COPADATA
do it your way

zenon Handbuch

Effizient projektieren mit zenon

v.7.00

© 2012 Ing. Punzenberger COPA-DATA GmbH

Alle Rechte vorbehalten.

Die Weitergabe und Vervielfältigung dieses Dokuments ist - gleich in welcher Art und Weise – nur mit schriftlicher Genehmigung der Firma COPA-DATA gestattet. Technische Daten dienen nur der Produktbeschreibung und sind keine zugesicherten Eigenschaften im Rechtssinn. Änderungen – auch in technischer Hinsicht - vorbehalten.

Inhalt

1. Willkommen bei der COPA-DATA Hilfe 5

2. Effizient projektieren mit zenon 5

3. Einfache Bedienung 7

4. Objektorientierung 8

4.1 Parametrieren statt programmieren 9

4.2 Global und zentral statt lokal 11

4.3 Objektorientiert parametrieren 13

4.3.1 Treiber 13

4.3.2 Datentypen 14

4.3.3 Tipps zu Adressierung und Import-Export 20

5. Elemente wiederverwenden 21

5.1 Variablen und Funktionen ersetzen 22

5.1.1 Namenskonventionen 22

5.1.2 Möglichkeiten der Ersetzung 23

5.2 Symbole 28

5.2.1 Frei gegebene Eigenschaften 28

5.2.2 Ersetzung bei Symbolen 31

5.2.3 Globale Symbolbibliothek 32

5.3 Struktur-Datentyp 33

5.4 Reaktionsmatrizen 35

5.5 Globalprojekt 36

5.6 XML 38

5.7 Wizards 40

5.8 Projekte wiederverwenden 41

5.8.1 Projektsicherung 41

5.8.2 Speichern unter 42

5.8.3 Mehrprojektverwaltung 43

6. Integriertes Netzwerk 44

1. Willkommen bei der COPA-DATA Hilfe

ALLGEMEINE HILFE

Falls Sie in diesem Hilfekapitel Informationen vermissen oder Wünsche für Ergänzungen haben, wenden Sie sich bitte per E-Mail an documentation@copadata.com (mailto:documentation@copadata.com).

PROJEKTUNTERSTÜTZUNG

Unterstützung bei Fragen zu konkreten eigenen Projekten erhalten Sie vom Support-Team, das Sie per E-Mail an support@copadata.com (mailto:support@copadata.com) erreichen.

LIZENZEN UND MODULE

Sollten Sie feststellen, dass Sie weitere Module oder Lizenzen benötigen, sind unsere Mitarbeiter unter sales@copadata.com (mailto:sales@copadata.com) gerne für Sie da.

2. Effizient projektieren mit zenon

Sie arbeiten zum ersten Mal mit zenon?
Hier lesen Sie, wie Sie zenon effizient einsetzen. Auch wenn Sie bereits Erfahrung mit anderen Prozessleitsystemen haben, empfehlen wir Ihnen einen Blick in dieses Kapitel. Sie erfahren, welche Grundidee hinter zenon steht und wie es funktioniert, warum und wie zenon Ihre Arbeit einfacher und schneller macht.

ZENON IM ÜBERBLICK

zenon besteht aus Editor und Runtime.
Projekte werden im Editor erstellt. Das Bedienen und Beobachten erfolgt in der Runtime. Editor und Runtime sind ab Version 6.20 versionsübergreifend kompatibel. Ein Projekt, das mit Version 6.50 erstellt wurde, läuft zum Beispiel auch auf einer Runtime mit Version 6.22 und umgekehrt.

zenon unterscheidet sich von anderen Systemen vor allem durch:

- Einfache Bedienung (auf Seite 7): Viele Aktionen benötigen nur wenige Mausklicks, Assistenten und Wizards machen die Arbeit leicht und schnell, Sie arbeiten mit Drag&Drop per Maus oder mit der Tastatur, wie Sie es wünschen.
- Praktische Objektorientierung (auf Seite 8): Sie arbeiten durchgängig mit Objekten, die Sie schnell parametrieren statt aufwändig programmieren. Sie nutzen einmal definierte Objekte auch projektübergreifend, profitieren von Vererbung und blitzschnellen Änderungen von Eigenschaften.
- Exzellente Wiederverwendbarkeit (auf Seite 21) einmal projektierter Objekte und Elemente: Von einzelnen Variablen bis zu kompletten Projekten sind in zenon viele Elemente sehr einfach und effektiv wiederverwendbar.
- Integriertes Netzwerk (auf Seite 44): Verteiltes Engineering im Netzwerk und 100% Redundanz sind kein besonderer Aufwand mehr. Netzwerke erstellen Sie durch Aktivieren der entsprechenden Checkbox. Und genau so einfach vernetzten Sie vereinzelte stand alone Projekte zu effizienten, redundanten Systemen.

TIPPS UND TRICKS

Viele nützliche Tipps finden Sie jederzeit im Forum von www.copadata.com. Wie Sie zenon optimal einsetzen, lernen Sie am besten in den COPA-DATA Trainings und Workshops, abgestimmt auf Ihre Schwerpunkte und Projekte. Ihr Vertriebspartner informiert Sie gerne - oder schreiben Sie eine kurze Mail an sales@copadata.com (mailto:sales@copadata.com).

3. Einfache Bedienung

zenon macht Ihnen das Projektieren durch viele kleine Annehmlichkeiten leichter und erhöht so Ihre Produktivität. Bei der Arbeit mit zenon werden Sie entdecken, dass Sie viele Aufgaben ganz nach Ihren Präferenzen lösen können. Zum Beispiel:

FAVORITEN

In den Eigenschaften der Objekte und im Funktionsauswahldialog können Sie eigene Favoriten definieren. Diese werden Ihnen immer am Beginn der Eigenschaftenlisten angeboten und geben Ihnen so schnellen Zugriff auf jene Eigenschaften, die sie besonders häufig benötigen.

ALLGEMEINE TASTATURBEDIENUNG IM EDITOR

Der Editor lässt sich bequem über Symbole und Menüs mit der Maus bedienen, aber auch durchgängig per Tastatur. Die von Windows gewohnten Tastenkürzel funktionieren auch in zenon. Zum Beispiel:

- `Strg+C` und `Strg+V` für **`kopieren`** und **`einfügen`**
- `Einfg` erstellt neue Objekte
- `Entf` löscht markierte Objekte
- **`Cursor`** Tasten dienen zum Navigieren in Ansichten
- `Esc` bricht Aktion ab
- `Rückwärts löschen` (Backspace) springt im Verzeichnisbaum auch auf die darüber liegende Ebene
- `Strg+Z` macht letzte Aktionen rückgängig, ausgenommen Datenbankoperationen

Mehr zur Tastaturbedienung lesen Sie im Kapitel Tastenkombinationen.

DRAG&DROP

Im Editor haben Sie die Möglichkeit, Arbeitsschritte per Drag&Drop zu vereinfachen. Zum Beispiel:

- Reihenfolge der Menüeinträge ändern
- Schriften auf Dynamische Elemente wie Universalregler, Uhr, Combi-Element etc. verknüpfen
- Reihenfolge von Strukturelemente für Strukturdatentypen ändern

- Reihenfolge innerhalb eines Scripts ändern
- Kopieren oder verschieben von Funktionen über verschiedene Scripts
- Skripte kopieren, indem sie auf den Stammknoten gezogen werden

INTELLIGENTE ASSISTENTEN UND WIZARDS FÜR KOMPLEXE AUFGABEN

Die in zenon integrierten Assistenten und Wizards helfen Ihnen, schnell ein Basisprojekt zu erstellen oder komplexe Einstellungen zielgerichtet vorzunehmen. Wir empfehlen vor allem Einsteigern die Assistenten für den Universalregler, das Combi-Element und die Archivierung.

TOOLTIPS FÜR ELEMENTE UND EIGENSCHAFTEN

Wenn Sie den Mauszeiger über ein Anzeigeelement (z. B. eine Text-Schaltfläche) oder eine Eigenschaft mit Verknüpfung (z. B. die mit einer Text-Schaltfläche verknüpfte Funktion) bewegen, erscheint ein Tooltip. Dieser zeigt Informationen der verknüpften Elemente an, wie die verknüpfte Funktion und deren Parameter.

ERWEITERTE ZEICHENMÖGLICHKEITEN

Der grafische Editor bietet Ihnen eine breite Auswahl an Funktionen, die Ihnen das Abbilden von Anlagen leicht macht und Ihnen gleichzeitig sehr vielfältige Gestaltungsmöglichkeiten gibt. Vom stufenloser Zoom über skalieren per Klick bis zu einrastenden Elementeckpunkten und automatischem Verteilen von Elementen nach Regeln. Details dazu finden Sie im Kapitel Bilder.

SPRINGEN ZU VERKNÜPFTEN ELEMENTEN

Bei allen grafischen Elementen, bei Funktionen, bei Variablen und auch bei anderen Elementen haben Sie die möglichkeiten direkt zu den Verknüpften Elementen zu springen. Beispielsweise kommen Sie von einem Anzeigeelement mit einem Mausklick direkt zur verknüpften Variable, von der Variable direkt zum verwendeten Datentyp und von dort sogar weiter zur verwendeten Einheit. Bei Bedarf können Sie alle diese Schritte auch bequem per Mausklick wieder zurück zum Ausgangselement springen.

4. Objektorientierung

COPA-DATA betrachtet das Thema Automatisierung erstens objektorientiert und zweitens ganz pragmatisch. So funktioniert auch zenon, das sich darüber hinaus auch selbst weitgehend

automatisieren lässt. Sie benötigen für zenon keine Programmiersprache. Sie müssen keine Zeile Code programmieren. Stattdessen weisen Sie Objekten per Mausklick Eigenschaften zu. Das funktioniert nicht nur mit einzelnen Objekten, sondern auch zentral für ein ganzes Projekt und sogar projektübergreifend.

