	Double	1/min	Characteristic Pump Speed
FlowStep	Double	l/s	Maximum Flow

Schieber/Rückschlagventil (FlowValve)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Type	Integer		Valve Type 1: Sliding valve 2: Non-return valve
Opening	Double	%	Degree of Opening
Diameter	Double	mm	Valve Diameter

Pos	Integer		Valve Position 0: Close 1: Open

Leck (FlowLeakage)

Attributname	Datentyp	Einheit	Beschreibung
OutputSurface	Double	mm²	Output Surface
fFlow	Double	pu	Flow Number
FlowStep	Double	l/s	Maximum Step for Flow
ConLineLength	Double	m	Connection Line Length
ConLineDiameter	Double	mm	Connection Line Diameter
ConLineRoughness	Double	mm	Connection Line Sand Roughness
ConLineZeta	Double		Connection Line Sand Zeta Value
dsh	Double	m	Delta Elevation
QFireWater	Double	l/s	Fire Water Flow
pFireWater	Double	bar	Fire Water Pressure
tFireWater	Double	h	Fire Water Time

Druckbuffer (FlowPressureBuffer)

Attributname	Datentyp	Einheit	Beschreibung
PMax	Double	bar	Maximum Pressure

Pumpeinspeisung (FlowPump)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Type	Integer		Pump Type 1: Centrifugal pump 2: Reciprocating pump
QOutput	Double	l/s	Output Flow
uPump	Double	1/min	Characteristic Pump Speed
FlowStep	Double	l/s	Maximum Flow
Flag_Limits	Integer		Limit Type 0: None 1: Flow
QOutputmin	Double	l/s	Minimum Output Flow
QOutputmax	Double	l/s	Maximum Output Flow

Hochbehälter (FlowWaterTower)

Attributname	Datentyp	Einheit	Beschreibung
hWaterLevel	Double	m	Water Level
hFillStart1	Double	m	Filling Level 1 Start
hFillStop1	Double	m	Filling Level 1 Stop
uPump1	Double	1/min	Pump Characteristics 1
hFillStart2	Double	m	Filling Level 1 Start

hFillStop2	Double	m	Filling Level 1 Stop
uPump2	Double	1/min	Pump Characteristics 1
hFillStart3	Double	m	Filling Level 1 Start
hFillStop3	Double	m	Filling Level 1 Stop
uPump3	Double	1/min	Pump Characteristics 1
Flag_Level	Integer		Level Data 0: No 1: Yes
Flag_Limits	Integer		Limit Type 0: None 1: Flow
Qmin	Double	l/s	Minimum Flow
Qmax	Double	l/s	Maximum Flow

Leitung (FlowLine)

Attributname	Datentyp	Einheit	Beschreibung
LineLength	Double	m	Length
Diameter	Double	mm	Diameter
SandRoughness	Double	mm	Sand Roughness
fLength	Double	%	Length Allowance Factor
fCurve	Double		Curve Factor
fDiameterAn	Double	%	Annual Diameter Reduction
fRoughnessAn	Double	%	Annual Roughness Increase
Zeta	Double		Zeta Value
LeakageRate	Double	l/sm	Leakage Rate

Knoten (FlowNode)

Attributname	Datentyp	Einheit	Beschreibung
Sh	Double	m	Elevation

Gasnetze

Berechnungsparameter (FlowCalcParameter)

Attributname	Datentyp	Einheit	Beschreibung
ITmax	Integer		Maximum Number of Iterations (non-linear)
ITmax2	Integer		Maximum Number of Iterations (linear)
MeshAccuracy	Double	bar	Mesh Accuracy
NodeAccuracyG	Double	m³/h	Node Accuracy
FlowStepG	Double	m³/h	Maximum Step for Flow
Flag_Operate	Integer		Check Operating Conditions 0: Warning 1: Error
SpecDensity	Double	kg/m³	Specific Density
HeatingAmount	Double	MJ/kg	Heating Amount
pAir	Double	bar	Air Pressure

