LEUPHANA
UNIVERSITÄT LÜNEBURG

Diplomarbeit

August 2007

André Sperling

Matr.-Nr.:1156597

Thema:

Entwicklung eines Software-Werkzeugs, zur intuitiven Projektierung von Datenpunkten der Prozessvisualisierung.

Fakultät III

Umwelt und Technik

Angewandte Automatisierungstechnik

1. Prüfer: Prof. Dr. rer. nat. H. Faasch
2. Prüfer: Prof. Dr.-Ing. E. C. Bollow

1 Übersicht

1.1 Kurzfassung

Entwicklung eines Software-Werkzeugs, zur intuitiven Projektierung von Datenpunkten der Prozessvisualisierung.

Stichwörter: OPC, iFix™, Prozessvisualisierung, VBA, Visual Basic for Application

In dieser Diplomarbeit ist die Entwicklung eines Software-Tools zur Erzeugung von Datenpunkten in einem Prozessvisualisierungssystem beschrieben. Die Software beschränkt sich zum einen auf iFIX™ als Visualisierungssoftware und zum anderen auf Visual Basic for Applikation als Programmiersprache. Mit Hilfe von diesem Tool werden alle nötigen Tags sowohl in der Datenbank als auch im OPC-Client von iFIX™ erzeugt. Die Erzeugung dieser Tags ist auf die Standardelemente begrenzt. Die Erstellung dieser Verknüpfungen soll mit möglichst wenig Wissen über die Anlage selbst und über den vor Ort herrschenden Standard erfolgen. Zusätzlich soll die Bedienung des Tools intuitiv und praxisnah sein.

Die entwickelte Anwendung soll in den von der GEA TBS gebauten Anlagen eingesetzt werden, um dem Anwender des Programms das Verknüpfen von neuen Bauteilen in einem bestehenden Prozessvisualisierungssystems zu erleichtern und dadurch Entwicklungszeit zu sparen.

1.2 Abstract

Development of a Software-Tool for the intuitive projecting of data points in a processvisualizationsystem.

Keywords: OPC, iFix™, prozessvisualization, VBA, Visual Basic for Application

This diploma thesis described the development of a software Tool. This tool generates data points in a processvisualizationsystem. On the one hand the software uses iFIX™ as visualization-software and on the other hand Visual basic for application as programming language. This Tool crates all necessary tags in the data base and in the OPC-Client with iFIX™. The creation of the tags is limited to the standard elements of Tuchenhagen. The parameterization of these data points has to be made with a minimum of knowledge about the installed system and the locally dominant standard. The using of the Tool has to be intuitive and with a practical orientation.

The developed application should be used in the arrangements built by GEA TDS to save development time and simplify the connection of new construction units in an existing processvisualizationsystem.

1.3 Inhaltsverzeichnis

1.4 Einleitung

In der vorliegenden Diplomarbeit habe ich mich mit meiner praktischen Tätigkeit in der Firma GEA Tuchenhagen Brewery Systems GmbH befasst. Grundlage ist hier der Vertrieb und die Herstellung von Prozessanlagen in der Getränkeindustrie, deren Abläufe rechnergestützt visualisiert werden. Das bedeutet, dass man mit geeigneter Software die Anlagen projektieren und die Prozesse am PC darstellen kann.

Von Vorteil an der Visualisierung ist, dass es möglich ist, den Status der Anlage jederzeit auf dem Rechner anzuzeigen und zu überprüfen. Dies dient nicht nur der Kostenkontrolle bzw. der Kostenminimierung, sondern es können auch Fehler, die bei manueller Bedienung auftreten, reduziert werden. Zum anderen hat man, neben der Möglichkeit der Chargenrückverfolgung, auch die Option eines Meldungssystems. Bei Fehlern in der Anlage informiert dieses beispielsweise automatisch das Wartungspersonal.

Alle planenden Arbeitsvorgänge, von der Angebotsphase bis hin zur vollständigen Inbetriebnahme der Steuerung bzw. Anlage, benötigen einen geraumen Zeit- und Personalaufwand. Um hier Kosten und Zeit zu sparen sowie Fehler zu vermeiden, ist es notwendig, mit Automatismen zu arbeiten. Darum beschäftigt sich diese Studienarbeit mit der Projektierung der Visualisierungssoftware. Diese benötigt Daten aus der Steuerung, um den aktuellen Prozess darstellen zu können.

Es wurde ein Software-Tool zur Erzeugung von Datenpunkten in einem Prozessvisualisierungssystem entwickelt, mit dessen Hilfe sich alle nötigen Tags, sowohl in der Datenbank als auch im OPC-Client, von iFIX™ erzeugen lassen. Die Erstellung dieser Verknüpfungen soll mit möglichst wenig Wissen über die Anlage selbst und über den vor Ort herrschenden Standard erfolgen.

Durch den Einsatz der entwickelten Anwendung, in den von der GEA TBS gebauten Anlagen, wird dem Anwender des Programms das Verknüpfen neuer Bauteile in einem bestehenden Prozessvisualisierungssystem erleichtert. Fehler lassen sich so reduzieren und auch die Entwicklungszeit kann enorm reduziert werden.

1.5 Beschreibung der Firma

Tuchenhagen Brewery System ist eine Tochterfirma der GEA Group. Innerhalb der GEA Group gehört sie zur Process Engineering Division (P-Division).

1.5.1 GEA Group

1920 gegründet als Gesellschaft für Entstaubungsanlagen ist die GEA Group heutzutage ein weltweit erfolgreicher Technologiekonzern mit operativen Unternehmen in rund 50 Ländern. GEA steht heute für Global Engineering Alliance. Nach einer im Jahre 2003 begonnenen, umfassenden strategischen Neuausrichtung liegt der operative Fokus des Unternehmens auf dem Bereich Spezialmaschinenbau. In 90 Prozent der Geschäftsfelder zählt die GEA Group zu den Markt- und Technologieführern. Sie profitiert dabei von einer ausgeprägten Innovationskultur. Als global tätiges Unternehmen generiert die GEA Group heute rund 80 Prozent des Umsatzes außerhalb Deutschlands und ist dabei präsent in allen Wachstumsregionen der Welt. [GEAG 07]

Im Geschäftsjahr 2006 erwirtschafteten rund 17.500 Mitarbeiter einen Konzernumsatz von 4,3 Milliarden Euro.

Die GEA Group Aktiengesellschaft ist im MDAX gelistet (WKN: 660 200, ISIN: DE0006602006).

1.5.2 Tuchenhagen Brewery System

Tuchenhagen Brewery Systems ist ein international etablierter Spezialist für technisch und wirtschaftlich optimierte Prozessanlagen in der Brauindustrie.

In enger Zusammenarbeit mit den Kunden findet Tuchenhagen Lösungen, die umsetzbar und finanzierbar sind. Als Prozessintegratoren innerhalb der GEA Group bieten sie fundiertes Fachwissen und umfangreiche Erfahrung im Engineering sowie exzellente Kenntnisse der Anlagen und einzelnen Komponenten.

Tuchenhagen ist ein starker und zuverlässiger Lieferant von Prozesskomponenten für die Brau-, Getränke-, Milch- und Nahrungsmittelindustrie. Engineering und projektbezogene Integrationsdienstleistungen sind seit jeher ein wichtiger Bestandteil des Leistungsspektrums des Unternehmens. Zum Ausbau dieses Geschäftsfeldes wurde 1998 Tuchenhagen Brewery Systems gegründet. Das Unternehmen ist ein Kompetenzzentrum für die Brauereiindustrie, für alle Anwendungen von einzelnen Aggregaten über System-Engineering bis hin zu Komplettanlagen.

Tuchenhagen Brewery Systems hat ihren Sitz in Büchen Nähe Hamburg, in direkter Nachbarschaft zu den Produktionsstätten, in denen die Komponenten für die Flüssigkeitsverarbeitung gefertigt werden. Diese unmittelbare Nähe bietet große Vorteile hinsichtlich Informationsaustausch und Know-how-Transfer. [Tuch 07]

1.5.3 Die Abteilung

Die Automatisierungsabteilung der Firma Tuchenhagen Brewery Systems GmbH ist für die Konzeptionierung und Installation von Visualisierungssoftware und die dazugehörige Hardware zuständig.

Zu den Aufgaben gehört die Unterstützung beim Vertrieb, die Planung, Projektierung und die anschließende Installation von kundenspezifischer Soft- und Hardware. Wichtig ist hierbei der ständige Dialog mit dem Kunden, um dessen Wünsche und Vorstellungen bestmöglich erfüllen zu können.

Um einen reibungslosen Ablauf gewährleisten zu können, ist es notwendig, sowohl das Bedienungspersonal in die neu installierten Programme einzuführen als auch interne Schulungen in der Firma Tuchenhagen durchzuführen. Auf diese Weise soll ein bestmöglicher Standard erreicht werden. Dies ist ein weiterer Aufgabenbereich der Abteilung.

Sind Soft- und Hardware erfolgreich installiert und abgenommen, leistet die Abteilung weiterhin Support für die laufenden und abgewickelten Projekte der Kunden. Hierbei stehen Störungsbehebung, Unterstützung und Beratung bei möglicherweise auftretenden Problemen im Vordergrund. Hier kann zum Beispiel auf die von der Abteilung erstellte Dokumentation zurückgegriffen werden. Wenn nötig ist auch eine persönliche Beratung gewährleistet.

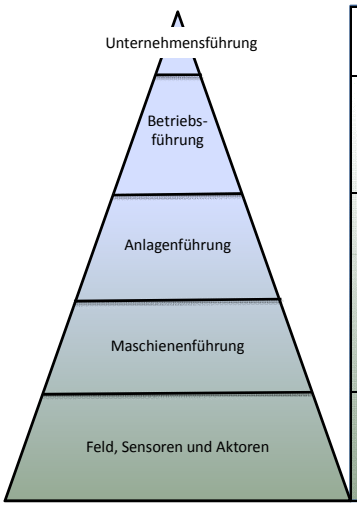
Um den neusten Stand der Technik erfüllen zu können, ist die Weiterentwicklung der Soft- und Hardwarekomponenten von Bedeutung. Beispielsweise muss die neuste Software eingekauft, weiterentwickelt und integriert werden.

2 Grundlagen

2.1 Theoretische Grundlagen

2.1.1 Strukturierung der Automatisierungsaufgaben

Ein bewährtes Prinzip bei der Führung komplexer Systeme, sei es in technischen oder auch organisatorischen Hinsicht, besteht in der Einführung einer Hierarchie von Entscheidungs- und Ausführungsebenen. Entsprechend diesem Prinzip gehört die Gliederung von Automatisierungsaufgaben in hierarchische Ebenen zu den allgemein anerkannten Grundkonzepten(siehe Abbildung 1). [Lauber&Göhner 99b]



Unternehmens-leitebene	Langfristige Produktionsplanung, Kostenanalysen, Auftragsvergabe und -abwicklung, Statistiken
Produktions-leitebene	Betriebsablaufplanung, Kapazitätsplanung, Terminüberwachung, Auswertung der Prozessergebnisse, Qualitätssicherung
Prozess-leitebene	Prozessüberwachung, Fehlerdiagnose, Prozessoptimierung, An- und Abfahren, Störungsbehandlung, Prozesssicherung
	Steuern, verriegeln, regeln, notbedienen, Grenzwert-überwachung, Schutz, anzeigen, registrieren
Feldebene	Messen der analogen und binären Prozessgrößen, Stelleingriffe über Motoren, Ventile usw., Automatik/Hand Umschaltung, anzeigen vor Ort

Abbildung 1: Prozessleitebenen

- In der untersten Ebene, hier als Feldebene bezeichnet, geht es um die Erfassung und Beeinflussung einzelner Prozessgrößen. Diese Beeinflussung kann im Sinne einer Steuerung oder einer Regelung erfolgen.
- Die darüber liegende Ebene beinhaltet die sogenannten gehobenen Prozessleitfunktionen, die zum Führen von Anlagen, Verfahrensgruppen, Apparaten usw. notwendig sind. In dieser Ebene ist auch die Prozessvisualisierung anzusiedeln.
- Die Produktionsleitebene – auch als Betriebsleitebene bezeichnet – umfasst Funktionen, die zum Führen von Betriebsteilen oder ganzen Fabriken gebraucht werden.
- Auf der obersten Ebene der Unternehmensführung geht es um langfristig wirkende Entscheidungen, wie etwa die Produktionsplanung. [Lauber&Göhner 99a]

2.1.2 Die OPC - Schnittstelle

OPC, die Standardschnittstelle der Zukunft, steht für **O**LE for **P**rocess **C**ontrol und basiert auf dem Komponentenmodell der Firma Microsoft. Der Begriff OLE "**O**bject **L**inking and **E**mbedding" wurde von Microsoft zeitweise für die gesamte Komponenten-Architektur verwendet. Heute spricht man jedoch von COM, dem **C**omponent **O**bjekt **M**odel, als Bezeichnung für die Komponenten-Architektur. COM ist ein zentraler Bestandteil der Microsoft-Betriebssysteme Windows 98 und Windows NT und stellt die Umgebung zur Zusammenarbeit von Software-Komponenten bereit.

Ziel ist es, ein einheitliches Softwareinterface zu schaffen, das auf der bekannten Microsoft Technologie (COM/OLE, nicht zu verwechseln mit der Rechner-Schnittstelle Com-Port) aufgebaut und somit für den Anwender einfach zu nutzen ist. So können "Echtzeit-Server", wie DCS, API und Feldgeräte, ihre Daten an OPC-Client-Anwendungen übertragen.