Wenn Sie jetzt befürchten, dass Sie zenon nicht individualisieren können und ganz auf die Objekteigenschaften angewiesen sind, heißt die gute Nachricht: Natürlich können Sie zenon auch sehr individuell gestalten. Dazu nutzen Sie vor allem VBA oder die VSTA (.NET) Entwicklungsumgebung. Außerdem bietet zenon viele Schnittstellen, um mit Hardware und Software zu kommunizieren und verfügt mit zenon Logic über eine integrierte SCADA Logic.

DREI WICHTIGE GRUNDGEDANKEN FÜR DIE ARBEIT MIT ZENON

Drei zentrale Prinzipien machen die die Projektierung mit zenon einfach, zuverlässig und wartungsfreundlich:

1. Parametrieren statt programmieren (auf Seite 9):

   Einfach die nötigen Parameter setzen statt zu programmieren oder Skripte anzupassen.

2. Global/zentral statt lokal (auf Seite 11):

   Objekte einmal definieren und immer wieder verwenden, auch projektübergreifend, statt Skripte immer wieder zu schreiben oder zu kopieren.

3. Objektorientiert parametrieren: (auf Seite 13)

   Alle Vorteile objektorientierten Denkens nutzen.

4.1 Parametrieren statt programmieren

Welchen Vorteil bringt es, Projekte zu parametrieren statt zu programmieren?

Sie verzichten vor allem auf fehleranfälligen Code und gewinnen hohe Flexibilität, Übersicht, Tempo und einen sehr hohen Grad an Wiederverwendbarkeit. Vier Punkte sprechen besonders dafür:

1. <u>Vorgefertigte Module bringen Zeitgewinn</u>
   zenon enthält viele vorgefertigte Module. Statt selbst Skripte zu programmieren, wählt der Projektant das passende Modul und konfiguriert es, indem er die gewünschten Eigenschaften und Parameter selektiert. So erstellen Sie in kurzer Zeit lauffähige Projekte.

2. Produkt kommt ausgereift vom Hersteller
   Der Projektant konzentriert sich auf seine Projekte. Die Software wird beim Hersteller entwickelt und getestet, Projektanten müssen keinen eigenen Code schreiben. Damit lässt sich der Umgang mit zenon schnell erlernen, auch für Mitarbeiter ohne Erfahrung im Software-Engineering.

3. Einfache Wartung und Anpassung des Projektes
   zenon arbeitet streng objektorientiert. In Kombination mit dem Grundsatz "parametrieren statt programmieren" heißt das: leichte Wartung und Anpassung. Für Änderungen und Kontrollen muss man nur die Eigenschaften der einzelnen Objekte überprüfen oder ändern statt lange Codezeilen zu lesen und umzuschreiben. Auf die Lebenszeit eines Projektes gesehen bringt das
   - Übersicht: Projekte lassen sich auch nach Jahren und auch von neuen Mitarbeitern noch nachvollziehen.
   - Sicherheit: Fehleranfällige Skripte und Programmierung werden vermieden.
   - Zeitgewinn: Projektierung, Anpassung und Wartung lassen sich sehr viel schneller realisieren.

4. Einfache Anpassung an Maschinen
   zenon lässt sich schnell und einfach an neue oder veränderte Maschinen anpassen. Projektanpassungen verlangen nur wenigen Mausklicks.

ZENON ANPASSEN

Parametrieren verhindert auch, dass Fehler in Skripten wieder und wieder kopiert werden. Es vermeidet umständliche Änderungen in vielen Einzelskripten, wenn lediglich eine Kleinigkeit angepasst werden soll. Das heißt aber nicht, unbeweglich sein. Auch zenon gibt Ihnen Möglichkeiten, das System individuell zu erweitern und anzupassen.

- VBA/VSTA:
  Mit Hilfe der integrierten Skriptsprachen Visual Basics for Applications bzw. C# und VB.NET können Sie auch in der Runtime beliebigen Code ausführen. So lösen Sie Automatisierungsaufgaben, Logikaufgaben und Schnittstellenaufgaben wie Datenbankzugriffe.

- zenon Logic:
  Die vollständig integrierte Steuerungsumgebung erlaubt Ihnen als SCADA Logic alle 5 IEC 61131-3 Sprachen zu nutzen. Damit erledigen Sie alle Berechnungen und Logikaufgaben.

- PCE:
  Die integrierte Process Control Engine ermöglicht Ihnen, Automatsierungsaufgaben in den Sprachen VB-Skript und Java Skript umzusetzen.

- ActiveX Controls:
  Beliebige Standard-Controls wie Flash Player oder Acrobat Reader lassen sich in zenon integrieren. Die Schnittstelle ist offen gelegt und steht damit auch selbst programmierten ActiveX Controls offen.
- WPF-Elemente
  Durch die direkte Integration des Microsoft Formats Windows Presentation Foundation können Sie grafische Elemente wie z.B,. einen Button, ein Zeigerinstrument und vieles anderes in diesem neuen Grafikformat direkt in zenon Bildern einbinden. Diese Elemente können Sie frei mit Variablen und Funktionen verknüpfen und damit beliebige grafische Effekte umsetzen.
- COM-Schnittstelle:
  Über die vollständig offen gelegte COM-Schnittstelle greifen Sie von außen auf zenon zu. Für die Kommunikation stehen beliebige Programmiersprachen wie C++, .NET, Delphi und so weiter zur Verfügung.

4.2 Global und zentral statt lokal

Eigenschaften lassen sich bei zenon auf unterschiedlichen Wegen definieren, je nachdem, was Ihre Aufgabe verlangt:

- direkt am Objekt
- zentral
- global

Natürlich lassen sich auch in zenon alle Einstellungen direkt beim Objekt vornehmen. Aber das ist nicht immer der beste Weg.

EIGENSCHAFTEN ZENTRAL DEFINIEREN

Oft ist es sinnvoll, Einstellungen einmal zentral vorzunehmen. Damit haben neue Objekte von Anfang an die gewünschten Eigenschaften. Änderungen sind besonders einfach: Die gewünschte Eigenschaft wird nur an einer einzigen Stelle zentral geändert - und alle betroffenen Objekte übernehmen sie automatisch. Also nur ein Klick statt vieler Zeilen Code oder vieler Mausklicks.

Diese zentralen Einstellungen lassen sich aus jeder Stelle im Editor leicht finden. Jedes Objekt, das Einstellungen von einer zentralen Quelle bezieht, bietet die Möglichkeit, diese direkt anzuspringen. Sie folgen einfach einem Link und sehen sofort, wie Elemente verbunden sind.

Eigenschaften können Sie aber nicht nur zentral für ein Projekt definieren, sondern auch projektübergreifend global.

GLOBAL PROJEKTIEREN

zenon lässt Sie mehr als ein Projekt zur gleichen Zeit bearbeiten - und spart dabei die wertvollste Ressource: Arbeitszeit. In allen Projekten gleiche Schriften verwenden, ist zum Beispiel sehr einfach. Einmal definiert, stehen Eigenschaften wie Schriftart oder Schriftgröße sofort allen Projekten eines Arbeitsbereiches zur Verfügung. Benötigen Sie eine andere Schriftart für alle Projekte, kostet Sie das nur einige Sekunden, um sie im Globalprojekt zu ändern. Und wenn Sie nur ein Projekt mit einer anderen Schrift versehen möchten? Dann ändern Sie diese Schrift eben nur zentral in diesem Projekt.

> **Info**
>
> *Verwenden Sie im Globalprojekt am besten andere Nummernkreise als in den lokalen Projekten. Denn zenon gibt bei identen Nummern immer den lokalen Einstellungen den Vorzug.*
>
> *Und: Verwenden Sie eigene Namen für Schriftlisten, die Sie im Globalprojekt und in den Einzelprojekten konsistent halten. So stellen Sie sicher, dass auch bei einem Sprachwechsel alle Schriften gefunden werden.*

BEZIEHUNGEN VON OBJEKTEN ÜBERBLICKEN

Übersicht über Ihre Projekte und Beziehungen von Objekten gewinnen Sie zusätzlich mit:

- Querverweisliste (QVL):
  Sie zeigt Querbeziehungen in einem Projekt an und ermöglicht die einfache Suche nach der Verwendung von Variablen und Funktionen.
- Wizard für Dokumentation
  Er erstellt automatisch eine komplette Projektdokumentation. Was dokumentiert wird, lässt sich für jede Eigenschaft exakt einstellen.

4.3 Objektorientiert parametrieren

'Objektorientiert parametrieren' ist die grundlegende Philosophie bei zenon. Sie bringt Übersicht und spart viel Zeit beim Anlegen und Verwalten von Variablen.

Jede Variable in zenon basiert auf zwei Elementen:

- Treiber (auf Seite 13)
- Datentyp (auf Seite 14)

4.3.1 Treiber

Treiber werden in zenon nicht direkt angebunden, sondern über einen Treiber-Objekttyp. Dieser bestimmt, welcher Bereich auf der Steuerung angesprochen werden soll.

TREIBER-OBJEKTTYP

Es existieren viele unterschiedliche Treiber-Objekttypen, zum Beispiel Standard SPS-Merker, Datenbausteine, Eingänge, Ausgänge, Zähler, aber auch Spezialtypen wie Alarm- oder Treibervariablen.

Der Treiber-Objekttyp bestimmt:

- den Bereich auf der Steuerung
- welche Granularität der Treiber in diesem Bereich hat
- welche Datentypen auf dem Bereich angelegt werden können

Tipp: Nicht jeder Datentyp kann in jedem Bereich angelegt werden. Wählen Sie daher beim Anlegen einer Variable immer:

- zuerst den Treiber,
- dann den Treiber-Objekttyp und
- zuletzt den Datentypen

EXKURS: GRANULARITAT

Granularität ist vor allem bei numerisch adressierenden Steuerungen wie z. B. der Siemens S7 wichtig. Nicht jede Steuerung hat dieselbe Rasterung und auch nicht alle Bereiche in einer Steuerung haben dieselbe Rasterung.

So ist z. B. der Datenbausteinbereich der Siemens S5 wortorientiert, der Merkerbereich aber Byte-orientiert. Das bedeutet, dass die kleinste Einheit, die adressiert werden kann, ein Byte oder eben ein Wort ist. Will man z. B. in einen solchen Bereich ein Bit schreiben, muss man mindestens ein ganzes Byte/Wort lesen, das entsprechende Bit ausmaskieren, ändern und dann das ganze Byte/Wort wieder auf die Steuerung schreiben.