SutherlandConst	Double	K	Sutherland Constant
AdiabaticExp	Double		Adiabatic Exponent
fConst	Double		Constant Factor
fLinear	Double		Linear Factor
Flag_Result	Integer		Store Results in Database 0: None 1: All 2: Restricted elements only 3: All elements in case of restrictions
StartTime	Double	s	Starting Time
Duration	Double	s	Duration
TimeStepGeo	Double	s	Time Step Geo-stationary

Netzebene (FlowVoltageLevel)

Attributname	Datentyp	Einheit	Beschreibung
pRated	Double	bar	Rated Pressure
TGas	Double	°C	Gas Temperature
TAir	Double	°C	Air Temperature
vMax	Double	m/s	Maximum Flow Velocity
pMin	Double	bar	Minimum Operating Pressure
pMax	Double	bar	Maximum Operating Pressure

Netzbereich (FlowNetworkGroup)

Attributname	Datentyp	Einheit	Beschreibung
Flag_MarkedForCaused	Integer		Marked for Caused Malfunction 0: No 1: Yes
Flag_Malfunc	Integer		Malfunction 0: None 1: All elements 2: All lines 3: All restricted elements 4: All restricted lines
Speed_BaseLimit	Double	m/s	Base Speed Limit
Flag_CausedMalfunc	Integer		Caused Malfunction 0: None 1: Marked areas 2: Own area
Flag_CausedElem	Integer		Caused Elements 1: Restricted elements 2: Restricted lines
Speed_CausedLimit	Double	m/s	Caused Speed Limit
Flag_Report	Integer		Reporting 0: None 1: Elements and nodes 2: Lines and nodes 3: Elements 4: Lines 5: Nodes

Kompressor (FlowCompressor)

Attributname	Datentyp	Einheit	Beschreibung
pInlet	Double	bar	Max. Pressure Deviation
pOutlet	Double	bar	Pressure at Outlet Node
pDevation	Double	bar	Pressure at Inlet Node

Konst. Druckabfall/Konst. Fluss (FlowConstLine)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Typ	Integer		Line Type 1: Constant pressure drop 2: Constant flow
PressureDecr	Double	bar	Pressure Drop
FlowGas	Double	mN³/h	Flow

Verbraucher (FlowConsumer)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Q	Integer		Consumption Type 1: Standard 2: Operating Conditions 3: Power
Q1	Double	m³/h	Constant Consumption – Standard
Q2	Double	m³/h	Constant Consumption – Operating Cond.
Q3	Double	MW	Constant Consumption – Power
pDiffMin	Double	bar	Minimum Pressure Difference
pRelMin	Double	bar	Minimum Relative Pressure

Druckregler (FlowPressureReg)

Attributname	Datentyp	Einheit	Beschreibung
pInlet	Double	bar	Max. Pressure Deviation
pOutlet	Double	bar	Pressure at Outlet Node
pDevation	Double	bar	Pressure at Inlet Node
Flag_PessInc	Integer		Function 1: Pressure increase 2: Pressure drop 3: Pressure increase and drop
QReturn	Double	m³/h	Maximum Return Flow

Schieber/Rückschlagventil (FlowValve)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Type	Integer		Valve Type 1: Sliding valve 2: Non-return valve

Pos	Integer		Valve Position 0: Close 1: Open

Leck (FlowLeakage)

Attributname	Datentyp	Einheit	Beschreibung
OutputSurface	Double	mm²	Output Surface
fFlow	Double	pu	Flow Number
FlowStpG	Double	m³/h	Maximum Step for Flow

Einspeisung Gas (FlowInfeederG)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Typ	Integer		Infeeder Type 1: Pressure supply 2: Flow supply
QReturn	Double	m³/h	Maximum Return Flow
pConst	Double	bar	Constant Excess Pressure
FlagQ	Integer		Flow Supply Type 1: Flow supply 2: Operating Conditions 3: Power
Q1	Double	m³/h	Constant Supply – Standard
Q2	Double	m³/h	Constant Supply – Operating Condition
Q3	Double	MW	Constant Supply – Power
Flag_Limits	Integer		Limit Type 0: None 1: Flow
Qmin	Double		Minimum Supply
Qmax	Double		Maximum Supply