Dabei steckt die OPC-Schnittstelle vollständig in der Software, die auf einem PC als Plattform für Bedien- und Beobachtungssysteme oder andere Anwendungen läuft. Sie liegt unterhalb des Anwendungsprogramms und ist vollständig durch Software implementiert. OPC konkurriert also nicht mit Bussystemen, wie dem PROFIBUS, sondern verbindet Anwendungsprogramme und Baugruppentreiber auf einem Computer miteinander.

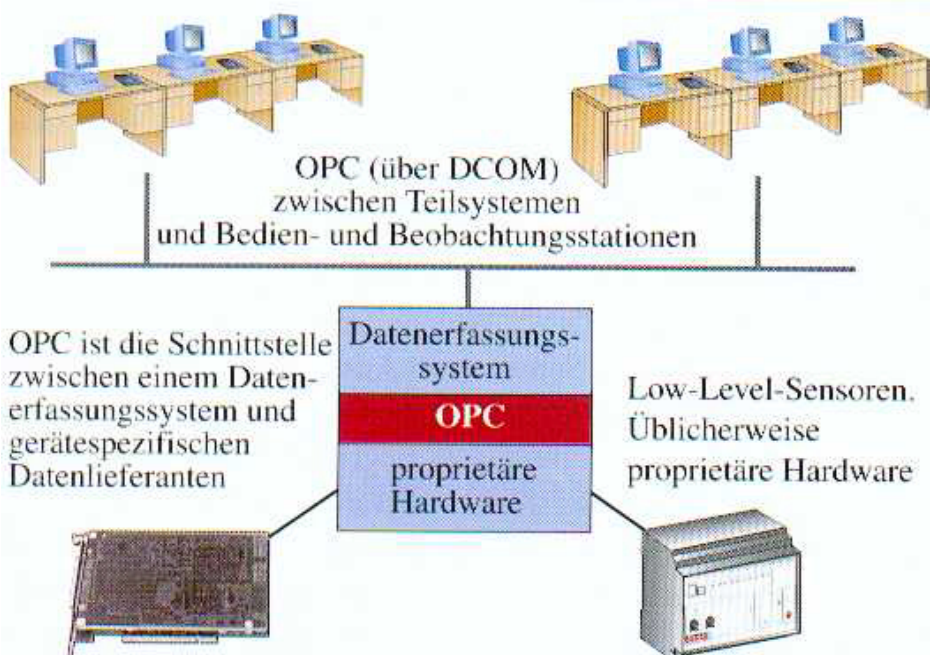


Abbildung 2: OPC Schnittstelle **[Langmann 07]**

2.1.3 Scan, Alarm and Control Program

Das Programm zur Datensammlung, Alarmierung und Steuerung (Scan, Alarm and Control, SAC) ist eine Systemanwendung, die auf einer Leitsystem-Station ausgeführt wird. Dieses Programm ist für die Ausführung einer logischen Verkettung von Datenbankblöcken verantwortlich und reguliert somit die gesamte Datenbank. Dieses Programm führt folgende Funktionen aus:

- Datensammlung von einer Reihe von Datenquellen
- Übersetzung der Daten in das von der Datenbank erwartete Format
- Prüfung der Daten auf eine Überschreitung der Alarmgrenzen und Erzeugung von Alarmmeldungen
- Ausführung von Steuerungslogik
- Erkennung von Ausnahmesituationen
- Durchführung der geforderten Schreiboperationen auf der Datenbank

Jede Datenbankverkettung enthält zusätzlich Informationen, ob SAC mit einer festen Zykluszeit oder beim Auftreten von Ausnahmesituationen nur einmal abgearbeitet wird. Dabei kann eine Verkettung sowohl mit einer festen Zykluszeit als auch beim Auftreten von Ausnahmesituationen ausgeführt werden. Auf diese Weise kann für die einzelnen Datenpunkte jeweils die am Besten passende Verarbeitungsstrategie genutzt werden. [iFix 03]

Bei Tuchenhagen haben die Datenbankverkettungen eine marginale Bedeutung. Sie werden lediglich zur Verarbeitung der Reglerdaten verwendet, da die Anlage selbst über eine SPS gesteuert wird und möglichst alle Steuer- und Regelungssignale von dieser verarbeitet werden. Die Prozessvisualisierung soll hier nur als HMI dienen.

2.1.4 API

Application Programming Interfaces sind Schnittstellen für die Anwendungsprogrammierung. Eine API ist eine Summe von Routinen, die von einem Programm aufgerufen werden kann, um diverse Aufgaben zu erledigen. Nicht nur Betriebssysteme und Gerätetreiber besitzen eine API, mit deren Hilfe sie ihre Dienste einer Anwendung anbieten, sondern auch einige Anwendungen selbst. So auch der iFix™ für den OPC-Client und die Prozessdatenbank.

2.1.5 Standardelemente der Firma Tuchenhagen

Bei Tuchenhagen gibt es vier unterschiedliche Typen von Standardelementen, die in der Prozessvisualisierung dargestellt werden, da man sie realen Objekten zugeordnen kann (Abbildung 3). Dies ist nicht nur in der Prozessvisualisierung durch spezielle Methoden und Funktionen, sondern auch in der SPS so. So werden den einzelnen Objekten bestimmte Daten-(DB), Funktions-(FC), Systemfunktionsbausteine(SFC) und anwenderdefinierte Datentypen (UDT) zugeordnet.

Digitaler Input

Stellt binäre Signale dar, wie zum Beispiel eine Vollmeldung oder ein Endwertschalter.

Analoger Input

Entspricht kontinuierlicheren Prozessgrößen, wie zum Beispiel einem Durchfluss, einer Temperatur oder einem Füllstand.

VM

Ventile und Motoren stellen die Stellglieder in einem Prozess dar. Wobei es sich bei den Motoren meist um Pumpen handelt. Ventile können unterschiedlich viele Anschlüsse haben.

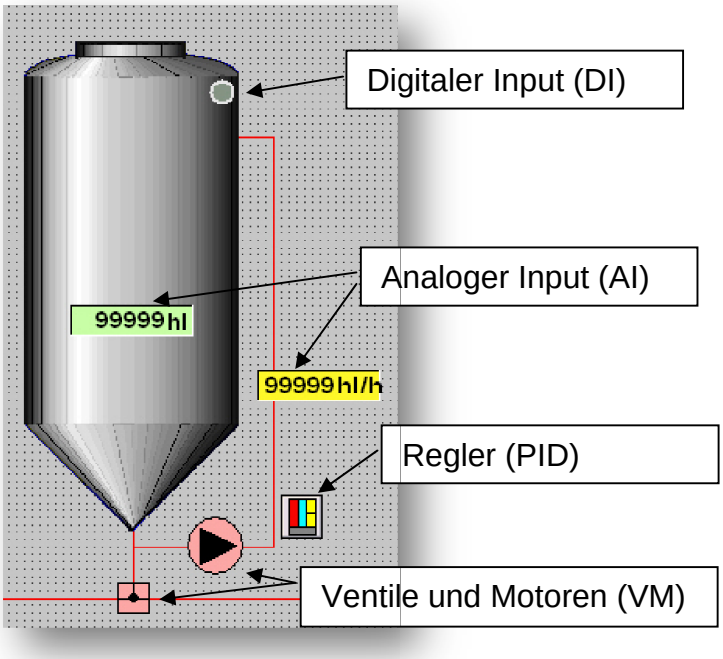


Abbildung 3: Standardelemente

Regler

Mit ihnen werden Prozessgrößen geregelt. Als Istwert dient zumeist ein analoger Eingang, der auf einen Sollwert geregelt wird. PID steht dabei für die Art des Reglers.

In der SPS gibt es noch weitere Standardbaugruppen, die hier jedoch nicht erwähnt werden, da sie für diese Diplomarbeit keine Relevanz haben.

2.1.6 Prozessvisualisierung mit dem Industrie-PC

Technische Prozesse – zum Beispiel von Produktionsanlagen, Kraftwerken oder großen Bürogebäuden – werden heutzutage meist mit Rechenanlagen überwacht und auf Monitoren dargestellt. Vorteile der Darstellung auf Grafikmonitoren gegenüber einem Übersichts- oder Anlagenbild in der früher üblichen Mosaiktechnik, mit Messschreibern, Anzeigen und Meldeleuchten, sind zum einen die Änderungsfreundlichkeit der Bilder und zum anderen eine umfangreichere Darstellung der Anlage in mehreren Monitorbildern, als dies sonst möglich wäre (Abbildung 4).

[Bitzer 91]

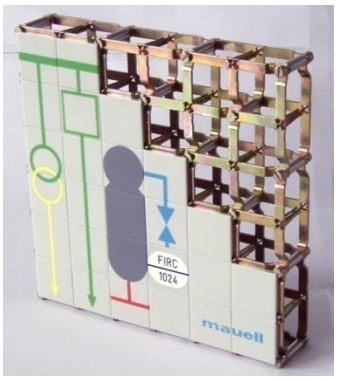


Abbildung 4: Mosaiksystem

2.1.7 iFIX™: Logischer Aufbau

In Abbildung 5 soll dargestellt werden, wie die einzelnen Softwarekomponenten untereinander interagieren. Dies ist eine deutlich vereinfachte Darstellung der Kommunikation und dient dem Verständnis des Gesamtsystems.

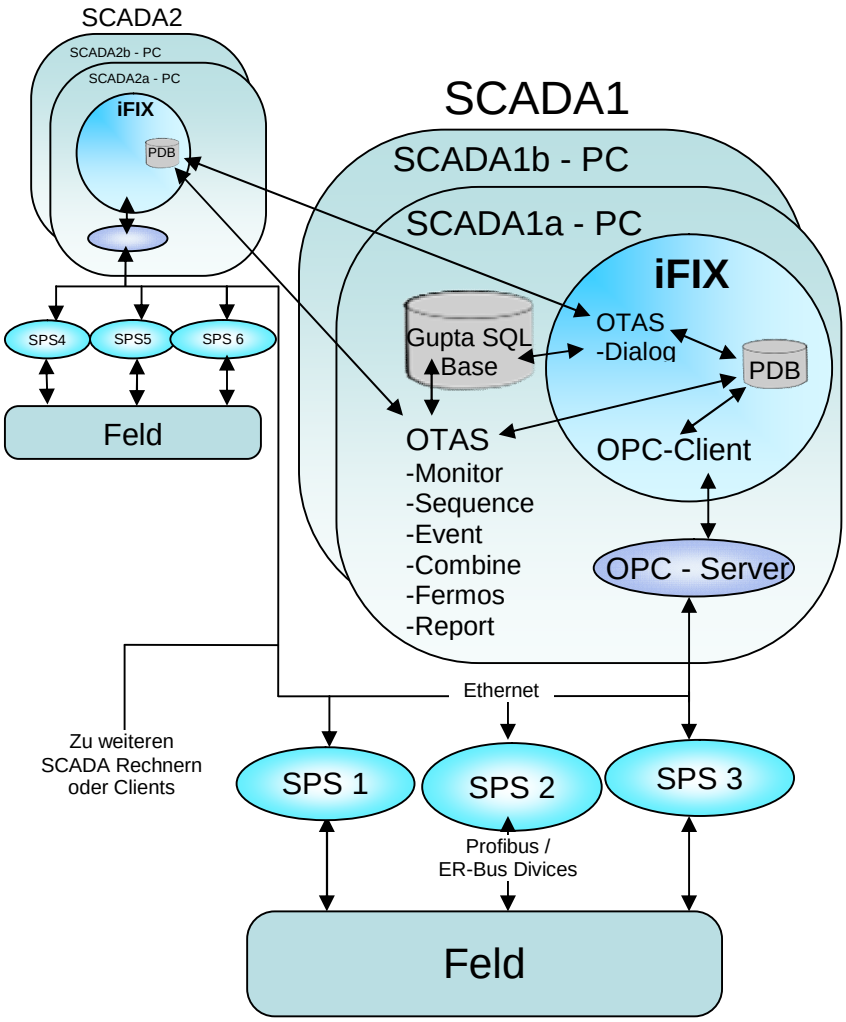


Abbildung 5: Logischer Aufbau iFIX™

Als Ergänzungsprodukt zu iFIX™ dient OTAS-Dialog zur Projektierung und Darstellung des Prozesses und der Eingabeoberflächen. OTAS-Dialog nutzt die Bordmittel von iFIX™.

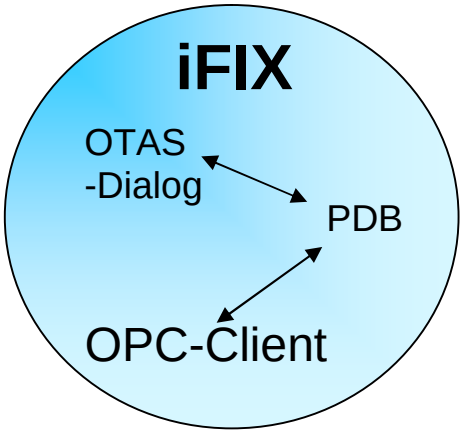


Abbildung 6: Kommunikation innerhalb von iFIX™

Bei der Projektierung greift OTAS-Dialog auf Templates zurück, bei denen VBA-Skripte im Hintergrund liegen, die die Objekte von iFIX™ parametrieren. Während iFIX™ sich in der Runtime befindet, werden diese Templates genutzt um den Zustand einzelner Bauteile darzustellen.

Die Daten dafür werden aus der lokalen Prozessdatenbank oder aus Prozessdatenbanken anderer SCADA-Knoten angefordert.