Dieselbe Information benötigt man auch bei der Adressierung der Bereiche, Offset genannt. Man beginnt dabei bei Null und zählt in Byte- oder Wort-Schritten weiter. Nutzt man in zenon die automatische Adressierung (auf Seite 20), wird diese Granularität automatisch berücksichtigt. Die Granularität ist also nicht vom Treiber abhängig, sondern vom Treiber-Objekttyp.

TREIBERVARIABLEN

Die Treibervariablen werden von jedem Treiber angeboten und bringen viele Vorteile für die Projektierung. Sie kommunizieren nicht mit der Steuerung, sondern lesen einen internen Speicherbereich des Treibers mit vor allem statistischen Informationen aus. Aber auch spezielle Funktionen lassen sich mit Variablen steuern, z. B. Telefonnummern für Modemverbindungen oder Befehle zum Wählen oder Auflegen.

Genauere Informationen dazu lesen Sie im Hilfekapitel Treibervariablen.

Auf der Installations-CD von zenonfinden Sie eine dBase-Datei mit den wichtigsten Treibervariablen. Damit können Sie diese Variablen zu jedem Treiber importieren statt sie manuell anzulegen.

4.3.2 Datentypen

Datentypen sind das Herzstück des objektorientierten Parametrierens. Da bei den Treiber-Objektypen keine Einstellungen vorgenommen werden können, sind diese für die Objektorientierung nicht nutzbar. Die Datentypen dagegen bieten hier sehr viele Möglichkeiten.

Wir unterscheiden drei Arten:

- einfache Datentypen
- Struktur-Datentypen

- Strukturelemente

Einfache Datentypen

Einfache Datentypen sind immer IEC Datentypen, also die im IEC Standard 61131-3 definierten Datentypen wie BOOL, INT, USINT, UDINT, USINT, STRING, WSTRING usw. Sie bringen die Länge eines Bereichs mit - zum Beispiel:
BOOL: 1 Bit
INT: 16 Bit mit Vorzeichen
UINT: 16 Bit ohne Vorzeichen

Die Datentypen in zenon erlauben neben dem IEC Datentyp aber die Einstellung von weit mehr Eigenschaften. Diese haben dieselben Bezeichnungen wie die Eigenschaften der Variablen:

- Kennung
- Einheit
- Wertbereich
- Grenzwerte

Wozu diese "doppelte Buchführung"?

Der Hintergrund ist so einfach wie praktisch:
Die Variable übernimmt Wertberechnung, Einheit und eben auch Grenzwerte vom zugrundeliegenden Datentyp - so wie ein Anzeigelement z. B. die Schriftart und Schriftgröße von der verknüpften Schrift übernimmt.

OBJEKTORIENTIERUNG

Anders als bei der Verknüpfung von Dynamischen Elementen und Schriften haben wir es hier mit einem objektorientierter Ansatz zu tun: Das Datentypobjekt vererbt seine Eigenschaften auf die Variable. Der Unterschied?

Verknüpfte/abgeleitete Eigenschaften lassen sich bei der Variablen trennen oder überschreiben. Das funktioniert für jede Eigenschaft einzeln, aber auch für alle Eigenschaften gleichzeitig. Sie können also z. B. die Einheit vom Datentyp zentral übernehmen, die Kennung, oder die Adresse jedoch direkt bei der Variablen überschreiben. Das funktioniert so:

- Ändern Sie bei der Variablen den entsprechenden Wert, dann wird die Verknüpfung getrennt. Sie erkennen das auch daran, dass in den Eigenschaften das Häkchen verschwunden ist.

Über das Kontextmenü können Sie folgende Funktionen nutzen:

- Eigenschaft vom Datentyp ableiten:

Wenn Sie die Verknüpfung getrennt haben, können Sie diese wieder herstellen.

- Eigenschaft trennen:

Sie können eine einzelne Eigenschaft trennen, ohne dass Sie den Wert ändern müssen. Wenn Sie später den Wert beim Datentyp ändern, können Sie sicher sein, dass die Variable davon nicht beeinflusst wird. So können Sie ganz bewusst einstellen, ob das Vererbungskonzept greift oder nicht.

Als besondere Hilfestellung können Sie auch alle Eigenschaften mit einem Mausklick vom Datentyp übernehmen oder trennen. Damit können Sie besonders schnell den Ursprungszustand (alles abgeleitet) wiederherstellen oder eben aufheben. Was bringt Ihnen das Ganze in der Praxis?

Besonders hohe Flexibilität: Wenn Sie viele Variablen mit gleichem Grenzwert haben, z. B. einen Alarm bei Wert 1, dann stellen Sie diesen beim Datentyp ein. Den Grenzwertext und andere optionale Parameter stellen Sie dann bei jeder Variable gesondert ein. Sie sparen sich damit das Anlegen und Warten des Grenzwerts bei jeder einzelnen Variablen und Ihre Engineeringleistung steigt deutlich.Der standardmäßig angelegte Datentyp BOOL hat übrigens bereits einen Grenzwert bei 0 und bei 1!

Wenn Sie nun gar nicht bei jeder Bool-Variablen einen Grenzwert haben, müssen Sie aber nicht bei jeder Variablen extra aufpassen und den Grenzwert wieder ausschalten. Sie nutzen einfach eine bequeme Möglichkeit unter zenon und legen einen eigenen Bool-Datentyp an. Gehen Sie so vor:

- Standard-Datentyp einfach lassen wie er ist
- auf `Datentyp neu` klicken
- eine Liste mit den existierenden Datentypen wird angeboten
- `Standard-Bool` auswählen
- Namen vergeben, z. B. `MyBool`
- alle Eigenschaften des neuen Datentyps werden vom alten automatisch übernommen
- bei `MyBool` Grenzwerte löschen

Alle Bool-Variablen, die keine Grenzwerte haben sollen, legen Sie dann auf diesem neuen Datentyp `MyBool` an – und schon sind die nicht gewünschten Grenzwerte verschwunden. Jene, die erwünscht sind, bleiben aber weiter erhalten.

Sie können natürlich nicht nur einen eigenen Bool-Datentyp erstellen, sondern beliebig viele anlegen. Alle Variablen, die über weite Bereiche idente Eigenschaften benötigen, bekommen einen eigenen Datentyp. Ändert sich eine Eigenschaft, ändern Sie diese zentral und alle abgeleiteten Variablen werden automatisch mitgeändert – ausgenommen jene, bei denen die Verknüpfung getrennt wurde.

Struktur-Datentypen - Strukturelemente - Strukturarray

STRUKTUR-DATENTYP

Struktur-Datentyp meint eine Gruppe von Datentypen, die in ihrer Reihenfolge und ihrer Anordnung genau definiert sind. Zum Beispiel kann sich die Struktur Motor aus den Elementen Ist-Drehzahl und Leistungsaufnahme zusammensetzen.

Eine Struktur kann auch verschachtelt sein, wobei beliebige Hierarchieebenen möglich sind.

Mit Struktur-Datentypen lässt sich der Variablenhaushalt genau so aufbauen, wie er in der Steuerung bzw. logisch in der Realität existiert.

STRUKTURELEMENT

Ein Struktur-Datentyp ist eigentlich nur eine Hülle, die den Namen für die Struktur vergibt, selbst jedoch keine eigenen Eigenschaften mitbringt.

Diese kommen erst mit den Strukturelementen, die dem Struktur-Datentyp hinzugefügt werden. Dabei unterscheiden wir:

- verknüpfter Struktur-Datentyp:

  Bezug zu einem bereits bestehenden Datentyp. Alle Eigenschaften werden übernommen, ausgenommen der Name.

  Vorteil: Hat das Strukturelement sehr viele Eigenschaften eines bereits existierenden Datentyps, können diese einfach wiederverwendet werde. Bei der Variablen können natürlich nach wie vor einzelne Eigenschaften geändert oder vom Datentyp getrennt werden.

- eigener eingebetteter Struktur-Datentyp:

  Erlaubt individuelle Adressierungseinstellungen. Dieser neue Datentyp ist ausschließlich innerhalb dieser Struktur gültig.

> **Info**
>
> Vorteil Objektorientierung: Ändert sich die Struktur irgendwann, dann genügt es, den Struktur-Datentyp zu ändern, um alle Variablen, die auf dieser Struktur liegen, automatisch anzupassen.

Änderungen sind in allen Ausprägungen möglich:

- einfachen Eigenschaften wie die Einheit beim Datentyp
- komplexe Eigenschaften wie das Hinzufügen oder Löschen eines Grenzwertes
- Änderung der gesamten Struktur durch z. B. Änderung der Reihenfolge der Elemente oder Hinzufügen oder Löschen einzelner Strukturelemente im Struktur-Datentyp

STRUKTURARRAY

Wenn es nicht nur um eine Maschine geht, sondern um viele, kommt das Strukturarray zum Einsatz.

Zum Beispiel: 100 Pumpen statt 1 Pumpe.
Dann verwandeln Sie die Strukturvariablen einfach ein Strukturarray. So werden mit einem einzigen Mausklick 100 Variablen zu 100 Strukturvariablen.

Wenn wir pro Pumpe zum Beispiel je 12 Einzelvariablen anlegen müssen, benötigen wir 1200 Variablen insgesamt. Die Methode „objektorientiert parametrieren" ermöglicht das mit wenigen Mausklicks. Dazu kommt der Vorteil, dass alle Variablen schon voreingestellt sind. Jede einzelne Variable bringt bereits alle Eigenschaften mit, die sie braucht , wie Einheit, Wertberechnung, Alarme, CEL Einträge usw.

Info

Struktur-Datentypen eigenen sich auch für die Wiederverwendung (auf Seite 33) von Variablen.

Beispiel Pumpe

AUSGANGSLAGE

Eine Pumpe besteht aus zwei Motoren. Jeder davon hat einige Variablen wie:

- `Ist-Drehzahl`
- `Stromaufnahme`

- `Leistung`
- usw.