Druckbuffer (FlowPressureBuffer)

Attributname	Datentyp	Einheit	Beschreibung
PMax	Double	bar	Maximum Pressure

Leitung (FlowLine)

Attributname	Datentyp	Einheit	Beschreibung
LineLength	Double	m	Length
Diameter	Double	mm	Diameter
SandRoughness	Double	mm	Sand Roughness
fLength	Double	%	Length Allowance Factor
fCurve	Double		Curve Factor
fDiameterAn	Double	%	Annual Diameter Reduction
fRoughnessAn	Double	%	Annual Roughness Increase
Zeta	Double		Zeta Value

Knoten (FlowNode)

Attributname	Datentyp	Einheit	Beschreibung
Sh	Double	m	Elevation
Pres	Double	bar	Pressure Reservation

Wärme-/Kältenetze

Berechnungsparameter (FlowCalcParameter)

Attributname	Datentyp	Einheit	Beschreibung
ITmax	Integer		Maximum Number of Iterations (non-linear)
ITmax2	Integer		Maximum Number of Iterations (linear)
MeshAccuracy	Double	bar	Mesh Accuracy
NodeAccuracy	Double	l/s	Node Accuracy
FlowStep	Double	l/s	Maximum Step for Flow
Flag_Operate	Integer		Check Operating Conditions 0: Warning 1: Error
qSpec	Double	J/kgK	Specific Thermal Capacity
Flag_MalFunc	Integer		Circuit for Malfunction 1: Supply line 2: Return line 3: Supply and return line
Flag_Result	Integer		Store Results in Database 0: None 1: All 2: Restricted elements only 3: All elements in case of restrictions
StartTime	Double	s	Starting Time
Duration	Double	s	Duration
TimeStepGeo	Double	s	Time Step Geo-stationary

Netzebene (FlowVoltageLevel)

Attributname	Datentyp	Einheit	Beschreibung
pRated	Double	bar	Rated Pressure
TRated	Double	°C	Rated Temperature
TAir	Double	°C	Air Temperature
vMax	Double	m/s	Maximum Flow Velocity
pMin	Double	bar	Minimum Operating Pressure Supply Line
pMax	Double	bar	Maximum Operating Pressure Supply Line
TSupplyLine	Double	°C	Temperature Supply Line
pMinR	Double	bar	Minimum Operating Pressure Return Line
pMaxR	Double	bar	Maximum Operating Pressure Return Line
TReturnLine	Double	°C	Temperature Return Line

Netzbereich (FlowNetworkGroup)

Attributname	Datentyp	Einheit	Beschreibung
Flag_MarkedForCaused	Integer		Marked for Caused Malfunction 0: No 1: Yes
Flag_Malfunc	Integer		Malfunction 0: None 1: All elements 2: All lines 3: All restricted elements 4: All restricted lines
Speed_BaseLimit	Double	m/s	Base Speed Limit
Flag_CausedMalfunc	Integer		Caused Malfunction 0: None 1: Marked areas 2: Own area
Flag_CausedElem	Integer		Caused Elements 1: Restricted elements 2: Restricted lines
Speed_CausedLimit	Double	m/s	Caused Speed Limit
Flag_Report	Integer		Reporting 0: None 1: Elements and nodes 2: Lines and nodes 3: Elements 4: Lines 5: Nodes

Konst. Druckabfall/Konst. Fluss (FlowConstLine)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Typ	Integer		Line Type 1: Constant pressure drop 2: Constant flow
PressureDecr	Double	bar	Pressure Drop
FlowHeating	Double	t/h	Flow

Verbraucher (FlowConsumer)