Diese Datenbanken sind keine relationalen Datenbanken, eher große Tabellen mit unterschiedlichen Objekten. Die Prozessdatenbanken erhalten die Prozessdaten vom OPC-Client (Power Tool), der wiederum zu unterschiedlichen OPC-Servern per DCOM kommunizieren kann. Bei dieser Installation interagiert der Client nur mit einem lokalen Server von der Firma INAT. Es gibt auch Installationen, in denen auf den OPC-Client und den OPC-Server verzichtet wird. In solchen Fällen kommt der sogenannte SIX-Treiber von Siemens zum Einsatz, der direkt mit der Prozessdatenbank und den Steuerungen kommunizieren kann.

Die restliche OTAS-Produktfamilie benutzt wie OTAS-Dialog die Prozessdatenbank, nur dass diese Anwendungen nicht auf iFIX™ basieren und deshalb über die EDA API auf die Prozessdatenbank zugreifen. Sie sind eigenständige Programme, die in Gupta programmiert worden sind. Gespeicherte Prozessdaten werden von diesen Programmen, wie auch von OTAS-Dialog, in einer SQL-Datenbank abgelegt und wenn nötig wieder abgerufen.

Dieses gesamte Softwarepaket läuft auf SCADA-Systemen. Diese sind redundante Rechnerpaare die meist bestimmten Bereichen in der Produktion zugeordnet sind, wobei beide Rechner als Bedienstationen fungieren können. Das heißt, dass beide Rechner Server sind und die Clients auf jeweils einen der Server zugreifen. Wenn einer der Rechner ausfällt, übernimmt der andere seine Aufgaben.

2.1.8 WinCC: Logischer Aufbau

Bei WinCC als Prozessvisualisierungssystem verhält es sich ähnlich wie bei iFIX™. Nur dass hier auf extra Treiber oder auf einen OPC-Server verzichtet werden kann, da WinCC die Möglichkeit bietet, direkt mit den Steuerungen zu kommunizieren. Die Prozessdatenbank ist nicht mit der PDB von iFIX™ zu vergleichen. Sie besteht aus mehreren Datenbanken. Da aber WinCC ähnlich wie iFIX™ ein API (ODK) bereit stellt, können die OTAS-Tools genauso auf die Prozessdaten zugreifen wie bei einem iFIX™ System.

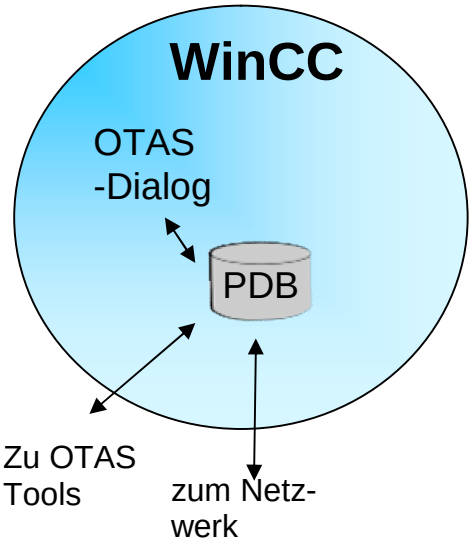


Abbildung 7: Kommunikation innerhalb von WinCC

2.1.9 Physikalischer Aufbau des Netzwerks

Dies ist ein Beispiel wie ein komplettes Prozessleitsystem, von der Prozessleitebene bis hin zur Regelungs- und Steuerungsebene, aufgebaut sein könnte. Die Bedienstationen greifen auf die Prozessdaten der Server zu, die wiederum ihre Daten von den Steuerungen erhalten. Die einzelnen Teilbereiche der Produktion sind über einen LWL-Ring verbunden. Diese Technik kommt hier aufgrund der hohen Reichweite und Bandbreite zum Einsatz.

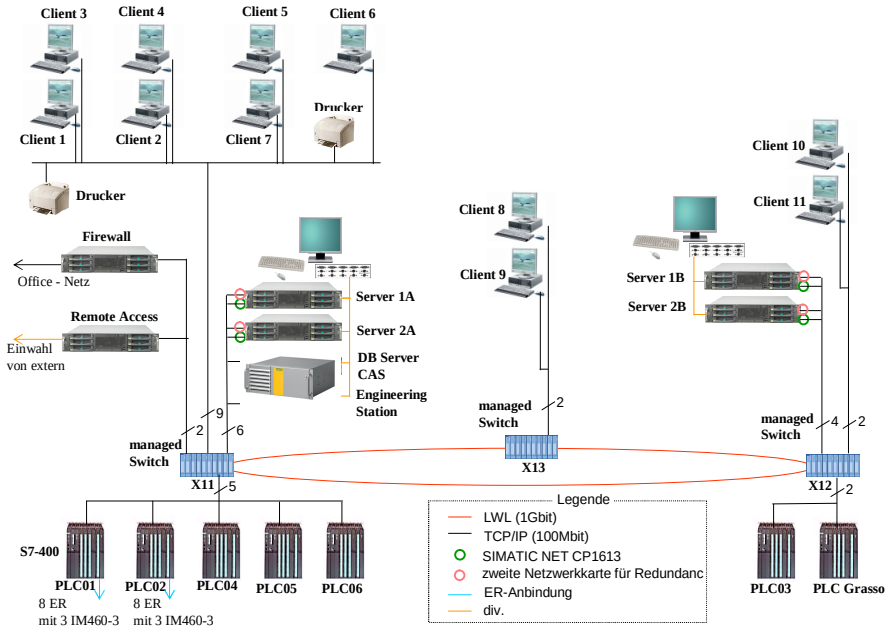


Abbildung 8 : Physikalischer Aufbau eines Prozessleitsystems

2.1.10 Initialisierungsdatei

In einer Initialisierungsdatei, auch INI-Datei genannt, werden Informationen für das Betriebssystem oder für ein Anwendungsprogramm gespeichert bzw. verwaltet. Zum Beispiel wird die Win.ini, System.ini oder die Desktop.ini von Windows zur Initialsierung genutzt.

```
[Gruppe01]
;Kommentar
Schlüssel_01=Schlüsselwert 1
Schlüssel_02=Schlüsselwert 2
[Gruppe02]
Schlüssel_01=Schlüsselwert 1
Schlüssel_03=Schlüsselwert 1
Schlüssel_04=Schlüsselwert 3
[Gruppe03]
…
..
.
```

Abbildung 9: Beispiel für INI-Datei

Diese Datei ist eine reine ASCII-Datei, die in Gruppen eingeteilt ist (Abbildung 9). Diese Gruppen bestehen aus Schlüsseln und Schlüsselwerten. Jeder Gruppenname darf innerhalb der Datei nur einmal vorkommen, genauso wie jeder Schlüssel nur einmal innerhalb einer Gruppe existieren darf. Zusätzliche Kommentare können mit einem Semikolon am Anfang der Zeile eingefügt werden. [Wilhelm 07]

2.1.11 Störcode

Der Störcode ist ein Bestandteil von OTAS-Steptext. Dieses Programm dient der Erzeugung von Texten in der Prozessdatenbank, wo das Automatisierungsgerät nur Zahlen oder Kurztexte liefert. Anhand einer dieser Zahlen, dem Störcode, lassen sich defekte Komponenten innerhalb eines Systems finden.

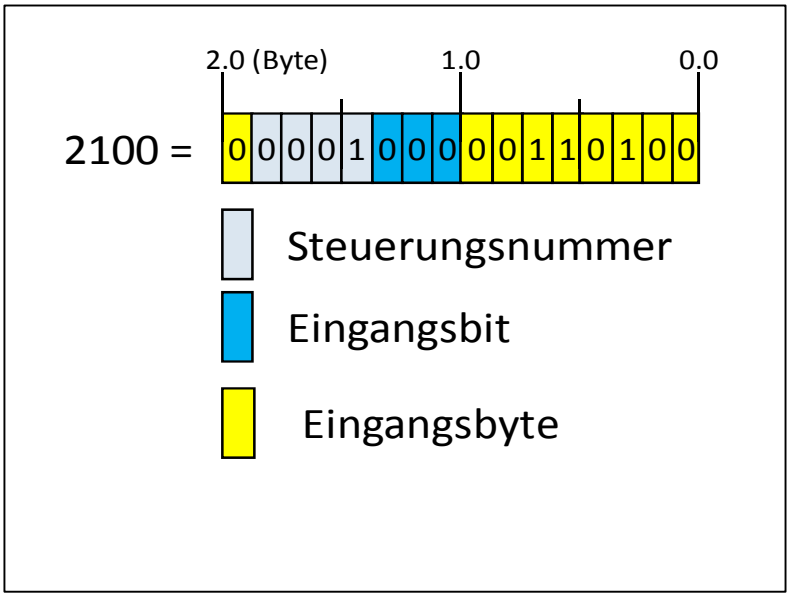


Abbildung 10: Zusammensetzung des Störcodes

Wenn die Steuerung beispielsweise die Zahl 2100 (Binär: 0000.1000.0011.0100) sendet, wandelt Steptext die Zahl um und erzeugt dann eine entsprechende Fehlermeldung sowohl in der Visualisierungsoberfläche als auch im Meldeprotokoll (Abbildung 10).

Die Fehlermeldung könnte dann beispielsweise lauten: "VM-Störung AG01: GT1_1_MV1(52.0)". Daraufhin wird der Name angezeigt, wenn er in der Steptext-Datei hinterlegt ist.

2.2 Ausgangssituation

Zur Erzeugung beziehungsweise Änderung der Tags in der Datenbank sowie im OPC Client, gibt es die unterschiedlichsten Möglichkeiten. Hier sind einige aufgezählt.

- Jeder Tag wird einzeln in der Datenbank und im OPC-Client angelegt oder geändert.
- Aus bestehenden Anlagen können die Tags mit Hilfe des Databasemanagers und des Powertools (OPC-Client siehe S.60) exportiert und in ein neues Projekt importiert werden. Bei Bedarf können die Tags auch mit Excel nachbearbeitet werden. In diesem Fall kann auch OTAS_Dialog_TOOL genutzt werden, wobei die neu generierten Daten dann einfach an die exportierten Daten angehängt werden können.
- Beim Erzeugen eines neuen Projektes werden CSV-Dateien mit Hilfe des OTAS_Dialog_TOOLs generiert und dann mit dem Databasemanager und dem Powertool importiert.
- Der Databasemanager bietet die Möglichkeit, einzelne Tags zu duplizieren und danach entsprechend den jeweiligen Anforderungen anzupassen.
- Mit dem Databasemanager ist es auch möglich, Tags automatisch zu generieren. Zunächst muss dafür ein Tag angelegt werden, der als Schablone dient. Dann kann mit Hilfe einer Laufzahl eine Vielzahl von Kopien mit geänderten Eigenschaften erzeugt werden.
- Powertool bietet die Möglichkeit, die notwendigen Items automatisch zu generieren.

Alle diese Methoden setzen viel Erfahrung und eine gute Kenntnis der einzelnen Softwarekomponenten des installierten Systems voraus.

2.2.1 Probleme

Das Importieren und Exportieren der einzelnen Dateien ist, wenn es sich um viele Änderungen handelt, eine sehr einfache und effektive Möglichkeit. Auch beim Erstellen von neuen Projekten hat sich das Erzeugen der notwendigen CSV-Dateien, mit Hilfe des OTAS_Dialog_TOOLs, bewährt. Das Problematische dieser Methode ist, dass das Exportieren und das anschließende Importieren eine geraume Zeit in Anspruch nimmt. Handelt es sich nur um kleinere Änderungen, so ist dies nicht besonders effizient. Zu-

dem können Eingabefehler nur mit erheblichem Aufwand korrigiert werden, da hier zumeist mit der Funktion "suche und ersetzte" gearbeitet wird.

Das Erstellen einzelner beziehungsweise das Generieren von mehreren Tags mit Hilfe von einer Laufzahl ist die variabelste Methode zur Generierung von Tags. Jedoch ist sie sehr zeitaufwendig und daher nur empfehlenswert, wenn es sich nur um einige wenige Tags handelt.

Automatisch generierte Items im OPC-Client bekommen kryptische Namen, die beim späteren Ändern erhebliche Probleme verursachen. Es besteht zwar die Möglichkeit, die Items einfach im OPC-Client zu belassen, doch verursachen diese einen erhöhten Datendurchsatz, was das Gesamtsystem langsamer macht.

Bei unzureichendem Wissen über die vor Ort installierte Anlage oder über den installierten Standard, kann es bei all diesen Methoden zu erheblichen Problemen kommen, denn der Aufbau des Systems ändert sich von Projekt zu Projekt. Das ist zum einen abhängig von den Wünschen des Kunden, zum anderen vom Technologiestand, der zum Erstellen der Anlage prävalent war.

2.2.2 Problemlösung

Um dem Projektierer mit einem Mindestmaß an Informationen eine an die jeweilige Anlage angepasste standardisierte Generierung der Tags zu ermöglichen, ist es nötig, die Erzeugung der Tags zu automatisieren. Hierbei ist es wichtig, dass der Benutzer möglichst intuitiv die Verknüpfungen bis zur Feldebene erstellt.

Eine Lösung wäre es, die Tags schon bei der ersten Projektierung zu generieren und sie dann zu deaktivieren, um eine unnötige Kopplung zu vermeiden, wenn später neue Komponenten der Anlage zugefügt werden sollen. Dann bräuchte der Benutzer nur noch die nötigen Tags zu aktivieren. Dafür müsste er zum einen wissen, welche Tags genau nötig sind für ein bestimmtes Bauteil, zum anderen, ob die Reserve-Tags auf dem aktuellen Stand sind. Hinzu kommt, das diese "Reserve Tags" unnötig viel Speicherplatz brauchen und nicht vollständig deaktiviert werden können, so dass sie das System langsamer machen.