Jeder Motor wiederum hat einen Motorregler mit folgenden Variablen:

- `Soll-Drehzahl`
- `P-Anteil`
- `I-Anteil`

Diese Konstellation bauen Sie so auf:

1. Erzeugen Sie eine Struktur `Regler` aus je einem Datentyp für
   - `Soll-Drehzahl`
   - `P-Anteil`
   - `I-Anteil`
2. Erstellen Sie eine Struktur `Motor` mit den jeweils eigenen Datentypen:
   - `Ist-Drehzahl`
   - `Stromaufnahme`
   - `Leistung`
   - usw.
3. In die Struktur `Motor` übernehmen Sie den `Regler`.
4. Erstellen Sie die Pumpenstruktur und integrieren Sie dort den Motor.

   Da wir zwei Motoren haben, übernehmen Sie den Motordatentyp einfach zweimal in Ihre Struktur `Pumpe`.

5. Erstellen Sie eine Variable, die auf diesem Strukturdatentyp beruht:
   - Neue Variable anlegen
   - Struktur-Datentyp `Pumpe` auswählen

   Die einzelnen Elemente dieser Strukturvariablen heißen Strukturelemente (auf Seite 17) und können in zenon überall als einzeln Elemente genutzt werden, zum Beispiel in Bildern, Alarmen, Archiven, Rezepten usw.

4.3.3 Tipps zu Adressierung und Import-Export

Tipps aus der Praxis zu:

- Adressierung
- Export/Import

ADRESSIERUNG:

Bei numerischen Steuerungen haben Sie die Wahl, ob die Adressierung automatisch oder halbautomatisch erfolgen soll. Alle hier beschriebenen Eigenschaften sind auch über VBA/VSTA erreichbar und können damit für die automatische Projektierung genutzt werden.

AUTOMATISCHE ADRESSIERUNG:

Offset und - falls erforderlich - Byte- und Bitadresse werden auf Basis der Position der Strukturelemente automatisch errechnet, bei Arrays auch über das gesamte Array hinweg. Wenn Sie in der Steuerung dieselbe Struktur wie in zenon angelegt haben, passt das Ganze zu 100% zusammen und Sie brauchen sich um die richtige Adressierung keinerlei Gedanken machen.

HALBAUTOMATISCHE ADRESSIERUNG:

Sie vergeben einzelne Startadressen bereits beim Datentyp. Anhand dieser Adressen werden dann die weiteren Adressen berechnet. Diese können Sie bei Bedarf natürlich auch anpassen. Details dazu finden bei Automatische Adressierung.

Bei symbolisch adressierten Steuerungen müssen in zenon und in der Steuerung die gleichen Namen vergeben werden.

XML EXPORT/IMPORT:

Beim Export und Import werden auch Informationen wie Datentypen, Strukturen, Ableitungen usw. mitgenommen. Sie können also Ihre einmal erstellten Struktur-Datentypen und Strukturvariablen bequem exportieren und über Import in anderen Projekten eins zu eins weiter nutzen oder nach Bedarf modifizieren.

5. Elemente wiederverwenden

Beim ersten Anlegen eines Projektes benötigen Sie Zeit für das Erstellen von Variablen, Funktionen, Bildern und deren Verknüpfungen. Diese Zeit können Sie bei folgenden Projekten deutlich reduzieren. Denn mit zenon haben Sie die Möglichkeit, einmal erstellte Elemente einfach in andere Projekte zu übernehmen.

Zum Beispiel benötigen Sie in vielen Projekte Standardelemente wie Bilder für den Systemstatus oder Detailbilder für immer wieder verwendete Hardwarekomponenten (Pumpen, Ventile, Motoren usw.).

Um Elemente wiederzuverwenden, stehen Ihnen verschiedene Wege zur Verfügung:

Thema	Wiederverwendung von
Variablen und Funktionen ersetzen (auf Seite 22)	Elementen
Symbole (auf Seite 28)	Bildern
Struktur-Datentyp (auf Seite 33)	Variablen
Reaktionsmatrizen (auf Seite 35)	Variablen
Globalprojekt (auf Seite 36)	zentralen Elementen eines Projekts
XML (auf Seite 38)	Projektteilen mittels Import und Export
Wizards (auf Seite 40)	Bildern und Elementen mittel Import oder Individualisierung
Projekte wiederverwenden (auf Seite 41)	Projekten

Info

Um Element effizient wiederverwenden zu können, achten Sie schon bei der Erstellung von Variablen, Funktionen und Bildern auf eindeutige Benennung (auf Seite 22).

5.1 Variablen und Funktionen ersetzen

Variablen und Funktionen, die bei Dynamischen Elementen hinterlegt werden, können automatisiert ersetzt werden. Dies kann an unterschiedlichen Stellen geschehen:

- Verknüpfung im Bild ersetzen (auf Seite 23)
- Verknüpfung bei Bildumschaltung ersetzen (auf Seite 25)
- Indizes ersetzen (auf Seite 26)
- Mehr über Ersetzung von Variablen und Funktionen lesen Sie auch im Handbuch Bilder im Abschnitt Verknüpfungen von Variablen und Funktionen ersetzen.

5.1.1 Namenskonventionen

Um Variablen und andere Elemente sicher ersetzen zu können, sollte die Benennung systematisch und möglichst normiert erfolgen. Damit unterstützen Sie nicht nur die Wiederverwendbarkeit, sondern auch Wartung und Reverese Engineering.

Verschiedene Systeme unterstützen Sie bei der systematischen Benennung.

ZUM BEISPIEL: ENERGIEWIRTSCHAFT

Deutschland

- KKS (Kraftwerk-Kennzeichen-System), Details siehe http://de.wikipedia.org/wiki/Kraftwerk-Kennzeichensystem (http://de.wikipedia.org/wiki/Kraftwerk-Kennzeichensystem)
- DIN 6779 (Kennzeichnungssystematik für technische Produkte und technische Produktdokumentation), Details siehe http://de.wikipedia.org/wiki/DIN_6779 (http://de.wikipedia.org/wiki/DIN_6779)
- AKZ (Anlagenkennzeichnungssystem), Details siehe http://de.wikipedia.org/wiki/Anlagenkennzeichnungssystem (http://de.wikipedia.org/wiki/Anlagenkennzeichnungssystem)

International

- KKS (Power Plant Classification System), Details siehe http://en.wikipedia.org/wiki/KKS_Power_Plant_Classification_System (http://en.wikipedia.org/wiki/KKS_Power_Plant_Classification_System)

Solche Normen existieren für alle Branchen. Es empfiehlt sich, deren Namenskonventionen zu benutzen.

BEISPIEL KKS:

In einem Energieprojekt sollten Variablen gemäß dem KKS benannt werden. Eine entsprechend benannte Variable mit der Bezeichnung `C01_MDY10-QA001_QA07` sagt aus:

- Windenergieanlage `C01` (Reihe C, Nr. 1)
- Windturbinensteuerung `MDY10`, Leistungsteil `QA001`,
- Leistungsschutz `QA07`

5.1.2 Möglichkeiten der Ersetzung

Ersetzungen können an unterschiedlichen Stellen eines Projekts verwendet werden:

- Verknüpfung im Bild ersetzen (auf Seite 23): Bilder werden kopiert und im kopierten Bild werden die Verknüpfungen ersetzt.
- Verknüpfung bei Bildumschaltung ersetzen (auf Seite 25): Es wird für unterschiedliche Steuerungen nur ein einziges Bild verwendet und beim Aufschalten werden die Verknüpfungen angepasst.
- Indizes ersetzen (auf Seite 26): Ersetzen von Variablen in einem Prozessbild über den Wert von Indexvariablen.

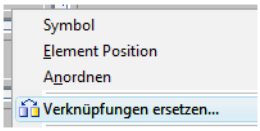
Verknüpfung im Bild ersetzen

Wenn in einem Bild Variablen verknüpft sind, können diese über einen Ersetzungsdialog bequem ausgetauscht werden. Voraussetzung:

- klare Benennung der Variablen (auf Seite 22)

Damit können einmal erstellte Bilder über Copy & Paste weiter verwendet werden. Die Ersetzung wird über das Kontextmenü gestartet:

- Rechtsklick auf das Bildelement
- Klick auf **`Verknüpfungen ersetzen`**



Damit wird der Dialog für die Ersetzung geöffnet.

BEISPIEL

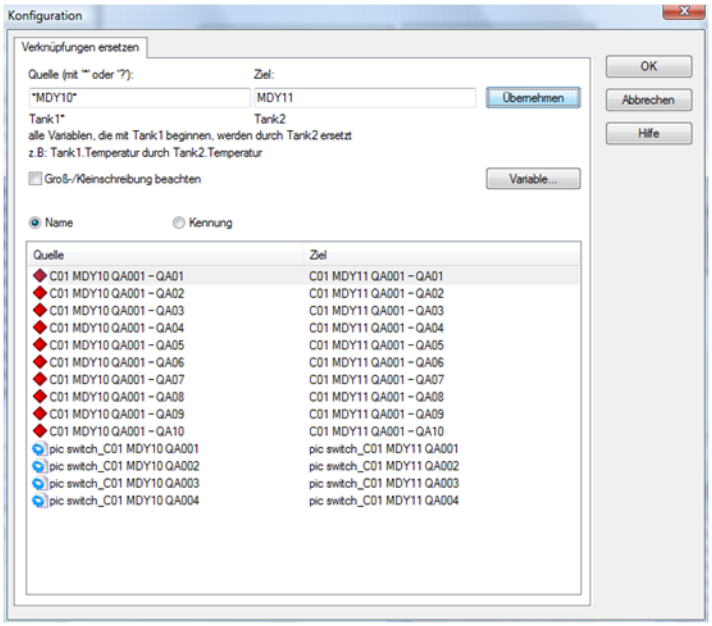
In unserem Beispiel verfügt das Projekt über eine Menge an Variablen aus unterschiedlichen Anlagenteilen. Auf dem Prozessbild sind:

- 10 Variablen, die aus der Windturbinensteuerung **`MDY10`** kommen, mit dynamischen Elementen verknüpft
- Buttons mit Bildumschaltung auf unterschiedliche Bereiche von **`MDY10`** vorhanden