Attributname	Datentyp	Einheit	Beschreibung
Q	Double	l/s	Const. Consumption
Flag_ConTyp	Integer		Consumption Type 1: Constant consumption 2: Constant power consumption 3: Sum of consumption and power
Q3	Double	MW	Constant Consumption – Power
Q4	Double	t/h	Constant Consumption
Power	Double	MW	Constant Consumption – Power
Flag_ConControl	Integer		Pressure Dependent Consumption Decrease 0: No 1: Yes
pDiffMin	Double	bar	Minimum Pressure Difference
Flag_Temp	Integer		Temperature Type 1: Return temperature 2: Difference of temperature

T	Double	°C	Temperature

Wärmetauscher (FlowHeatExchanger)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Typ	Integer		Heat Exchanger Type 1: Hydraulic uncoupling 2: Power apply
Flag_ConControl	Integer		Primary Pressure Dependent Consumption Decrease 0: No 1: Yes
Flag_Temp	Integer		Temperature Type 1: Return temperature 2: Difference of temperature sup – ret 3: Difference of temperature sec – prim
Flag_Maint	Integer		Pressure Maintenance Type 1: Medium pressure, difference and parts 2: Supply pressure and difference 3: Return pressure and difference 4: Pump data and parts 5: Supply pressure and pump data 6: Return pressure and pump data
Flag_Master	Integer		Leading Supply 0: No 1: Yes
Efficiency	Double	%	Efficiency
pDiffMin	Double	bar	Primary Minimum Pressure Difference
tPrim	Double	°C	Primary Temperature
tFeed	Double	°C	Secondary Supply Temperature
uPump	Double	1/min	Characteristic Pump Speed
FlowStep	Double	l/s	Maximum Step for Flow
Power	Double	MW	Power
pMedium	Double	bar	Medium Pressure
pSupRet	Double	bar	Difference Pressure
pSupplyMaint	Double	bar	Supply Pressure
pReturnMaint	Double	bar	Return Pressure
SupplyPart	Double	%	Part – Supply Pressure
ReturnPart	Double	%	Part – Return Pressure
QOutput	Double	l/s	Output Flow

Druckregler (FlowPressureReg)

Attributname	Datentyp	Einheit	Beschreibung
pInlet	Double	bar	Max. Pressure Deviation
pOutlet	Double	bar	Pressure at Outlet Node
pDevation	Double	bar	Pressure at Inlet Node
Flag_PessInc	Integer		Function 1: Pressure increase 2: Pressure drop 3: Pressure increase and drop
Flag_PressDif	Integer		Difference Pressure Regulator 0: No 1: Yes
pSupRet	Double	bar	Difference Pressure

Druckverstärkerpumpe (FlowPumpLine)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Type	Integer		Pump Type 1: Centrifugal pump 2: Reciprocating pump
QOutput	Double	l/s	Output Flow
uPump	Double	1/min	Characteristic Pump Speed
FlowStep	Double	l/s	Maximum Flow

Schieber/Rückschlagventil (FlowValve)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Type	Integer		Valve Type 1: Sliding valve 2: Non-return valve
Opening	Double	%	Degree of Opening
Diameter	Double	mm	Valve Diameter
Pos	Integer		Valve Position 0: Close 1: Open

Leck (FlowLeakage)

Attributname	Datentyp	Einheit	Beschreibung
OutputSurface	Double	mm²	Output Surface
fFlow	Double	pu	Flow Number
FlowStep	Double	l/s	Maximum Step for Flow

Temperaturregler (FlowThermoReg)

Attributname	Datentyp	Einheit	Beschreibung
tMin	Double	°C	Minimum Temperature
tMax	Double	°C	Maximum Temperature
TempAccuracy	Double	°C	Temperature Accuracy
FlowStep	Double	t/h	Maximum Step for Flow

Einspeisung Wärme/Kälte (FlowInfeederH)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Typ	Integer		Infeeder Type 1: Pressure supply 2: Power Supply 3: Pressure maintenance