Eine andere Lösung ist die automatische Erzeugung der Tags über Excel, wie mit dem OTAS_Dialog_TOOL, da iFIX™ eine API bereitstellt, mit der man auf alle internen Parameter zugreifen kann. Alle nötigen Parameter kann man direkt in der Excel-Datei hinterlegen. Die Parametrierung wäre entsprechend einfach und die jeweilige Anlage von jedem Rechner aus, der über ein Microsoft Office Paket verfügt, dynamisch anpassbar.

Dieser Vorteil ist aber auch gleichzeitig problematisch, denn auf dem iFIX™-Rechner muss auch Excel installiert sein, da man, um den OPC-Client zu parametrieren, die Software lokal ausführen muss. Ein weiteres Problem ist, dass so zwar alle nötigen Tags korrekt angelegt werden, die Verknüpfung innerhalb des Fließbildes jedoch immer noch manuell erzeugt werden muss.

Darum liegt es nahe, eine Software direkt in iFIX™ zu integrieren, damit man somit unabhängig von Microsoft Office ist und der Projektierer direkt beim Zeichnen der Bilder die Datenpunkte erzeugen kann.

Um in iFIX™ eine Anwendung zu integrieren, bietet sich Visual Basic for Application (VBA) an, da diese Programmiersprache bereits ein Bestandteil von iFIX™ ist und somit auf alle Funktionalitäten zugreifen kann. Zudem sind schon einige Prozeduren zur Konfiguration von den Templates vorhanden.

2.2.3 Aufgabenstellung

Nachdem die Programmierung auf VBA in der Umgebung von iFIX™ festgelegt wurde, muss noch bestimmt werden, was die Software alles können soll.

In den folgenden Stichpunkten wird aufgeführt, welche Eigenschaften und Attribute die Software haben sollte.

- intuitive Bedienung
- leicht in die bestehenden Anlagen zu implementieren
- starten der Software über die bestehenden Templates für die Standardelemente
- möglichst unabhängig von anderer Software
- Erstellung aller nötigen Tags für alle Standardelemente in der Prozessdatenbank
- Erstellung aller nötigen Items im OPC-Client
- Überprüfung von Fehleingaben des Benutzers
- Komptabilität zum SIX-Treiber
- Unterstützung von Siemens S7 und S5 Steuerungen
- bei der Erzeugung der VMs wird der Störcode mit in die Stepptextdatei eingetragen
- bei der Erzeugung von Reglern soll eine Taggroup für den Regler generiert werden und eine Anpassung in der jeweiligen Reglerliste erfolgen.
- bei VM und AI zusätzliche Digital Tags frei konfigurierbar. Diese Tags sollen zum erzeugen der Störmeldungen dienen.

3 Beschreibung der Software

3.1 Installationsanweisungen

Um die Software nutzen zu können, muss sie erst installiert werden. Der gesamte Quellcode befindet sich in der User.fxg, ein Projekt das in VBA eingebunden werden kann. Bei iFIX™ muss die Datei nur in das "Pic" Verzeichnis kopiert werden.

Die Software benötigt dynamische Bibliotheken (DLL-Dateien), damit alle benötigten Objekte initialisiert werden können.

3.1.1 FixEdaEnum.dll

Diese Datei ist im Allgemeinen nicht bei einer Installation von iFIX™ vorhanden. Sie ist Bestandteil des "iFIX™ Integration Toolkit". EDA ist eine Bibliothek von Unterprogrammen, die einen einfachen Zugriff auf die iFix™-Daten erlauben. Es werden gewisse Datentypen unterstützt, die auf diversen Rechnern liegen können. EDA übernimmt dabei das Speichermanagement, die Fehlererkennung und -behandlung sowie die Vernetzung. Diese Datei sollte mit in das Dynamics-Verzeichnis kopiert werden.

3.1.2 TagGroupDefinitionDll.dll

Diese Bibliothek sollte im Dynamics-Verzeichnis liegen. Mit Hilfe der enthaltenen Unterprogramme werden die Taggroups für die Regler erstellt und Regler-Listen bearbeitet.

3.1.3 OPCdrv.tlb

Mit Hilfe diese Typenbibliothek kann man auf den OPC-Client zurückgreifen. Diese Datei muss, auch wenn nicht über OPC gekoppelt wird, eingebunden werden, da VBA das Objekt trotzdem braucht.

3.1.4 Database_Item_config.ini

Diese Konfigurationsdatei dient zur Parametrierung des Projektes (siehe Seite 16). Hier werden zum einen Informationen abgelegt, welche die Automatisierungsgeräte betreffen und zum anderen alle Parameter, die zur Erzeugung der Tags notwendig sind. Diese Datei muss im "PDB" Verzeichnis abgelegt werden.

3.1.5 Einrichten der Software

Nachdem alle Dateien an die richtige Stelle kopiert wurden (siehe Seite 20), ist es nun wichtig zu überprüfen, ob die Bibliotheken richtig eingebunden wurden. Im Regelfall findet VBA die Bibliotheken automatisch. Bevor man jedoch erstmals das Script ausführt, sollte man überprüfen, ob sie alle richtig eingebunden wurden. Dies kann man testen, indem man den Microsoft Visual Basic Editor startet. In der Auswahlleiste unter Tools/References finden sich alle eingebunden Bibliotheksnamen (Abbildung 11). Wenn man auf einen Eintrag klickt, bekommt man für diese Bibliothek den dazugehörigen Pfad mit Dateinamen angezeigt. Falls eine der eingerahmten Bibliotheken fehlt, kann sie über die "Browse"-Taste nachinstalliert werden.

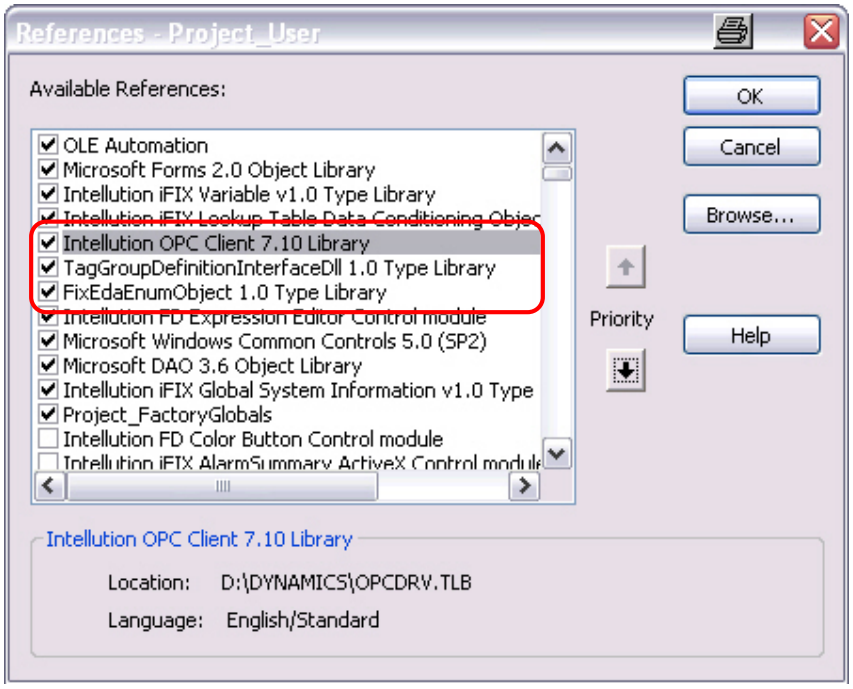


Abbildung 11: References - Project_User

Nach dem Einfügen der User.fxg sollte der Strukturbaum wie in Abbildung 12 aussehen.

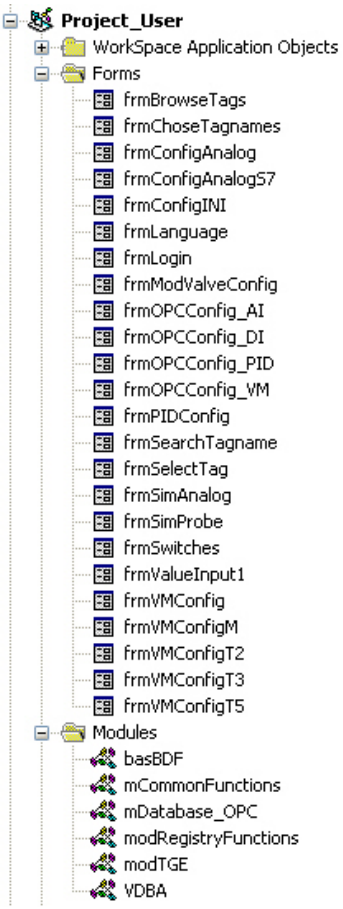


Abbildung 12: Project_User Strukturbaum

3.2 Funktionsbeschreibung

Um eines der Standardelement in einem Fließbild darzustellen zu können, wird ein "Tag" in der Prozessdatenbank benötig. Mit dem Database Wizard soll es erleichtert werden einen dieser Tags anzulegen. Dieses Tool legt nicht nur die entsprechenden Tags in der Datenbank an sondern auch im OPC-Client.

Um die entsprechenden Tags zu erstellen, sind einige Eingaben nötig. Diese lehnen sich stark an die Standard Bausteine, beziehungsweise an die Hardwarekonfigurationen von Tuchenhagen Brewery Systems GmbH an.

Um den Database Wizard zu starten, muss erst ein Template vorhanden sein. Dies kann durch Kopieren aus einem anderen oder aus dem gleichen Bild geschehen. Man hat auch die Möglichkeit, durch "Drag and Drop" aus den "Dynamo Sets" ein Template zu implementieren. Danach kann man durch Doppelklick auf das Template die Konfigurationsoberfläche öffnen. Dort findet sich dann das Icon zum Starten des Database Wizards. Die Eingaben, die in dieser Oberfläche gemacht werden können, dienen nur der Konfigurierung der grafischen Objekte im Fließbild. In Abbildung 13 ist die Konfigurationsoberfläche für einen digitalen Sensor dargestellt. Je Typ des Sensors, wird sich das Aussehen im Fließbild des Sensors ändern.

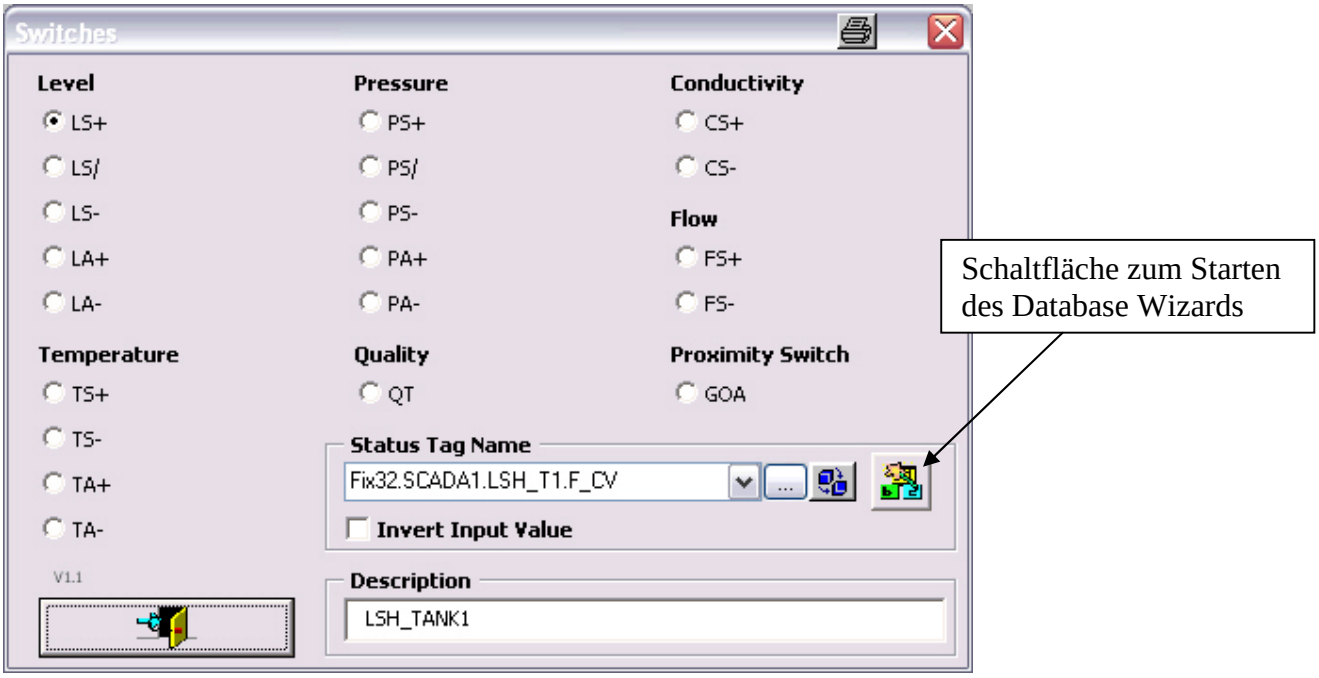


Abbildung 13: Konfigurationsoberfläche Digitale Inputs

In Abbildung 14 ist die Konfigurationsoberfläche eines Standardelements dargestellt. Die meisten der Eingabeboxen, Symbole und Knöpfe (Button) finden sich auf allen Oberflächen wieder.