Über Copy & Paste will der Projektant das Bild 1:1 für die Windturbinensteuerung **`MDY11`** wiederverwenden und die Variablen bzw. Funktionen durch die entsprechenden der neuen Steuerung ersetzen. Dazu:

1. werden die dynamischen Elemente, auf denen die Variablen und Funktionen verknüpft sind, markiert

2. wird der Ersetzungsdialog geöffnet



3. im Feld `Quelle` wir die Anlagenkennzeichnung `*MDY10*` eingegeben
   (d führenden und schließenden `*` sind Wildcards)

4. im Feld `Ziel` wird `MDY11` eingegeben

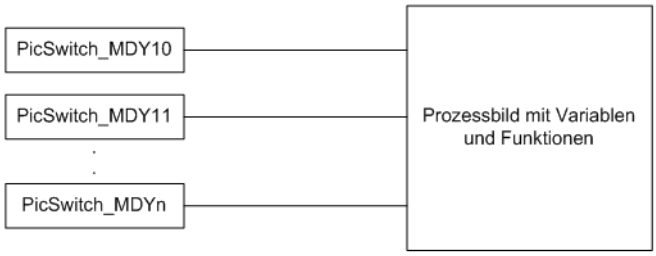
5. mit **`Übernehmen`** wird die Ersetzung durchgeführt und mit **`OK`** der Dialog geschlossen

Das Bild kann jetzt für die neue Steuerung verwendet werden.

Verknüpfung bei Bildumschaltung ersetzen

Bei dieser Methode wird immer das Originalbild verwendet und zur Runtime mit unterschiedlichen Variablen und Funktionen aufgeschaltet. Das Bild enthält so wie im Beispiel "Verknüpfung im Bild ersetzen (auf Seite 23)" verschiedene Variablen und Funktionen. Die Ersetzung erfolgt beim Umschalten. Dazu:

1. werden mehrere Funktionen Bildumschaltung auf dieses Bild projektiert
2. wird beim Erstellen der Funktion der Dialog zur Ersetzung (auf Seite 23) angeboten
3. beinhaltet jede Funktion einen eigenen Ersetzungsvorgang



Eine Variante dazu, die mit nur einer Bildumschaltung auskommt: Indizes ersetzen (auf Seite 26) für Arrays.

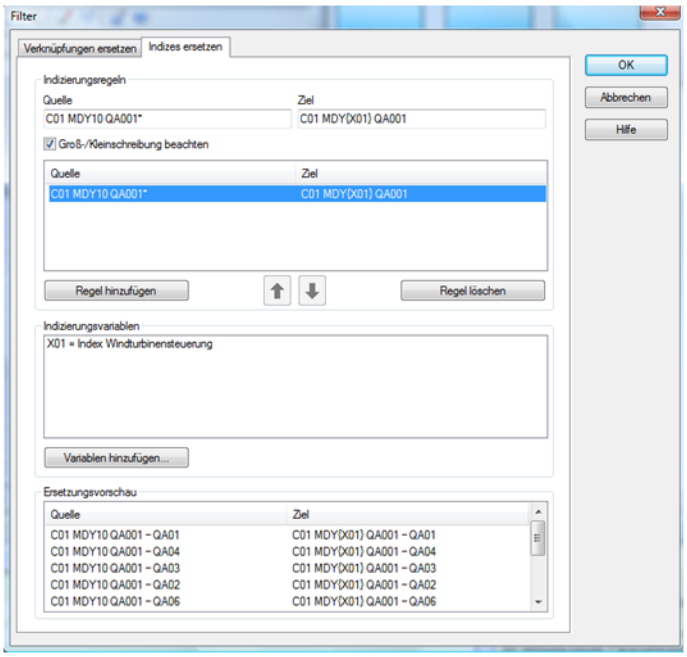
Indizes ersetzen

In Arrays kann die Ersetzung für Variablen über Indexvariablen erfolgen.

Wie in Verknüpfung bei Bildumschaltung ersetzen (auf Seite 25) wird das Bild nur einmal verwendet. Die verknüpften Variablen werden bei der Umschaltung ersetzt. Die Bildumschaltung wird nur einmal projektiert (Unterschied zu Verknüpfung im Bild ersetzen (auf Seite 23)) und kann über eine Index-Variable mehrmals wiederverwendet werden kann.

BEISPIEL

Die Bildumschaltung wird auf ein Prozessbild durchgeführt, das 10 Variablen der Windturbinensteuerung `MDY10` beinhaltet. Ziel des Projektanten ist es, dieses Bild 1:1 wiederzuverwenden, da die Windturbinensteuerungen `MDY11`, `MDY12` und `MDY13` über die selbe Anzahl an Variablen verfügen.



Die Indizierungsregel:

- Quelle: C01 MDY10 QA001*
- Ziel: C01 MDY{X01} QA001

bewirkt:

- bei der Bildumschaltung wird der Funktion mitgeteilt, dass sie bei der Bildumschaltung zur Runtime den Variablenwert von `X01 Index Windturbinensteuerung` in die Variablennamen einsetzen muss.

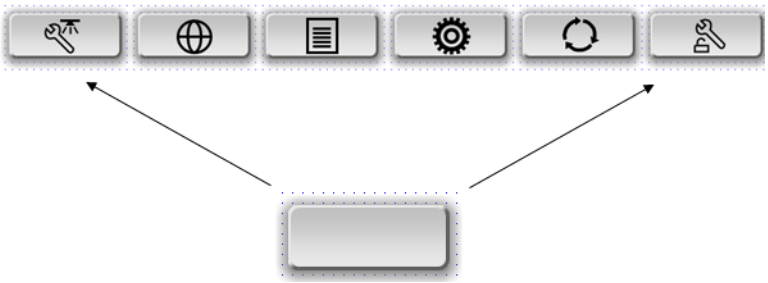
Beispiel: Besitzt diese Variable den Wert 12 so wird beim Ausführen der Bildumschaltung das Prozessbild mit allen Variablen der Windturbinensteuerung `MDY12` angezeigt

5.2 Symbole

Symbole bieten ein großes Potenzial zur Wiederverwendung. Symbole lassen sich sehr komplex aufbauen und unterstützern Vererbung. Symbole können eingebettet oder verknüpft werden. Bei Verlinkung genügt die Modifikation an einer Stelle, um alle Bilder, die dieses Symbol verwenden, zu aktualisieren. Symbole können auch in andere Symbole verknüpft werden.

BEISPIEL BUTTONLEISTE

Ein Symbol (leeres Rechteck) wird als Basis in alle Buttons verknüpft.



Frei gegebene Eigenschaften (auf Seite 28) ermöglichen es, jeden Button mit unterschiedlichen Grafiken zu versehen. Sollen Form oder Farbe der Buttons verändert werden, muss nur das Symbol "leerer Button" modifiziert werden. Alle Buttons übernehmen automatisch die neue Form und/oder Farbe.

5.2.1 Frei gegebene Eigenschaften

Verknüpfte Symbole vererben ihre Eigenschaften auf die Objekte, in die sie verknüpft wurden. Allerdings können einzelne Eigenschaften aus der Vererbung gelöst werden. Zum Beispiel die angezeigte Grafik in unserem Button. Wird die Vererbung gelöst, kann diese Eigenschaft für jedes Objekt individuell gesetzt werden. Änderungen dieses details im Ausgangselement beeinflussen die anderen Objekte nicht mehr.

BILDER WIEDERVERWENDEN

Die Kombination aus Symbolen und freigegebenen Eigenschaften kann eine effektive Lösung für die Wiederverwendung von Prozessbildern sein.

BEISPIEL

In einem Projekt gibt es einen zentralen Parametrierungsdialog, der für unterschiedlichste Parametrierungen verwendet werden soll. Da dieser in unterschiedlichsten Bereichen eingesetzt wird, besteht auch die Forderung, dass gewisse Anpassungen (z.B. Hintergrundfarben, Bilder, …) möglich sein sollen.

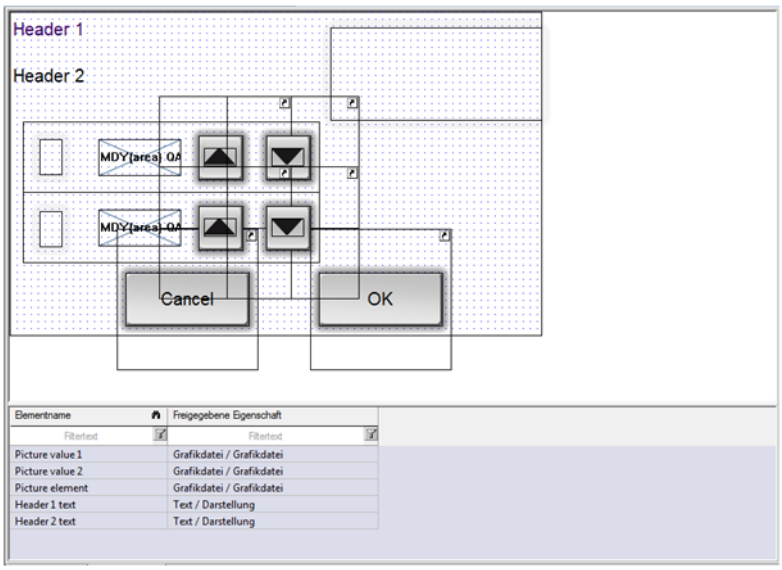
Copy and Paste:

1. ein Prozessbild erstellen, das diesen Parametrierungsdialog enthält
2. Bild über Copy & Paste vervielfältigen
3. die mehrfach vorhandenen Prozessbilder können grafisch an die Anforderungen angepasst werden

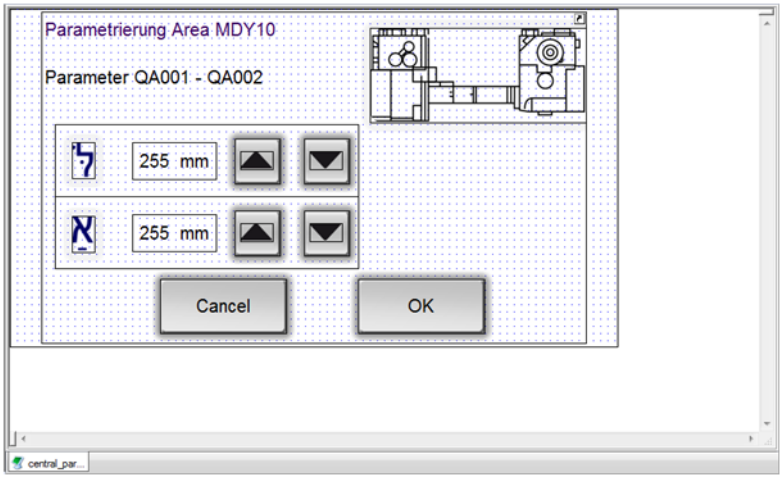
Aber: Vererbung ist hier nicht möglich. Möchte der Projektant im Nachhinein zentrale Anpassungen (betreffend der Grundstruktur des Parametrierungsdialogs) durchführen, so muss er diese in jedem einzelnen Prozessbild nachziehen.