Flag_SupTyp	Integer		Power Supply Type 1: Constant supply 2: Constant supply power
Flag_ConControl	Integer		Pressure Dependent Supply Decrease 0: No 1: Yes
Flag_T	Integer		Temperature Type 1: Supply temperature 2: Difference of temperature
Flag_Maint	Integer		Pressure Maintenance Type 1: Medium pressure, difference and parts 2: Supply pressure and difference 3: Return pressure and difference 4: Pump data and parts 5: Supply pressure and pump data 6: Return pressure and pump data
Flag_Master	Integer		Leading Supply 0: No 1: Yes
T	Double	°C	Temperature
pDiffMin	Double	bar	Minimum Pressure Difference
uPump	Double	1/min	Characteristic Pump Speed
FlowStep	Double	l/s	Maximum Step for Flow
pSupply	Double	bar	Pressure Supply
Q	Double	t/h	Constant Supply Volume
Power	Double	MW	Constant Power Supply
pMedium	Double	bar	Medium Pressure
pSupRet	Double	bar	Difference Pressure
QOutput	Double	l/s	Output Flow
pSupplyMain	Double	bar	Supply Pressure
pReturnMain	Double	bar	Return Pressure
SupplyPart	Double	%	Part – Supply Pressure
ReturnPart	Double	%	Part – Return Pressure
Flag_Limits	Integer		Limit Type 0: None 1: Flow 2: Power
Qmin	Double	t/h	Minimum Flow
Qmax	Double	t/h	Maximum Flow
Pmin	Double	MW	Minimum Power
Pmax	Double	MW	Maximum Power

Druckbuffer (FlowPressureBuffer)

Attributname	Datentyp	Einheit	Beschreibung
PMax	Double	bar	Maximum Pressure

Pumpeinspeisung (FlowPump)

Attributname	Datentyp	Einheit	Beschreibung
Flag_Type	Integer		Pump Type 1: Centrifugal pump 2: Reciprocating pump
QOutput	Double	l/s	Output Flow

uPump	Double	1/min	Characteristic Pump Speed
tSupply	Double	°C	Supply Temperature
FlowStep	Double	l/s	Maximum Flow
Flag_Limits	Integer		Limit Type 0: None 1: Flow
QOutputmin	Double	l/s	Minimum Output Flow
QOutputmax	Qmax	l/s	Maximum Output Flow

Leitung (FlowLine)

Attributname	Datentyp	Einheit	Beschreibung
LineLength	Double	m	Length
Diameter	Double	mm	Diameter
SandRoughness	Double	mm	Sand Roughness
fLength	Double	%	Length Allowance Factor
fCurve	Double		Curve Factor
Zeta	Double		Zeta Value
LeakageRate	Double	l/sm	Leakage Rate
HeatingCond	Double	W/Mk	Thermal Conductivity

Knoten (FlowNode)

Attributname	Datentyp	Einheit	Beschreibung
Sh	Double	m	Elevation
PDiffMin	Double	bar	Minimum Pressure Difference

5 Referenz

5.1 Dokumentation

Die vollständige Dokumentation von PSS SINCAL ist als Online-Hilfe verfügbar. Zusätzlich ist die Dokumentation auch in Form von PDF Dateien auf der Installations-DVD im Verzeichnis "Doc\German\Sincal" enthalten.

Benutzeroberfläche

Eine umfassende Beschreibung aller Funktionen der PSS SINCAL Benutzeroberfläche ist im Handbuch **Bedienung** verfügbar.

Zum Einstieg sollte hier vor allem das Kapitel **Netzbearbeitung anhand eines Beispiels** gelesen werden. Dies stellt Schritt für Schritt das Erfassen, die Bearbeitung, das Berechnen und die Ergebnisvisualisierung eines Elektronetzes dar.

Simulationsverfahren

Die **Fachhandbücher für Elektronetze** beinhalten detaillierte Beschreibungen der verschiedenen Berechnungsverfahren für Elektronetze sowie deren Eingabedaten. Diese Handbücher bestehen aus zwei Teilen:

- Eingabedaten
  Beschreibung der Eingabedaten für alle Simulationsverfahren.
- Fachhandbuch für das jeweilige Simulationsverfahren
  Hier sind erweiterte Beschreibungen zu den Simulationsverfahren verfügbar. Die Handbücher tragen den Titel des jeweiligen Verfahrens: Lastfluss, Kurzschluss, usw.