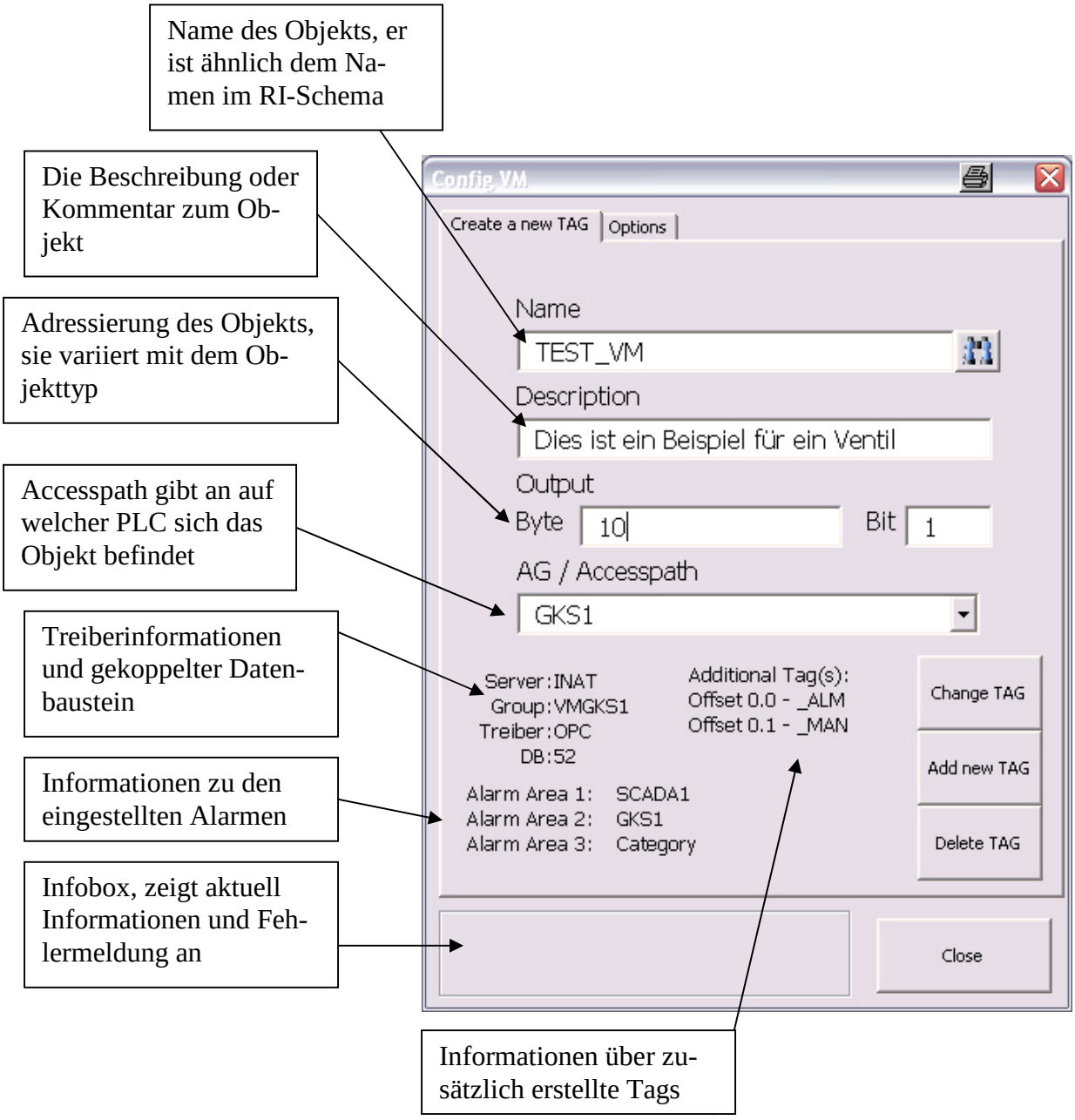


Abbildung 14: Beispiel für Bedienoberfläche

3.2.1 Name

Der Name wird aus dem RI-Schema übernommen (Abbildung 15). Hier gibt es einige Konventionen:

- Eindeutiger Name im ganzen Projekt
- Nur Großbuchstaben und Zahlen
- Mindestens ein Buchstabe im Namen
- Maximal 30 Zeichen
- Keine Sonderzeichen außer dem Tiefstrich

Unter diesem Namen ist dann das Objekt auch im OPC-Client zu finden.

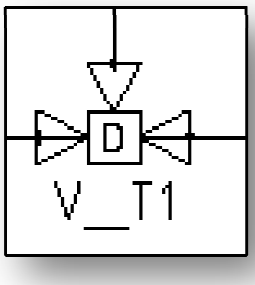


Abbildung 15: Ventil in einem RI Schema

3.2.2 Beschreibung (Description)

Die Beschreibung wird meist als zusätzliche Information angezeigt und sollte eine kurze Beschreibung von dem Bauteil liefern (Abbildung 16). Sie darf nicht mehr als 40 Zeichen enthalten.

Abbildung 16: Control Tip Text im Fließbild

Adressierung

Die Adressierungsart ist für jedes Standardelement anders und hängt mit dem Aufbau des Datenbausteins zusammen.

3.2.3 Analoger Input /Regler

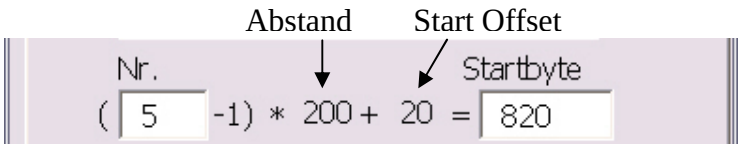


Abbildung 17: Adresszeile Analoge Inputs / Regler

Bei diesen beiden Standardelementen ist die Adressierung identisch, da der Aufbau des Datenbausteins ähnlich ist. Der Anwender hat die Möglichkeit entweder die Nummer des jeweiligen Objekts anzugeben oder er kann das Startbyte direkt eingeben, woraus dann die Nummer berechnet wird.

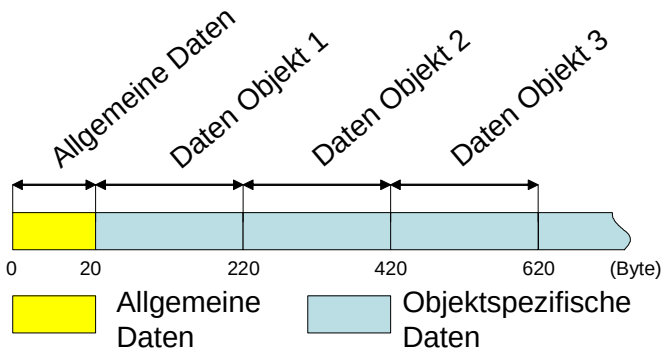


Abbildung 18 : Aufbau des Datenbaustein für AI und PID

Am Anfang des Bausteins befinden sich allgemeine Informationen zum Regler-Baustein, wie zum Beispiel die Anzahl der Regler, die im Datenbaustein angelegt sind und wie viele davon gekoppelt werden (Abbildung 18). Außerdem gibt es auch einige Datenwörter zur freien Benutzung.

In den objektspezifischen Daten sind alle Variablen gespeichert, die direkt einem Element zugeordnet werden können. Bei einem analogen Wert sind das zum Beispiel der Analogwert selbst sowie einige Statusvariablen.

3.2.4 Ventile und Motoren

Für jedes einzelne VM-Objekt wird ein Statuswort (16 Bit) benötigt, auf dem der Zustand der Ventile und Motoren dargestellt und verändert werden kann.

Der Aufbau des Datenbausteins ist ähnlich denen der Regler und der analogen Inputs, nur dass sich hier die Startadresse anders berechnet. Da genau ein Wort pro Ausgang und kein Start-Offset vorgesehen ist, kann die Anzahl der Bits gleich der Wort-Adresse gesetzt werden, wobei darauf zu achten ist, dass die S7 byteorientiert und die S5 wortorientiert adressiert.

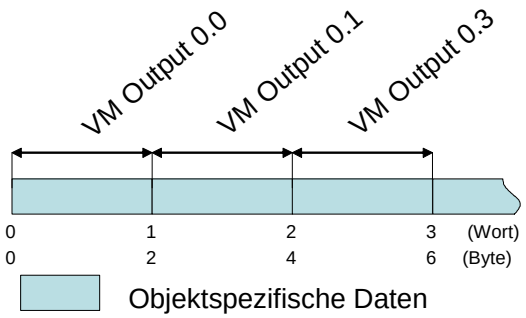


Abbildung 19: Aufbau VM Datenbaustein

- S5: Statusadresse = Ausgangsbyte*8 + Ausgangsbit

  (wortorientierte Verarbeitung)

- S7: Statusadresse = (Ausgangsbyte*8 + Ausgangsbit) *2

  (byteorientierte Verarbeitung)

Aufgrund dieser Rechnung wird hier auch nur das Ausgangsbyte und -bit angegeben (Abbildung 20).

Abbildung 20: Adresszeile Ventile und Motoren

Da aber einige Anwender wegen Sonderdatenbausteinen lieber gleich die Statusadresse angeben möchten, wurde ein verstecktes Feld hinzugefügt, das durch Doppelklick auf den Schriftzug "Output" sichtbar gemacht werden kann, wobei nur das erste Kästchen beachtet werden sollte (Abbildung 21). In diesem Kästchen steht die reale Adresse des Objekts im Datenbaustein.

Abbildung 21: Zusätzlich Adresszeile Ventile und Motoren

Digitale Inputs

Abbildung 23: Adresszeile Digitale Inputs

Der Datenbaustein für die digitalen Inputs, stellt eins zu eins die digitale Baugruppe der SPS dar. Das heißt, der Wert der an E4.4 anliegt, kann auch in Byte 4.4 im Datenbaustein wiedergefunden werden. Bei einer S5, die Wort orientiert arbeitet, wäre es Wort 4.4. Da der Eingang auch byte-orientiert ist, verschenkt man die Hälfte des Speicherplatzes. Da man aber bei einer großen Anlage von etwa 400 einzeln aufgelegten Sensoren redet, ist der verschenkte Platz relativ gering.

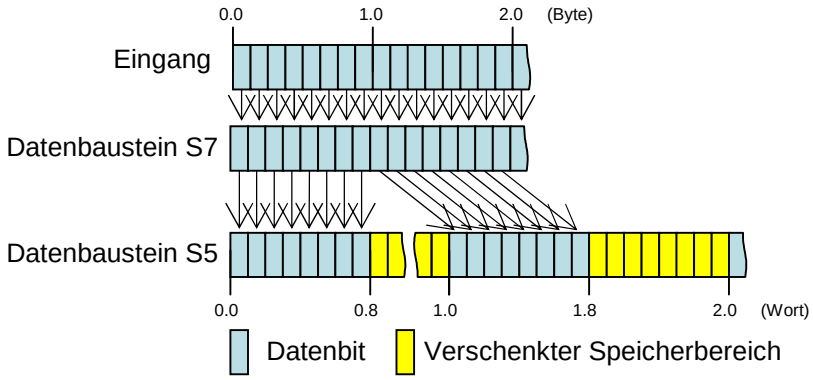


Abbildung 22: Aufbau Datenbaustein Digitaler Input

3.2.5 Steuerung (AG/Accesspath)

In dieser Auswahlbox sind alle installierten Steuerungen enthalten. Beim Ändern des Namens werden alle notwendigen Informationen aus der INI-Datei geladen, die der Steuerung zugeordnet werden. Eine Steuerung kann man mit Hilfe des INIConfig Fensters nachinstallieren.

3.2.6 Treiberinformationen

Hier finden sich alle Informationen, die sich auf den installierten Treiber beziehen. Wenn es sich um einen SIX-Treiber handelt, werden die oberen beiden Zeilen ausgeblendet. Wenn es sich um einen OPC-Treiber handelt, steht hier der Pfad unter dem sich das Item befindet. Die Werte lassen sich in der Registrierkarte "Options" ändern. (weitere Informationen auf Seite 34. und 33.)

3.2.7 DB

DB gibt an, in welchem Datenbaustein sich die Daten für den entsprechenden Tag befinden. Sollte die Nummer von der Standardnummer abweichen, wird die Zahl rot angezeigt. Regler lassen sich nicht außerhalb des Standarddatenbausteins erstellen.

Bei analogen Inputs werden die zusätzlichen Tags nicht erzeugt, während sich Ventile, Motoren und digitale Inputs ohne Einschränkungen in anderen Datenbausteinen erstellen lassen (siehe auch Seite 33).

3.2.8 Alarme

Einige der Tags lösen Alarme aus, die in unterschiedliche Bereiche gegliedert sind. Diese Bereiche sind zum Beispiel der Knoten (Node), den der Rechner darstellt, die Steuerung, auf der die Daten für die Tags liegen oder welchem Typ der Tag zugeordnet werden kann. "Category" ist ein spezieller Begriff und steht hier für unterschiedliche Bereiche. In dem oberen Beispiel eines VM-Standardelements werden zwei zusätzliche Tags erstellt. Zum einen ein "Alarm-Tag", zum anderen der "Manuel-Tag". Wenn sich der Zustand eines Tags ändert, wird ein Alarm ausgelöst. Der Knotenname und der Steuerungsname sind bei beiden gleich, jedoch gehören beide unterschiedlichen Kategorien an. Die unterschiedlichen Alarmbereichsnamen bekommt man, wenn man mit dem Mauszeiger über den Schriftzug geht.

Abbildung 24: Anzeige von mehren Alarmen

3.2.9 Infobox

In der Infobox stehen alle Informationen zu den aktuellen Vorgängen. Hier werden auch Fehler angezeigt. Der Inhalt der Infobox wird durch einen Doppelklick auf das Feld gelöscht.

3.2.10 Additional Tag(s)

Alle zusätzlich erstellten Tags werden hier angezeigt. Wobei "Offset" der Abstand zum Startbyte ist. Neben dem Bindestrich stehen die Suffixe, die beim Erstellen an den Tagnamen angehängt werden. Sie fangen meist mit einem Tiefstrich an um den Suffix vom Namen zu trennen.