Ansatz über Symbol mit freigegebenen Eigenschaften:

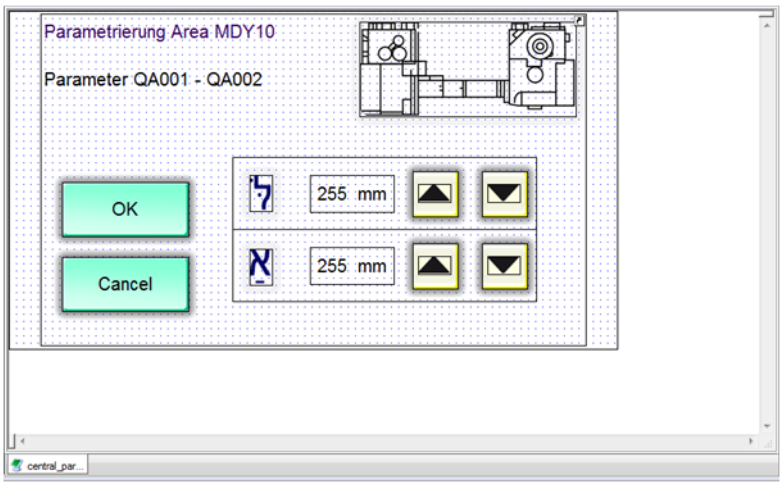
1. der Parametrierungsdialog wird über ein Symbol erstellt
2. Elemente, die für die unterschiedlichen Parametrierungsstellen angepasst werden müssen, werden über freigegebene Eigenschaften aus dem Vererbungskonzept entkoppelt
3. das Symbol wird in die entsprechenden Bildern verknüpft, positioniert und angepasst.



4. Sollte sich der Projektant dafür entscheiden das generelle Aussehen des Dialogs zu verändern, so führt er diese Änderung zentral im Symbol durch. Diese Änderung wirkt sich dann auf alle Positionen, in denen das Symbol verknüpft ist, aus.



5. Über die freigegebenen Eigenschaften und eine entsprechende Ersetzung (auf Seite 31) kann das globale Symbole individuell angepasst werden.



6. In diesem Fall hat sich der Projektant dafür entschieden, dass sich die generelle grafische Oberfläche von Parametrierungsdialogen ändern muss. Diese Änderung führt er zentral am Symbol „Parametrierung" durch. Die individuellen Eigenschaften der einzelnen Parametrierungsdialoge bleiben unverändert, es ändern sich nur jene Eigenschaften, die in der Vererbung enthalten sind.

5.2.2 Ersetzung bei Symbolen

Der im Abschnitt Variablen und Funktionen ersetzen (auf Seite 22) beschriebene Vorgang kann auch bei Symbolen durchgeführt werden. Auch hier ist eine durchdachte Benennung (auf Seite 22) der Objekte Voraussetzung.

Bei der Ersetzung für Symbole müssen zwei unterschiedliche Herangehensweisen unterschieden werden: Ein Symbol kann aus einer Bibliothek (Projekt/Global)

- in ein Prozessbild eingebettet werden.
- in ein Prozessbild verknüpft werden

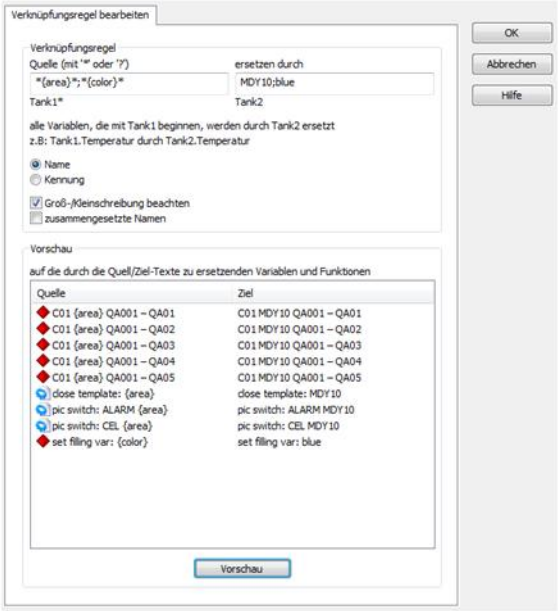
Je nach Herangehensweise werden unterschiedliche Dialoge zur Ersetzung der Variablen/Funktionen aufgeschaltet.

EINGEBETTETE SYMBOLE

Hier werden die Variablen direkt über den Ersetzungsdialog ersetzt. Der Dialog bietet mitunter die Möglichkeit, Variablen über einen Variablenauswahldialog auszuwählen. Symbole auf diese Art und Weise zu verwenden ist für die Wiederverwendung nicht ideal geeignet.

VERKNÜPFTE SYMBOLE

Bei diesem Verfahren wird über Ersetzungsstrings gearbeitet. Diese Strings werden bei der Kompilierung durch die entsprechenden Variablen ersetzt. Über den hier aufgeschalteten Dialog ist es nicht möglich Variablen über einen Variablen-Auswahldialog zu ersetzen. Der Projektant arbeitet hier über einzelne Ersetzungsstrings, die durch ein Semikolon (;) voneinander getrennt werden.



Das hinter diesem Ersetzungsdialog liegende Symbol kann nun in den unterschiedlichsten Anlagenbereichen wiederverwendet werden. Dieses Beispiel geht davon aus, dass in jedem Anlagenbereich ähnliche Variablen und Funktionen (Unterscheidung nur durch Anlagenkürzel im Namen) verwendet werden und diese so schnell und übersichtlich ersetzt werden können.

Mehr über Ersetzung bei Symbolen lesen Sie auch im Handbuch Bilder im Abschnitt Verknüpfte Symbole.

5.2.3 Globale Symbolbibliothek

Symbole werden entweder in der Projekt-Symbolbibliothek oder in der Globalen Symbolbibliothek verwaltet. Der wesentliche Unterschied:

Symbolbibliothek	Eigenschaft
Globale Symbolbibliothek	Symbole sind in allen Projekten verfügbar.Der Knoten **Globale Symbolbibliothek** befindet sich im Projektmanager unterhalb der aktuell geladenen Projekte Globale Symbole werden im zenon Programmordner gespeichert und nur beim Editorstart aktualisiert. Diese Symbole werden bei der Projektsicherung nicht gesichert.
Projekt-Symbolbibliothek	Symbole sind nur im aktuellen Projekt verfügbar. Die Symbole werden im Projektordner gespeichert. Die Projekt-Symbolbibliothek befindet sich im aktuellen Projekt im Knoten **Bilder** und wird bei einer Projektsicherung mitgesichert.

BEISPIEL

Eine zentrale Projektierungsstelle stellt eine Sammlung von Symbolen zusammen, die von den weltweit verteilten Projektierungsstellen in ihren individualisierten Projekten verwendet werden soll. Zum Beispiel werden komplexe Symbole zur Verfügung gestellt, die wichtige Teile von Prozessbildern darstellen. Diese Sammlung an Symbolen befindet sich in einem Ordner oder einer Datei, z.B. „`Global_Used_Symbols.SYM`". Werden zentral grafische Veränderungen an dieser globalen Symbolsammlung durchgeführt, so wird einfach die „`Global_Used_Symbols.SYM`" Datei an alle Projektierungsstellen versendet und dort ersetzt. Die individuelle Anpassung kann erfolgt über frei gegebene Eigenschaften (auf Seite 28) und Ersetzung (auf Seite 31).

Es ist nicht immer nötig, die gesamte Symbolsammlung zu verteilen. Um einzelne ausgewählte Symbole zu verteilen, kann auch der XML-Export/Import (auf Seite 38) genutzt werden.

Achtung

Bei Verwendung der Globalen Symbolbibliothek muss eine klare Regel bestehen: Lokale Projektierungsstellen dürfen die Symbole aus der Symbolsammlung nicht verändern. Jede Änderung würde beim nächsten Update überschrieben werden.

5.3 Struktur-Datentyp

Variablen können über Struktur Datentypen wiederverwendet werden. Damit lassen sich komplexe Strukturen aufbauen und die Vorteile einer Vererbung nutzen.