Die **Fachhandbücher für Strömungsnetze** beinhalten detaillierte Beschreibungen der verschiedenen Berechnungsverfahren für Strömungsnetze sowie deren Eingabedaten. Folgende Handbücher sind verfügbar: Wasser, Gas und Wärme/Kälte.

Netzdatenbank

Detaillierte Informationen über den Aufbau der Netzdatenbank sind im Handbuch **Datenbankbeschreibung** verfügbar. Dieses Handbuch enthält eine vollständige Darstellung des PSS SINCAL Datenmodells. Es werden sowohl Aufbau und Semantik des Datenmodells erläutert, als auch die Tabellen (Relationen) mit ihren Attributen detailliert dargestellt.

5.2 PSS SINCAL Architektur

Das folgende Bild zeigt die grundlegende Architektur von PSS SINCAL.

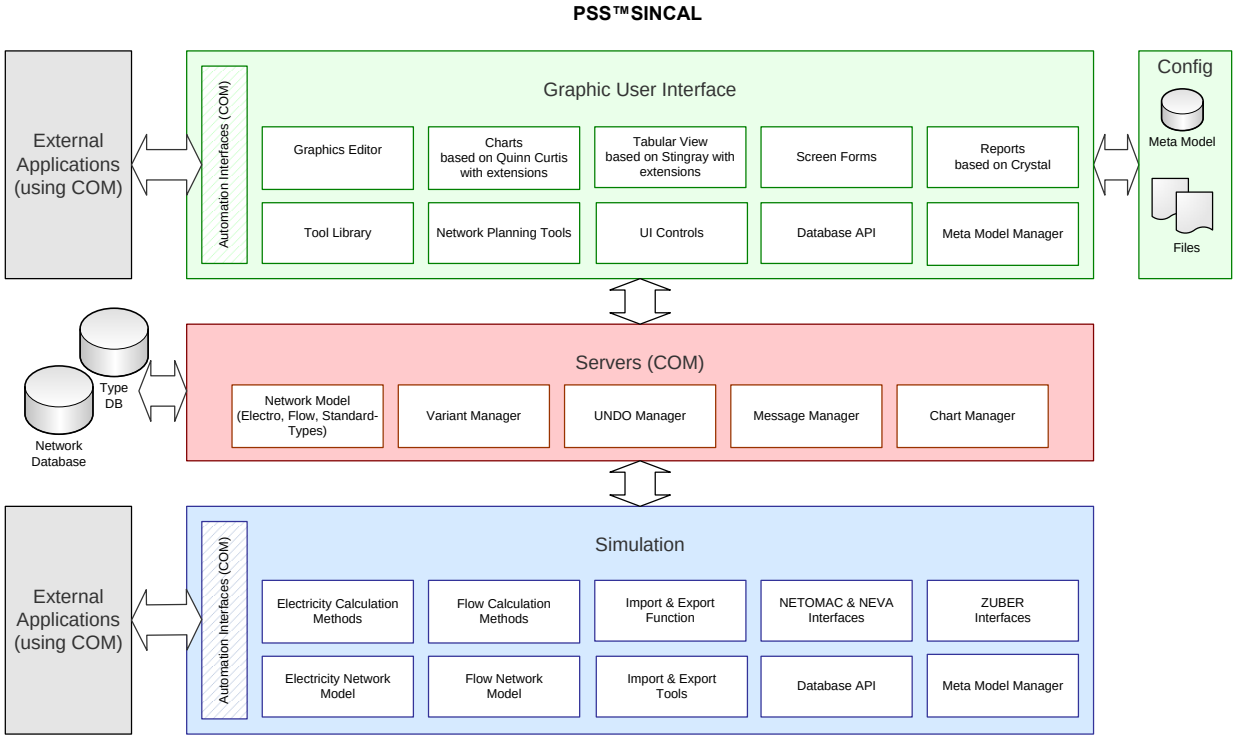


Bild: PSS SINCAL Architektur

Wie aus der Darstellung ersichtlich ist, gibt es hier zwei wichtige Komponenten:

- **Graphic User Interface**
  Dies ist die eigentliche Benutzeroberfläche von PSS SINCAL. Mit dieser erfolgen die Bearbeitung von Netzen, die Datenmodifikation, die Ergebnisauswertung, usw.
- **Simulation**
  Diese Komponente beinhaltet alle in PSS SINCAL verfügbaren Berechnungsmethoden. Die Berechnungskomponente kann auch Stand-alone (also ohne die Benutzeroberfläche) verwendet werden.