3.2.11 Zusätzliche Reglerparameter

Bei einem Regler sind einige Parameter mehr nötig:

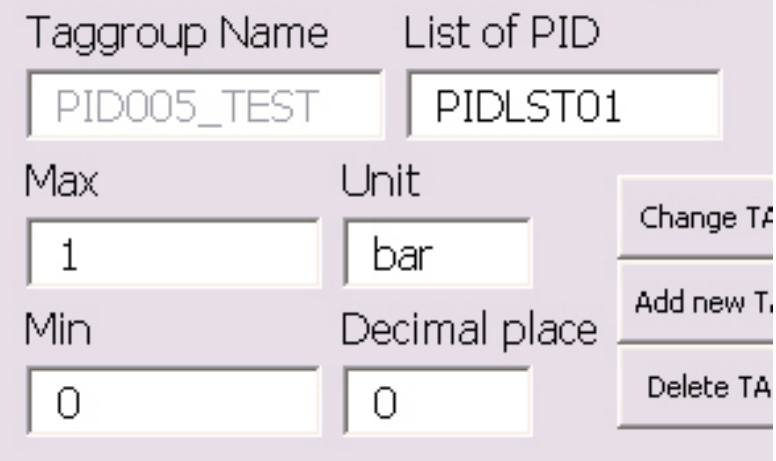


Abbildung 25: Zusätzliche Parameter für PID

Tagg- roup Name

Der Name der Taggroup kann nicht verändert werden, da er wie der Regler selbst heißen muss.

List of PID

Hinter der Liste der Regler verbirgt sich eine Taggroup, in der eine Anzahl von Reglern aufgelistet ist.

Max und Min

Unter Max und Min befinden sich die Grenzen des Regler-Inputs.

Unit

Unter Unit findet man die Einheit des Soll/Ist-Wertes.

Dezimalstelle

Die Dezimalstelle zeigt an, mit wie viel Nachkommastellen die aktuellen Werte des Reglers angezeigt werden sollen. Insgesamt können nicht mehr als sieben Stellen angezeigt werden.

3.2.12 "Change Tag-Taste"

Mit Hilfe dieser Schaltfläche können alle Parameter des Objektes geändert werden, abgesehen vom Tagnamen selbst. Wenn dieser geändert werden soll, muss das Objekt erst gelöscht werden, um dann wieder mit dem neuen Namen erzeugt zu werden.

3.2.13 "Add new Tag-Taste"

Wenn dieser Botton aktiviert wird, werden alle notwendigen Tags erstellt. Bei Ventilen und Motoren wird zusätzlich ein Eintrag in die Stepptext-Datei generiert. Bei Reglern wird eine Taggroup erzeugt und die entsprechende Reglerliste angepasst.

3.2.14 "Delete Tag-Taste"

Durch diese Schaltfläche wird ein Dialog aufgerufen, in dem der Benutzer das Löschen des Tags bestätigen soll. Bei den digitalen Inputs ist dies nur eine einfache Ja/Nein Anfrage, da es sich hier nur um einen einzelnen Tag handelt. Bei allen anderen wird ein spezielles Fenster geöffnet, in dem alle Tags angezeigt werden, die den gleichen Anfang haben. Dort hat der Benutzer dann die Möglichkeit, nur die Tags auszusuchen, die er wirklich löschen möchte.

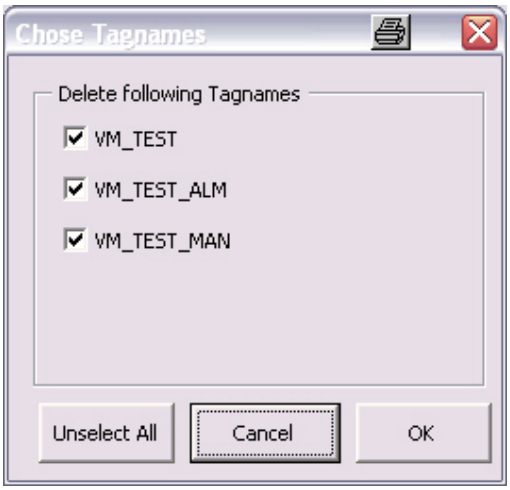


Abbildung 26: Auswahl der zu löschenden Tags

3.2.15 "Close-Taste"

Beim Betätigen dieses Schaltfläche wird das Fenster geschlossen und alle nötigen Eingaben in das darunter liegende übernommen. Sollte es den Tag noch nicht geben, wird eine Anfrage an den Benutzer gestartet, ob der Tag erzeugt werden soll (Abbildung 27).

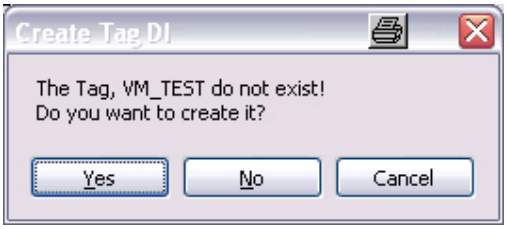


Abbildung 27: Abfragefenster wenn Tag nicht bekannt

3.2.16 "Search a Tagname-Taste"

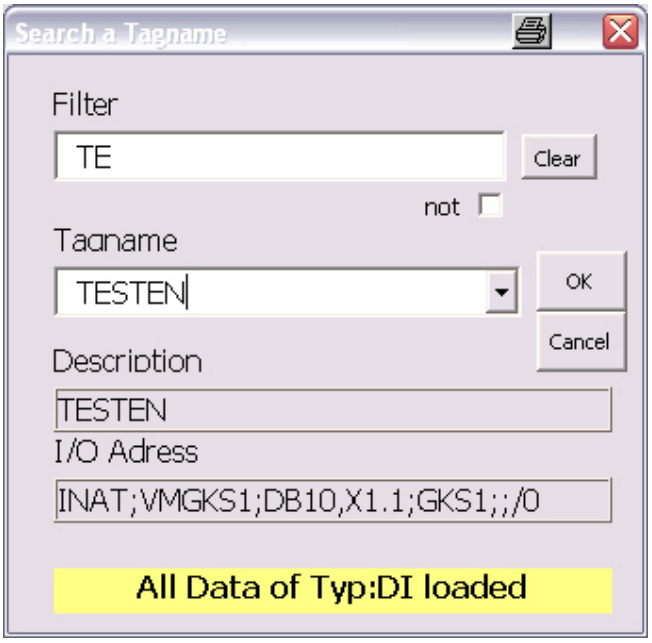


Abbildung 28: Search a Tag Fenster

Wenn es schon einen Tag gibt, kann er hier gesucht werden. Dieses Werkzeug durchsucht die Datenbank nach Tags, die im Standarddatenbaustein liegen. Wenn ein Tag ausgesucht wurde, kann er in die Userform übernommen werden.

Die Filter-Funktion durchsucht die Liste möglicher Tagnamen nach einer Zeichenkette und verringert so die Größe der Auswahlliste, die man sich mit Hilfe des Pfeils (im Feld Tagname) anzeigen lassen kann.

Die anderen beiden Felder enthalten zusätzliche Informationen für den Benutzer. Der gelbe Balken zeigt den Fortschritt beim Suchen an.

3.2.17 "Options" Register

In den Optionen kann man diverse Randparameter einstellen, die den "normalen" Benutzer nicht interessieren (Abbildung 29). Da dieses Tool aber möglichst variabel gestaltet werden sollte, kann man hier Einstellung tätigen, die von der Norm abweichen.

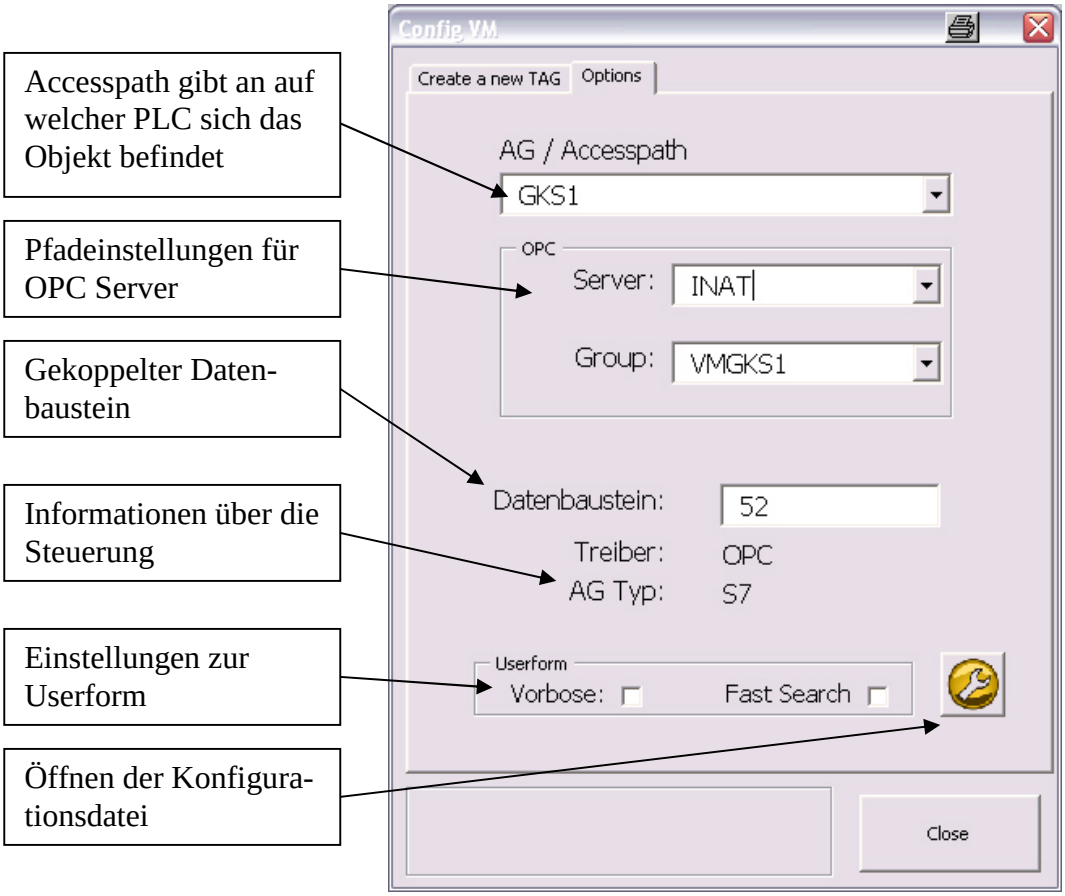


Abbildung 29: Optionsregister

3.2.18 Steuerung (AG/Accesspath)

In dieser Auswahlbox sind alle installierten Steuerungen enthalten. Beim Ändern des Namens werden alle notwendigen Informationen, die der Steuerung zugeordnet werden, aus der INI-Datei geladen. Eine Steuerung kann man mit Hilfe des INIConfig-Fensters nachinstallieren.

3.2.19 OPC-Rahmen

Wenn ein OPC-Server installiert ist, wird der Pfad im OPC-Client angezeigt. Sollte ein SIX-Treiber installiert sein, wird der gesamte Inhalt des Rahmens unsichtbar gemacht.

Wenn unter "Server" ein unbekannter Name eingeben wird, steht im Kästchen "Group" ein "Nothing", zusätzlich gibt es eine Fehlermeldung in der Infobox.

3.2.20 Datenbaustein

Der Datenbaustein gibt an, in welchem Datenbaustein sich die Daten für den entsprechenden Tag befinden (weitere Informationen Seite 28).

3.2.21 Steuerungsinformationen

Hier steht, wie zum Automatisierungsgerät gekoppelt wird und um welche Art von Steuerung es sich handelt. Diese Einstellungen lassen sich nur in der Konfigurationsdatei ändern (zusätzlich Informationen auf Seite 27 und 34).

3.2.22 Userform Rahmen

In diesem Bereich lässt sich das Verhalten der Fenster einstellen.

Wenn Vorbose aktiviert ist, werden mehr Informationen auf das Infofenster ausgegeben. Wenn es zu Fehlern kommt, kann man diese dadurch besser lokalisieren. Standardmäßig ist es deaktiviert, da die meisten Benutzer an so detailierten Informationen nicht interessiert sind.

Bei Aktivierung von "Fast Search" wird die Suche von Tags (Seite 31) beschleunigt. In diesem Fall werden alle Tags eines Typs ausgeben und es wird nicht mehr nach bestimmt Tags in einem DB gesucht.

3.2.23 "Config INI-Taste"

Die "Config INI-Taste" öffnet die aktuelle INI-Datei in einem speziellen Fenster. Die Datei kann auch mit Texteditor oder mit Excel bearbeitet werden.

3.3 Konfiguration der INI-Datei

Die Konfigurationsdatei muss sich im PDB-Verzeichnis von iFIX™ unter dem Namen "Database_Item_config.ini" befinden. In der Datei können alle Einstellungen für den Database Wizard vorgenommen werden.

Die Konfigurationsdatei sollte nur vom Fachpersonal geändert werden, da hier keine syntaktischen beziehungsweise semantischen Abfragen der Eingabe getätigt werden.