BEISPIEL

Der Struktur-Datentyp „Motor" besteht aus drei Elementen:

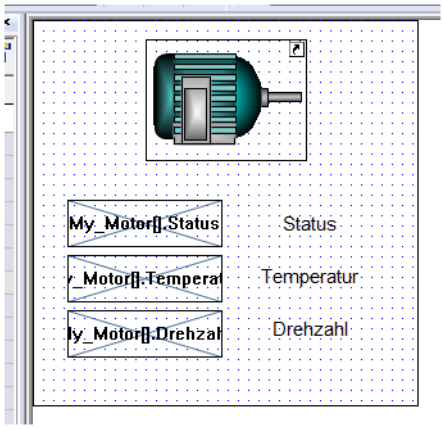
- Status
- Drehzahl
- Temperatur

−	Motor	Struktur-Datentyp	0
	Status	Struktur-Element	INT/<eingebettet> 2
	Drehzahl	Struktur-Element	INT/<eingebettet> 4
	Temperatur	Struktur-Element	INT

Die Elementen unterscheiden sich in ihrer Vererbungsart:

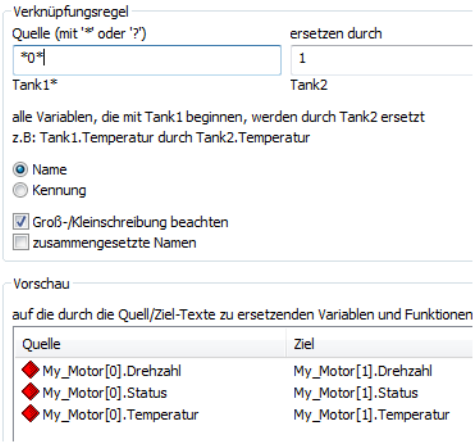
- „**`Status`**" und „**`Drehzahl`**" sind vom Datentyp `INT` abgeleitet, aber ihre Vererbung wurde durch das Einbetten aufgebrochen.
  D. h.: Spätere Aktualisierungen am `INT` Datentyp wirken sich nicht auf diese beiden Elemente aus.
- „**`Temperatur`**" stammt auch vom Datentyp `INT` ab, die Vererbung ist intakt.
  D. h.: Spätere Änderungen am `INT` Datentyp wirken sich auch auf die Einstellungen dieses Struktur-Elements aus.

Werden nun zum Beispiel 20 Motoren projektiert, deren Aufbau immer ident ist, können Variablen basierend auf dem selbst erzeugten Datentyp „Motor" erzeugt werden. Dies kann z.B. über das Anlegen eines Arrays von diesem Datentyp passieren. Die grafische Darstellung wird in diesem Beispiel über ein Symbol gelöst.



Beim Verknüpfen des Symbols in ein Prozessbild werden die Platzhalter-Variablen des Symbols durch jene des selbst erzeugten Struktur-Datentyps ersetzt. Achten Sie auch hier von Anfang an auf die durchdachte Benennung (auf Seite 22).

Gestaltet man ein Prozessbild als Übersicht über alle Motoren, kann man einfach über Copy&Paste das eingefügte Symbol vervielfachen. Es muss nur noch bei jedem Symbol die Zielvariable durch die entsprechende Laufnummer ersetzt werden:



Sollen zu einem späteren Zeitpunkt Grenzwerte für die „Drehzahl" Variablen hinzugefügt werden, profitiert man von der vorhandenen Vererbungsbeziehung zwischen dem selbst angelegten Datentyp und den daraus erzeugten Variablen. Für die geplante Änderung wird am Strukturelement „**`Drehzahl`**" des eigenen Datentyps „`Motor`" ein neuer Grenzwert angelegt. Diese an einer Position durchgeführte Änderung, wirkt sich auf alle 20 angelegte Motoren aus.

Dem Projektanten steht es aber auch durchaus frei diese Vererbungsbeziehung aufzulösen. Es kann also bei einer der 20 angelegten Motorvariablen der Grenzwert für „**`Drehzahl`**"geändert werden.

Hinweis: Es können einzelne oder alle Eigenschaften einer Variablen vom Datentyp entkoppelt werden.

> **Info**
>
> *Tipps zur Verwendung von Struktur-Datentypen lesen Sie auch im Abschnitt Struktur-Datentypen - Strukturelemente - Strukturarray (auf Seite 17) und im Handbuch Variablen im Abschnitt Struktur Datentypen.*

5.4 Reaktionsmatrizen

Reaktionsmatrizen bewirken, dass die Ausprägung einer Variable überall im Projekt gleich ist. Im Gegensatz zu Grenzwerten verfolgen sie einen zentralen Ansatz: Eine Reaktionsmatrix wird einmal projektiert und dann beliebigen Variablen zugeordnet. Alle Variablen, die mit der Reaktionsmatrix verknüpft sind, reagieren damit gleich. Der Vorteil: zentrale und einfache Wartung.

5.5 Globalprojekt

Mit einem Globalprojekt lassen sich Standards projektübergreifend festlegen. Damit ist es möglich, bestimmte Module oder Elemente für alle Standorte zu definieren, während lokale Projekte individuelle Parameter festlegen.

Im Globalprojekt können projektiert werden:

- Alarmierung:

  Deklaration der global zu verwendeten Alarmgruppen, Alarmklassen und Alarmbereiche.

- Anlagenmodellierung:

  Diese kann die zentrale Projektierung unterstützen, da man eine Aufteilung in zwei „Anlagen" vornehmen könnte. Dabei mus klar definiert werden, welche Projektierungselemente von der zentralen Projektierungsstelle kommen und welche von der lokalen Projektierungsstelle stammen.

- Benutzer:

  Sieht ein Standard die Existenz von bestimmten global gültigen Benutzern vor, so können diese über das Globalprojekt festgelegt werden.

- Dateien:

  Durch einen Standard können beispielsweise Grafiken vorgegeben sein, die in den individuell erstellten Projekten verwendet werden müssen. Diese können global zur Verfügung gestellt werden.

- Schablonen:

  Festlegen der Prozessbild-Anordnung und Größe der zu entwickelnden Projekte. In einer Richtlinie könnte beispielsweise klar festgelegt werden, dass in einem Projekt, neu angelegte Bilder nur über Schablonen eines Globalprojekts angelegt werden. Somit wird eine globale, grafische Einheitlichkeit gewährleistet.

- Schriften und Farbpaletten:

  Um eine Einheitlichkeit bei dargestellten Texten und Farben zu erreichen, kann ein vordefinierter Satz von Schriftlisten bzw. Farbpaletten zur globalen Verwendung festgelegt werden.

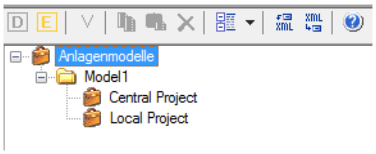
- Sprachtabellen:

  Wird durch einen Standard ein festgelegter Satz an zu verwendeten Begriffen und Aussagen in der Projektierung festgelegt, so ist es möglich diese über die definierten Sprachtabellen festzulegen und gleichzeitig die entsprechenden Übersetzungen durchzuführen.

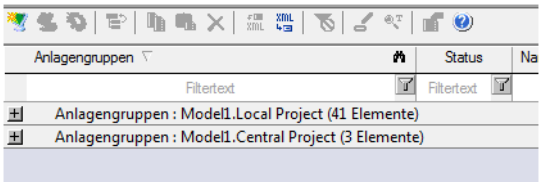
BEISPIEL

Eine zentrale Projektierungsstelle definiert Standards in den Bereichen Farben, Schriften, Schablonen usw. über ein Globalprojekt für alle Projektanten. Das Globalprojekt wird als Projekt-Backup (auf Seite 41) an alle Teams verteilt. Diese importieren es in ihren Arebitsbereich. Über eine Richtlinie muss klar festgelegt werden, dass lokale Projektierungsstellen keine Änderungen an einem Globalprojekt vornehmen dürfen, da diese immer wieder durch das Einspielen von Aktualisierungen der zentralen Projektierungsstelle überschrieben werden.

BEISPIEL ANLAGENMODELL



Ein einfaches Anlagenmodell, das nur eine Unterscheidung zwischen einer zentralen und einer lokalen Projektierungsgruppe trifft. In einem individualisierten Projekt könnte man sich diese Unterscheidung nun bei der Visualisierung im Editor zur Hilfe nehmen. Im Detail würde das so aussehen, dass jedes projektierte Elemente im Editor mit einer entsprechenden Anlagengruppen Zuordnung versehen wird.



Die vorhandenen Prozessbilder werden nach Anlagenzuordnung angezeigt.

Damit kann eine lokale Projektierungsstelle klar erkennen, welcher Bilder wieder von einer zentralen Projektierungsstelle über den XML Import aktualisiert werden. Für den lokalen Projektanten ist damit auch klar, welche Bilder er nicht verändern darf. Denn jede Änderung an global gewarteten Bildern wird beim nächsten Update wieder überschrieben.

Tipp: Für die Aktualisierung eines Globalprojekts muss nicht immer ein komplettes Projekt-Backup erstellt und importiert werden. Einzelne Änderungen können auch über den Export und Import per XML (auf Seite 38) lokal aktualisiert werden.

5.6 XML

Der Import und Export über die XML-Schnittstelle bietet vielfältige Möglichkeiten zur Wiederverwendung von Projektteilen.

> **Achtung**
>
> *Beim Import werden vorhandene Elemente mit gleichem Namen überschrieben.*

ABHÄNGIGKEITEN BEIM IMPORT

Wenn mehrere Projektteile importiert werden sollen, gibt es dementsprechend auch mehrere XML Dateien mit unterschiedlichen Inhalten. Um alle Inhalte mehrerer XML-Dateien korrekt zu übernehmen, müssen vorhandene Abhängigkeiten berücksichtigt werden:

Wird zum Beispiel ein Bild importiert und die darin verknüpften Variablen existieren im Projekt nicht, dann kann die Verknüpfung nicht wiederhergestellt werden. Um alle Verknüpfungen korrekt anzulegen, kann es nötig sein, den Import eines Bildes oder von Variablen zweimal durchzuführen.

BEISPIEL FUR BILD XML:

Die umfangreichste Variante ist, ein Bild zu exportieren, da die erzeugte XML-Datei nicht nur das Bild und die enthaltenen Elemente sondern auch die Schablone, verknüpfte Variablen und Funktionen beinhaltet, also alles, was im Bild zu sehen und direkt verknüpft ist.

Im Bild „**A**" wird mittels Button eine Funktion „**B**" ausgeführt, um wiederum das Bild „**A**" zu öffnen:

- Die Funktion „**B**" benötigt das Bild „**A**", damit dieses in der Funktion „**B**" verknüpft wird.
- Das Bild „**A**" wiederum benötigt die Funktion „**B**", damit diese am Button verknüpft wird.

Man muss also:

1. erst die Funktion importieren,
2. danach das Bild und dann
3. ein weiters mal dieselbe Funktion, da andernfalls der Bezug zum Bild in der Funktion nicht hergestellt werden kann.