Dazwischen befindet sich die Komponente **Servers (COM)**, über die der interne Datenaustausch erfolgt. Der komplette Datenbankzugriff erfolgt ebenfalls mit dieser Komponente.

COM-Interfaces

Alle Komponenten besitzen COM-Interfaces. Dabei wird zwischen internen und externen COM-Interfaces unterschieden. Die internen Interfaces werden nur innerhalb von PSS SINCAL verwendet. Diese sind nicht dokumentiert und auch nicht zur externen Nutzung konzipiert. Die externen Interfaces sind dokumentiert und können auch in eigenen Anwendungen beliebig verwendet werden. Damit können beispielsweise Arbeitsabläufe automatisiert oder die Berechnungsmethoden in eigene Anwendungen integriert werden.

5.3 Vorgefertigte Kopplungslösungen

Speziell im Bereich der GIS Kopplung bieten einige Partnerunternehmen bereits vorgefertigte Lösungen an, die von verschiedensten PSS SINCAL Kunden bereits erfolgreich eingesetzt werden.

L&Mark Informatika Kft.

L&MARK Informatika Kft. ist Partner bei der Realisierung von PSS SINCAL GIS Anbindungen weltweit. Die Firma ist Partner von AED-SICAD und hat mehrere Lösungseinsätze auf Basis von diversen GIS Plattformen (ESRI ArcFM, ArcFM-UT, Sicad/open, SICAD-UTE, Intergraph GNET, usw.).

L&MARK Informatika Kft.
Frau Andrea Lisziewicz
Margit krt. 43-45. 4/5.
1024 Budapest
Ungarn
Fon +36 1 201 7725
Fax +36 1 201 2817
e-mail andrea.lisziewicz@lmark.hu
http://www.lmark.hu

Mettenmeier GmbH

Die Mettenmeier GmbH gehört zu den marktführenden Dienstleistern für die Energie- und Wasserwirtschaft. Auf Basis der starken Position im deutschen Markt werden europaweit Kunden mit innovativen Asset-Management-Lösungen und hochwertigen Dienstleistungen im Bereich der Geoinformation betreut.

Mettenmeier stellt eine von Siemens zertifizierte Schnittstelle zwischen Smallworld GIS und PSS SINCAL zur Verfügung.

Mettenmeier GmbH Utility Solutions
Herr Benjamin Pehle
Klingenderstr. 10 – 14
33100 Paderborn
Deutschland
Fon +49 5251 150-375
Fax +49 5251 150-366
e-mail benjamin.pehle@mettenmeier.de
http://mettenmeier.de

Khatib & Alami – C.E.C.

Khatib & Alami – Consolidated Engineering Company (K&A) is a multi-disciplinary company specialized in consulting engineering studies, design and construction supervision of architectural, structural, civil, electrical, mechanical, industrial, environmental, transportation, telecommunication, information technology and geographic information systems (GIS) projects.

K&A stellt eine Schnittstellen zwischen ArcGIS und PSS SINCAL zur Verfügung. Diese Schnittstelle ist für elektrische Übertragungs- (Transmission) und Verteilungsnetze (Distribution) geeignet.

Khatib & Alami – C.E.C.
GIS Services Division
Herr Bilal A. Hassan
P.O. Box 2732, Abu Dhabi – UAE
Fon +971-2-6767300 X202
Fax +971-2-6767070
e-mail bahassan@gisadwea.ae
http://www.khatibalami.com