3.3.1 Common-Einstellungen

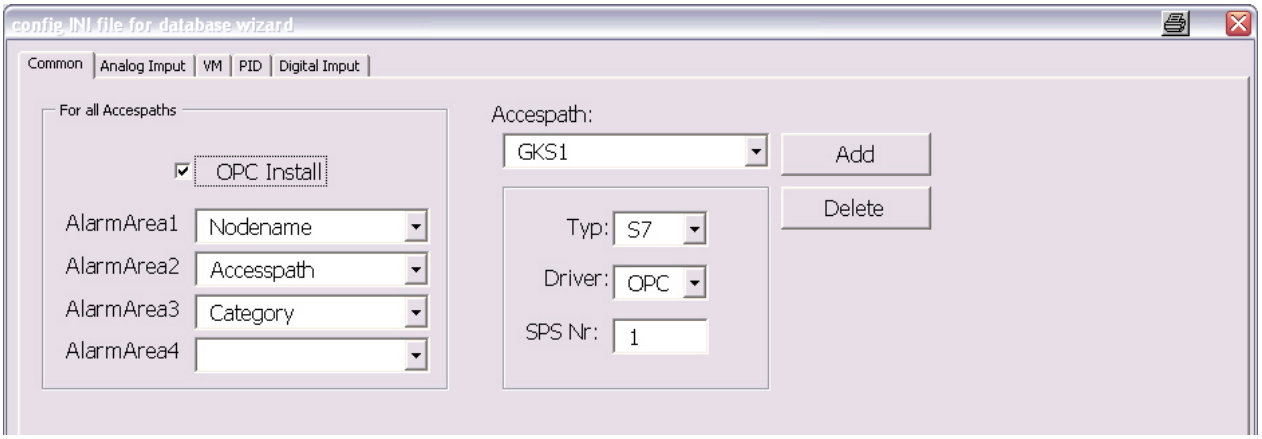


Abbildung 30: Register Common

In dem Reiter "Common" kann man allgemeine Einstellungen vornehmen und Steuerungen neu erzeugen beziehungsweise löschen (Abbildung 30).

3.3.2 OPC Install

"OPC Install" muss aktiviert sein wenn über den OPC-Client kommuniziert wird.

3.3.3 AlarmArea's

Alle Tags, die AlarmAreas enthalten werden mit diesen Einstellungen konfiguriert (Siehe Seite 28). Die in diesem Beispiel eingetragenen Werte sind Platzhalter und werden dynamisch angepasst.

3.3.4 Accesspath

Der Accesspath beinhaltet die allgemeinen Einstellungen für die Steuerungen. Die SPS Nr. ist wichtig für die Erstellung des Störcodes. Bei der Erzeugung einer neuen Steuerung wird eine Kopie der aktuellen Steuerung erzeugt (Siehe Seite 27 und 33).

3.3.5 Register VM und Analog Input

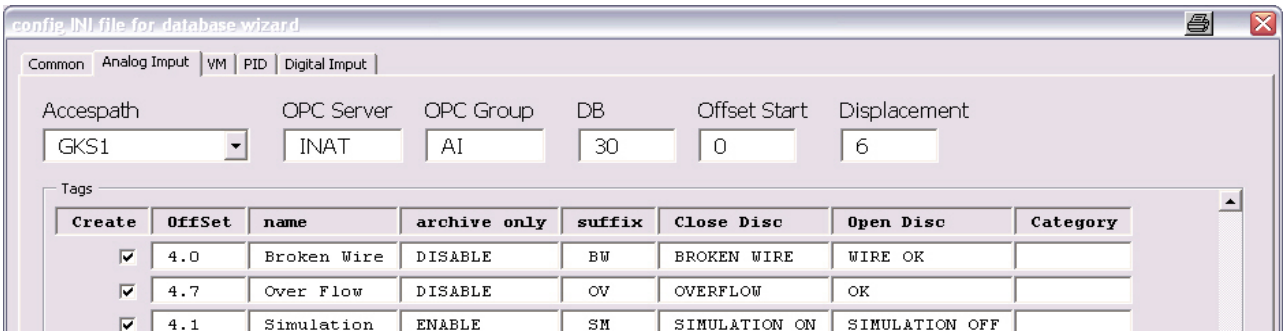


Abbildung 31: Register für analoge Inputs (Konfigurationsoberfläche)

Die beiden Registerkarten (Abbildung 31) sind dem Aufbau nach gleich, abgesehen von dem Fehlen des "Displacement" und "Offset Start" bei den VMs (siehe Seite 26).

Accesspath

Bei Accesspath kann man zwischen den Steuerungen umschalten.

OPC Server

Unter OPC "Server" kann man eingeben, unter welchem Server die Items im OPC-Client angelegt werden sollen.

OPC Group

Die "OPC Group" gibt an, unter welcher Gruppe die Items im OPC-Client angelegt werden sollen.

DB

DB gibt den Datenbaustein an, in welchem die Objekte abgelegt sind.

Offset Start

"Offset Start“ gibt an, wie groß der Platz am Anfang des Datenbausteins für extra Daten ist bei einer S5-Steuerung in Word und bei einer S7-Steuerung in Byte. Dies gilt auch für das Displacement.

Displacement

"Displacement" gibt an, wie viel Speicherplatz ein einzelnes Standardelement braucht.

Tags

In dieser Tabelle sind alle Informationen für zusätzlich angelegte Tags enthalten. Diese Tags werden erzeugt, wenn der Haken unter "Create" gesetzt ist. Der Name ist die Bezeichnung des Tags (S.24). Der "Offset" beschreibt, wie groß der Abstand vom Startbyte

ist. Diese beiden Eigenschaften sind von der Steuerung abhängig. Archive Only muss aktiviert werden, wenn die Tags auch im Meldeprotokoll angezeigt werden sollen. Die Suffixe bestimmen den zukünftigen Namen der Tags und sollten mit einem Unterstrich anfangen (S.28). Unter "Open" und "Close Disc" wird angegeben, welche Informationen im Alarmprotokoll stehen, wenn einer dieser Tags seinen Wert ändert. Bei "Category" kann man zusätzlich eine spezielle AlarmArea für diesen Tag angeben.

3.3.6 Register PID

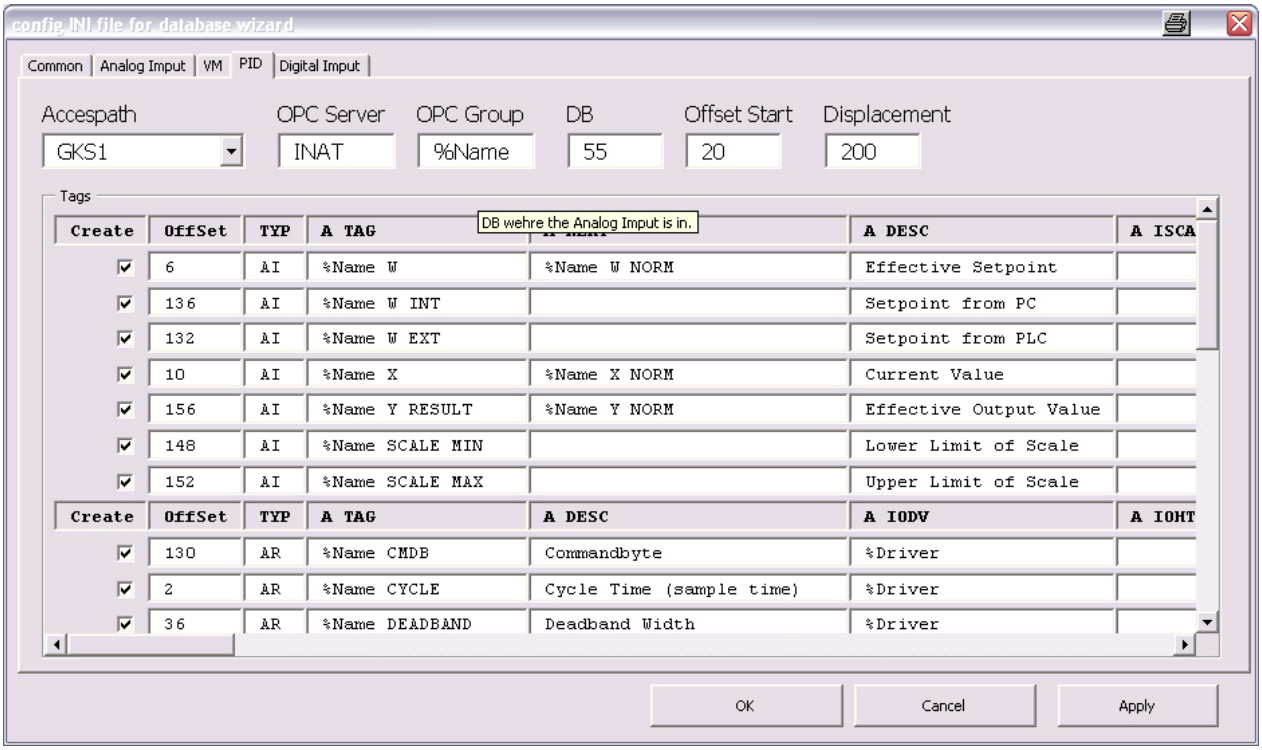


Abbildung 32: Register für Regler (Konfigurationsoberfläche)

In Abbildung 32 finden sich ähnliche Eingabeflächen wie im Register "Analoge Inputs" (Abbildung 31). Der Unterschied ist hier die Tabelle für die Tags. Da es sich um 31 Tags unterschiedlichsten Typs handelt, wurden alle Tags einzeln aufgeschlüsselt. Die ersten beiden Spalten haben die gleiche Funktion und Bedeutung wie die im Register "Analog Input". Alle anderen Spalten entsprechen den Eigenschaften der einzelnen Typen in der Prozessdatenbank. Um diese Tags variabel gestalten zu können, wurde mit Wildcards gearbeitet, die immer mit einem Prozentzeichen anfangen. Alle diese Werte sind steuerungsunabhängig und müssen demzufolge nur einmal geändert werden.

Die folgenden Wildcards sind möglich:

% Name	Name des Reglers
%Max, %Min	Maximalgrenzen des Reglers (siehe Seite 29)
%LinLow, %LinHigh	stellen eine Zahl aus sieben neunen und einem Vorzeichen dar. Die Anzahl der Nachkommastellen richtet sich nach der Kommastelle (Abbildung 25).
%Unit	Einheit des zu regelnden Mediums.(Abbildung 25)
%Description	die Beschreibung des Reglers (siehe Seite 24)
%Driver	entspricht der Art der Kopplung, OPC-Client oder SIX-Treiber (siehe Seite 27).
%AlarmArea1 (bis 4)	hier werden die Werte angenommen, die in der Userform für die Regler angezeigt werden.
%OffSet	hier wird der Wert eingetragen der in der Offsetspalte steht.
%IOAD	diese Wildcard sollte nur in die Spalte A_IOAD eingetragen werden, denn dann wird dieses Feld anders behandelt und die notwendige Input/Output Adresse erzeugt.

Wenn ein neuer Tag hinzugefügt werden soll, muss dies direkt über die INI-Datei und einen einfachen Texteditor geschehen. Dort kann dann problemlos eine Zeile kopiert werden und der neue Tag wird in der Tabelle mit angezeigt.

3.3.7 Register "Digital Input"

In dem Register "Digital Input" finden sich im Grunde die gleichen Eingabeboxen wie auf den anderen Registern. Der Unterschied ist nur, dass hier zusätzlich noch eine Eingabebox "Category" vorhanden ist. Der hier stehende Eintrag ist für die AlarmAreas gedacht und wird entsprechend der Konfiguration auf dem Common Register verarbeitet.