PROJEKTE WERKSÜBERGREIFENDE VEREINHEITLICHEN

Der XML-Export/Import eignet sich auch dazu, um die werksübergreifende Einheitlichkeit von Projekten zu gewährleisten. Legt ein Unternehmen fest, dass bestimmte Projektteile weltweit einheitlich ausgelegt sein müssen, dann werden die einheitlichen Elemente:

1. zentral projektiert
2. in eine XML Datei exportiert
3. als XML-Datei verteilt
4. von jedem lokalen Team per XML-Import übernommen

Alternativ kann der Export und Import auch über einen eigenen Wizard erfolgen.

Da beim XML-Import bestehende Elemente überschrieben werden, müssen diese einheitlichen Elemente klar und verbindlich benannt werden. Lokale Änderungen dieser Elemente werden beim nächsten Import wieder überschrieben.

WIZARDS (AUF SEITE 40)

Um den Benutzer beim Importieren zu unterstützten, bietet sich ein auf der zenon API basierende Wizard an. Im Wizard kann der Benutzer dann an Hand von Entscheidungshilfen auswählen, was importiert werden soll. Ein möglicher Ansatz für diesen Wizard ist, sich eine XML-Bibliothek anzulegen, die verschiedene Varianten von Bildern und Funktionalitäten als einzelne Dateien beinhaltet.

INDIVIDUALISIERUNG PER WIZARD

Über einen Wizard können auch Standardfunktionen, die im XML-Format vorliegen, individualisiert werden. Es wird eine Funktion per XML importiert und im Anschluss über die Änderung von bestimmten Eigenschaften, die nicht für jedes Objekt gleich sind, an die individuellen Anforderungen angepasst. Zum Beispiel kann eine Bildumschaltfunktion im XML-Format vorliegen und über eine Individualisierung der Eigenschaften mehrfach verwendet werden.

5.7 Wizards

Die Wiederverwendung von Elementen und Teilen von Projekten kann mit Wizards vereinfacht und unterstützt werden. Wizards eigen sich zum Beispiel vor allem für:

- Erstellung von Vorlage-Projekten:

  Ein Wizard ermöglicht es, bestimmte Standardteile eines neu zu erstellenden Projekts mittels weniger Mausklicks zu erzeugt. Im Gegensatz zum simplen Einspielen einer fertigen Projektsicherung (auf Seite 41) des Vorlage-Projekts, gibt man hier dem Projektanten mehr Spielraum zur Individualisierung.

  Zum Beispiel könnte ein Wizard maschinenabhängig unterschiedliche Projekttypen erzeugen.

- Erstellung von vordefinierten Projektteilen:

  Wizards können auch helfen, bestimmte Bereiche in einem Projekt zu erstellen.

  Zum Beispiel: Anlegen eines Projektbereichs „`Motor`".
  Der Wizard legt die für diesen Bereich vordefinierten Elemente an. Damit können mit wenigen Mausklicks können entsprechende Bilder, Funktionen, Variablen usw. projektiert werden. Neben den kürzeren Projektierungszeiten erreicht man gleichzeitig ein einheitliches Aussehen, da die Wizards von einer zentralen Projektierungsstelle programmiert und von den lokalen Projektierungsstellen verwendet werden.

- Projektierung über vordefinierte Datenbanken bzw. Dateien:

  Auch Projektdatenbanken oder vordefinierte Dateien können die Projekterstellung oder Projekterweiterung unterstützen. Dabei werden in einer Datenbank oder in einer Excel-Datei die zu projektierenden Elemente beschrieben (Funktionsname, Funktionstyp, Funktionsparameter, Variablenname, Offset …). Mit Hilfe eines Wizards werden diese ausgelesen und entsprechende Projektierungen in zenon vorgenommen.

Mehr über Wizards und deren Erstellung lesen Sie im Handbuch Wizards.

5.8 Projekte wiederverwenden

Projekte könne auch unterschiedliche weise wiederverwendet werden. Dabei spielen Mechanismen aus den anderen Kapitel der Wiederverwendbarkeit teilweise eine wesentliche Rolle.

Projektsicherung (auf Seite 41)

Speichern unter (auf Seite 42)

Mehrprojektverwaltung (auf Seite 43)

5.8.1 Projektsicherung

Projektsicherungen ermöglichen es, komplette Projekte zu übernehmen.

Zum Beispiel kann ein Projekt als Vorlage an alle Teams verteilt werden. Diese spielen die Projektsicherung ein und individualisieren das Projekt.

> **Achtung**
>
> *Diese Methode eignet sich nicht zum Übertragen von Projektänderungen, da bei der Übernahme sämtliche lokale Projektierungen überschrieben werden.*

Um Projekte, die auf diese Weise erstellt wurden teamübergreifend zu warten, empfiehlt sich die Verwendung des XML-Exports/Imports (auf Seite 38).

Beachten Sie dabei, dass bereits vorhandene Elemente mit gleichem Namen beim Import überschrieben werden! Bei der Projektierung muss also klar gekennzeichnet werden, welche projektierten Elemente beim Import überschrieben werden dürfen. Für die dazu nötige Identifikationeigent sich zum Beispiel ein entsprechender Zusatz im Elementnamen oder die Vorgehensweise über Anlagengruppen siehe Abschnitt Globalprojekt (auf Seite 36).

5.8.2 Speichern unter

`Speichern unter` erstellt im aktiven Arbeitsbereich eine Kopie eines Projekts unter neuem Namen und mit neuer GUID.

BEISPIEL

Auf Basis eines Vorlage-Projekts sollen individuelle Projekte für unterschiedliche Maschinentypen erstellt werden. Dazu:

1. wird die Projektsicherung (auf Seite 41) des Vorlage-Projekts eingespielt
2. werden über **`Speichern unter`** im aktiven Arbeitsbereich mehrere unterschiedliche benannte Projekte erstellt (z.B. **`Maschine 1`**, **`Maschine 2`**, …)

Damit hat jedes Maschinenprojekt dieselbe Projekt-Ausgangssituation. Die Projekte haben allerdings keinerlei Verbindung mehr zum Vorlage-Projekt. Änderungen, die im Vorlage-Projekt durchgeführt werden, werden nicht automatisch in die kopierten Projekte übernommen. Änderungen in der Vorlage können aber per XML Export-/Import (auf Seite 38) in die individualisierten Projekte eingespielt werden.

Beachten Sie dabei, dass bereits vorhandene Elemente mit gleichem Namen beim Import überschrieben werden! Bei der Projektierung muss also klar gekennzeichnet werden, welche projektierten Elemente beim Import überschrieben werden dürfen. Für die dazu nötige Identifikationeigent sich zum Beispiel ein entsprechender Zusatz im Elementnamen oder die Vorgehensweise über Anlagengruppen siehe Abschnitt Globalprojekt (auf Seite 36).

5.8.3 Mehrprojektverwaltung

Projekte können in der zenon Mehrprojektverwaltung auch in einem Arbeitsbereich unter dem Dach eines Integrationsprojekts zusammengefasst werden. So kann ein Projekt zentral verändert werden, während lokale Projektierungen in einem anderen Projekt vorgenommen werden.

BEISPIEL

Über eine zentrale Projektierungsstelle solle die grafische Wiederverwendung in lokalen Projekten realisiert werden, um weltweit eine einheitliche Oberfläche sicherzustellen. Dazu erstellt die zentrale Projektierung ein Vorlage-Projekt, das die grafische Grundlage für alle anderen Projektierungsteams darstellt.Wird aber an der zentralen Stelle eine Änderung der Oberflächengestaltung durchgeführt, so muss diese an allen Projektierungsstellen nachgezogen werden. Das kann über die Mehrprojektverwaltung erfolgen: ein Projekt enthält die zentral zu verwaltenden Elemente, ein anderes Projekt die lokale Projektierung.

Vorgangsweise:

1. initial wird eine Vorlage-Projekt an alle Projektierungsteams geschickt
2. dieses wird in den aktiven Arbeitsbereich eingespielt wird
3. es wird lokal ein zweites, leeres Projekt angelegt, das die spezifischen Anpassungen enthält
4. im Arbeitsbereich existieren jetzt zwei Projekte: das Vorlage-Projekt und das individualisierte Projekt
5. im nächsten Schritt erfolgt die multi-hierarchische Anordnung der Projekte
6. das Vorlage-Projekt wird zum Integrationsprojekt, das individualisierte Projekt zum Unterprojekt

> **Info**
>
> *Diese Methode ist für Projekte, die unter Windows CE laufen sollen, nicht geeignet, da unter CE immer nur ein Projekt gestartet werden kann.*

6. Integriertes Netzwerk

Verteiltes Engineering im Netzwerk und 100% Redundanz sind mit zenon kein besonderer Aufwand mehr für Sie. Netzwerke erstellen Sie durch Aktivieren der entsprechenden Checkbox bei den Projekteigenschaften. Und genau so einfach machen Sie aus vereinzelten stand alone Projekten effizient vernetzte, redundante Systeme.

Wenn Sie über ein TCP/IP-Windows-Netzwerk verfügen, stehen Ihnen in zenon automatisch Netzwerkfunktionen auf Mausklick zur Verfügung, als Client/Server Modell oder als multihierarchisches System mit Unterstationen, Bereichszentralen und Zentralen. Sie

- erstellen Projekte auch mit mehreren Anwendern gleichzeitig im Netzwerk
- sichern Anlagen durch die 100% zenon Kreisredundanz perfekt ab
- bauen problemlos verteilte Systeme auf
- greifen remote auf Stationen zu
- überwachen und steuern Anlagen per Web-Server
- sehen Prozessdaten auf allen Stationen in Echtzeit
- machen Aktionen eines Arbeitsplatzes auf allen anderen sichtbar, z. B. das Quittieren von Alarmen
- lassen Aktionen zentral protokollieren und archivieren
- nutzen Prozessdaten sofort für ERP-Systeme wie SAP

Um die unbedingt nötige Zeitsynchronität aller beteiligten Rechner kümmert sich zenon automatisch.