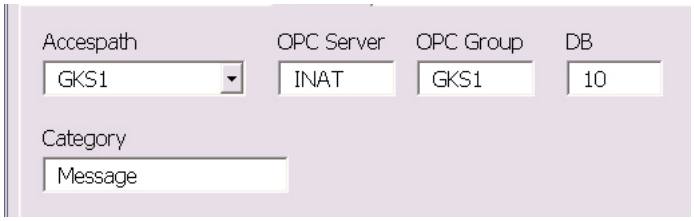


Abbildung 33: Register Digital Input

3.4 Aufbau der Module und Userformen

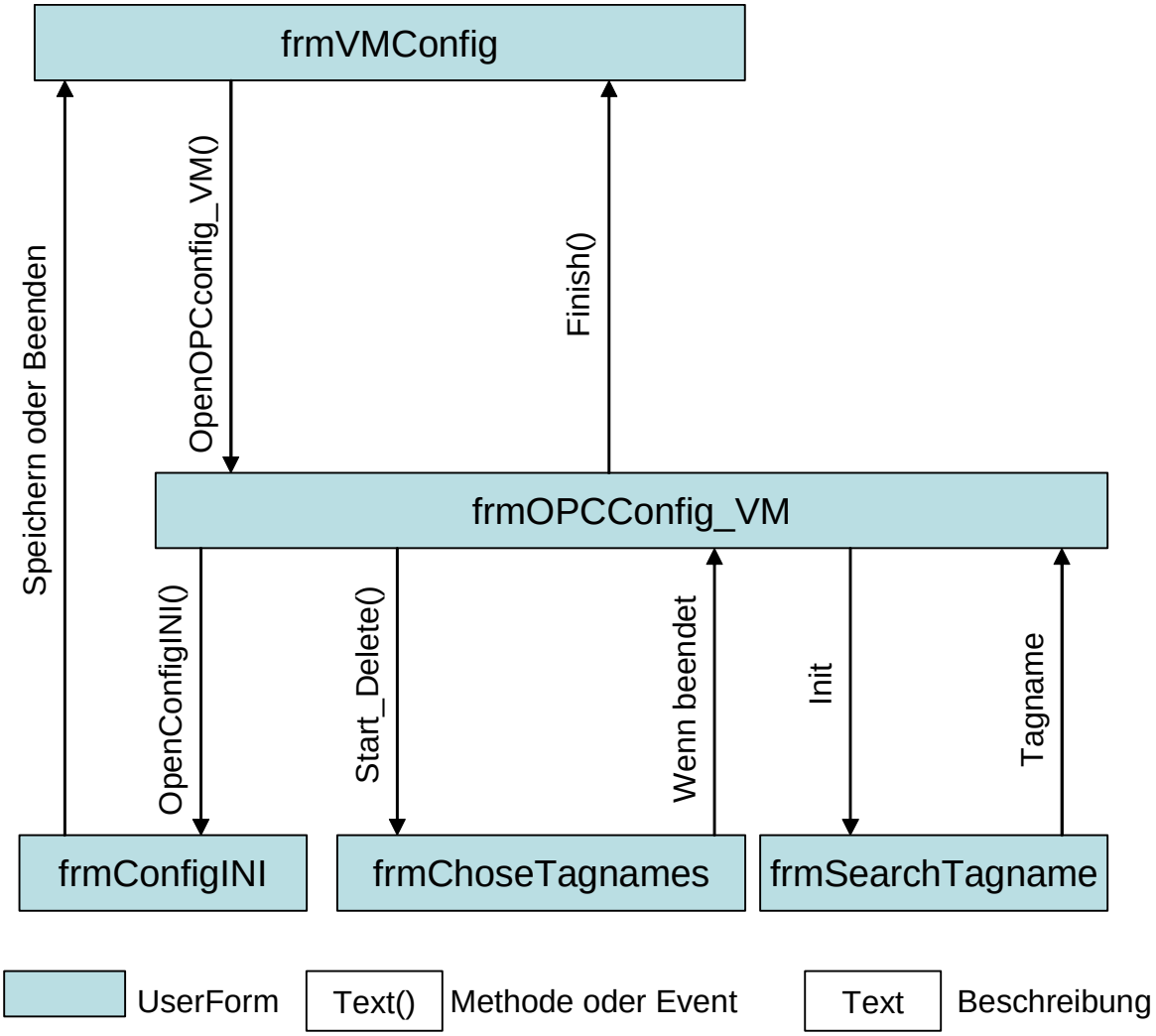


Abbildung 34: Ablaufdiagramm der Userformen

Die Userform frmVMConfig steht hier für die Userformen, die der Konfigurierung der Templates dienen. Diese Oberflächen wurden kaum verändert. Es wurde nur eine Schaltfläche (Button) hinzugefügt, der die entsprechende Methode startet, welche wiederum die entsprechende Userform initialisiert. In diesem Fall ist es die OpenOPCconfig_VM(), die hier das Fenster frmOPConfig_VM initialisiert. Es werden zum Beispiel feste Parameter aus der INI-Datei gelesen und in die Userform eingetragen sowie benötigte Objekte instanziert.

Mit Hilfe der Userform frmOPCConfig_VM kann der Benutzer einen Tag in der Prozessdatenbank ändern, löschen oder neu erzeugen.

Wenn er einen bestehenden Tag verknüpfen oder öffnen will, hat er die Möglichkeit, diesen mit der Userform frmSearchTagname zu suchen. In diesem Fenster werden nur

Tags angezeigt, die dem Standardelement zugeordnet werden können. Diese Userform bietet zusätzlich eine Filterfunktionalität, um den gesuchten Tag schneller zu finden. Wenn der entsprechende Tag gefunden wurde, wird er in die aktuelle Userform eingetragen.

Beim Löschen des Standardelements müssen oft mehrere Tags gelöscht werden. Um dem Benutzer die Möglichkeit zu geben nicht alle Tags zu löschen, kann er die zu löschenden Tags auswählen. Dies geschieht in der Userform frmChoseTagnames. Nach dem Löschen bleiben alle eingegebenen Daten erhalten. Falls der Benutzer einen neuen Tag mit den gleichen Parametern anlegen möchte, so muss er diese nicht komplett neu eingeben, sondern kann den neuen Tag mit geänderten Parametern erzeugen.

Um die INI-Datei bearbeiten zu können, muss man den Umweg über eine der Userformen machen, da dies die unkomplizierteste Möglichkeit ist, die Konfigurationsoberfläche in bestehende Projekte einzubinden. Die Einstellungen über eine extra Schaltfläche innerhalb von iFix™ einzubinden, wäre zwar auch möglich gewesen, jedoch hätte man dafür bei jedem Projekt eine spezielle Anpassung tätigen müssen.

Nachdem ein neues Standardelement erzeugt wurde, speichert VBA intern den Tagnamen. Ruft man das nächstemal ein Element gleichen Typs auf, wird der alte Tagname eingefügt. Existiert der Tagname im Konfigurationsfenster für die Templates schon in der Datenbank, wird dieser geladen.

3.5 Funktionsbeschreibungen der Methoden

user.mDatabase_OPC und *user.VDBA*

In diesen Visual Basic Modulen sind die von dem Database Wizard benötigten Funktionen sowie global verfügbare Variablen und Konstanten definiert. Im Folgenden werden die Inhalte der Module kurz erläutert.

3.6 Modul: mDatabase_OPC

3.6.1 Deklarationen:

Variablen:

PowerTool	Objekt der Schnittstelle des OPC-Clients
Database	Objekt der Schnittstelle der Datenbank
TempOldTagname	Gespeicherter Tagname der aktuellen Userform
Old_Tagname_DI	Gespeicherter Tagname des letzten geöffneten digitalen Inputs
Old_Tagname_PID	Gespeicherter Tagname des zuletzt geöffneten Reglers
Old_Tagname_VM	Gespeicherter Tagname des zuletzt geöffneten Ventils oder Motors
Old_Tagname_AI	Gespeicherter Tagname des zuletzt geöffneten analogen Inputs
IniFilename	Der Pfad der INI Datei
PicPath	Der Pfad in dem die iFIX™-Bilder gespeichert werden
Infocount	Anzahl der Meldungen im Infofenster

Globlae Variablen:

PDBName	Name der Prozessdatenbank
Nodename	Knotenname des Rechners
FastSerach	Aktiviert die Schnellsuche
infoVorbose	Alle Meldungen werden angezeigt
AllItems	String Array aller Tagnamen die dem aktuellen Standardelement zugeordnet werden können

Konstante:

RED	Farbe Rot als Zahl
GREEN	Farbe Grün als Zahl
BLACK	Farbe Schwarz als Zahl
LEGAL_CHARS	String von zugelassenen Zeichen

3.6.2 Funktionen und Prozeduren

CheckFile

- überprüft, ob es eine Datei gibt und gibt diese zusätzlich als Objekt wieder zurück.

CheckFloat

- überprüft, ob ein String eine Fließkommazahl ist.

CheckNumber

- überprüft, ob es sich bei dem String um eine ganze Zahl handelt.

CheckString

- überprüft, ob der String nur aus Zeichen, besteht die in LEGAL_CHARS definiert sind.

CheckTagname

- überprüft, ob ein String in einem eindimensionalen Array ist.

ClearInfobox

- löscht alle Einträge aus der Infoanzeige der aktuellen Userform.

CreateTAG_AI

- erzeugt alle Tags, die für einen analogen Input notwendig sind und gegebenenfalls auch für den OPC-Client.

CreateTAG_DI

- erzeugt alle Tags, die für einen digitalen Input notwendig sind und gegebenenfalls auch für den OPC-Client.

CreateTAG_PID

- erzeugt alle Tags die für einen Regler notwendig sind und gegebenenfalls auch für den OPC-Client. Des Weiteren wird die entsprechende Taggroup erzeugt und die Taggrouplist geändert.

CreateTAG_DI

- erzeugt einen Tag für digitalen Input in der Datenbank und gegebenenfalls auch im OPC-Client. Des Weiteren wird der Störcode in der Steptextdatei erstellt.

Delete_Tag_by_TagName

- löscht anhand des Namens einen Tag aus der Datenbank und gegebenenfalls aus dem OPC-Client.

DeleteOneStringInArray

- löscht anhand des Inhalts ein Element aus einem Array von Strings.

DeleteTAG_PID

- löscht alle ausgesuchten Tags und gegebenenfalls auch die entsprechenden Items im OPC-Client. Des Weiteren wird die entsprechende Taggrouplist geändert.

DeleteTAG_VM

- löscht alle ausgesuchten Tags und gegebenenfalls auch die entsprechenden Items im OPC-Client.

Finish

- speichert die Datenbank und den OPC-Client, wenn dieser initialisiert ist.

FormatFloat

- konvertiert einen String nach einem bestimmten Muster, wenn dieser eine Gleitkommazahl darstellt.

get_CollectionFromTable

- liefert eine Collection zurück, die dem Aussehen der Tabelle, die in der Userform frmConfigINI erzeugt wurde, entspricht.

Get_Control_by_name

- liefert ein Objekt des Typs Control aus einer Controlcollection anhand des Controlnamens

Get_Data_from_INI

- liefert einen einzelnen Wert aus der INI_Datei anhand des Abschnitts und des Key. (siehe S.16)

Get_IOAdressDI

- liefert einen speziell für den digitalen Input generierten Adressstring, der dann in die Datenbank oder in den OPC Server eingefügt werden kann.

Get_IOAdressPID

- liefert einen speziell für die Regler generierten AdressString, der dann in die Datenbank oder in den OPC_Server eingefügt werden kann. Der AdressString basiert auf Daten aus der Userform frmOPCConfig_PID.

Get_Items_by_Pfad_OPC

- liefert ein Array von Kindnamen anhand des übergebenen Pfades aus dem OPC-Client (Funktioniert nur mit Groups und Servern).

Get_Parameter_by_Pfad_OPC

- liefert ein Parameterarray eines Items anhand des Pfads und der Parameternamens.

Get_Parameter_by_Tagname_Database

- liefert ein Parameterarray eines Tags aus der Datenbank anhand des Tagnamens und der angeforderten Parameternamen.

Get_SectionValueArr

- liefert wahlweise ein eindimensionales oder zweidimensionales Array, was eine Section(AG) aus der INI-Datei darstellt(Siehe S.68).

Get_Tagname_from_expStatusTag

- liefert den Tagnamen von einem String, der diesem ähnlich ist.

  (String = "FIX32.PDBName.Tagname.A_VAl")

getTagnames_by_Typ

- liefert ein TagnameArray anhand der Kurzbezeichnungen für die Standardelemente ("VM", "PID", "DI", "AI"). Dabei wird nach den Kriterien gesucht, die in der INI-Datei hinterlegt sind.

Get_ValFromArr2D

- durchsucht ein zweidimensionales Array mit zwei Spalten nach einem Schlüssel, der in der ersten Spalte stehen muss. Der zugehörige Wert kann direkt ausgegeben werden, wenn der Wert eigentlich aus mehreren Werten besteht und diese mit einem Trennzeichen versehen sind. So kann auch dieser Wert separiert werden.

getGroupID

- gibt den Identifikator (Handle) einer Group anhand des Pfades im OPC-Client zurück.

getItemID

- gibt den Identifikator (Handle) eines Items anhand des Pfades im OPC-Client zurück.

GetNodename

- speichert den aktuellen Knotennamen in der globalen Variable "Nodename".

GetPDBName

- speichert den lokalen Datenbanknamen in der globalen Variable "PDBName".

getSectionnames

- liefert alle Abschnittsnamen aus der INI-Datei.

GetServerID

- gibt den Identifikator (Handle) eines Servers anhand des Pfades im OPC-Client zurück.

InfoUpdate

- schreibt einen übergeben Text auf die Infobox der aktuell offenen Userform. Die Methode schreibt nur auf die Userform, wenn es sich um einen Fehler handelt, eine direkte Anweisung oder die globale Variable InfoVorbose gesetzt ist. Fehler werden mit roter Schrift angezeigt.

Init

- diese Methode ist der Konstruktor für dieses Modul. Hier werden alle wichtigen globalen Variablen gesetzt. Diese Methode sollte immer vor der Nutzung anderer Methoden in diesem Modul ausgeführt werden.

Insert_into_Database

- erzeugt einen neuen Tag in der Prozessdatenbank. Übergeben werden müssen der Tagname, der Typ des Tags und die entsprechenden Parameter mit Parameternamen.

InsertStoercode

- erzeugt einen neuen Eintrag in der Stepptextdatei, wenn es den berechneten Störcode noch nicht gibt.

InsertTable

- erzeugt eine Tabelle in einem übergebenen Frame. Diese Methode ist speziell auf die INIDatei abgestimmt.

MultiDArrayTo1DArray

- wandelt ein zweidimensionales Array in ein eindimensionales um. Welche Spalte wiedergegeben werden soll, kann angegeben werden.

OpenConfigINI

- initialisiert und füllt die Userform frmConfigINI.

OpenOPCConfig_AI

- initialisiert und füllt die Userform frmOPCConfig_AI.

OpenOPCConfig_DI

- initialisiert und füllt die Userform frmOPCConfig_DI.

OpenOPCConfig_VM

- Initialisiert und füllt die Userform frmOPCConfig_VM. Dieser Methode muss die öffnende Userform noch mit übergeben werden, da es davon mehrere gibt.

OpenOPCConfig_PID

- Initialisiert und füllt die Userform frmOPCConfig_PID

SaveINI

- speichert die INI-Datei. Diese Methode wird von der Userform frmConfigINI ausgelöst.

Set_Data_to_INI

- schreibt einen einzelnen Wert in die INI Datei.

SplitIOAdress

- wandelt den String, der in der Datenbank unter dem Feld "I/O Adress" (A_IOAD) abgespeichert ist, in ein Array von Strings um. Dieses Array ist folgendermaßen aufgebaut:

  Daten(0) = OPC Server Name
  Daten(1) = OPC Group Name
  Daten(2) = In diesem DB ist der Tag gespeichert.
  Daten(3) = Typ der Variable, z.B. X,B,D
  Daten(4) = Zeigt an, welches Byte oder Wort es ist.
  Daten(5) = Zeigt entweder das Bit an oder den Zusatz.
  Daten(6) = AG-Name
  Daten(7) = Zeigt an, ob die Syntax einer S7 oder ein S5 entspricht

*UF_**

- alle Methoden, die mit diesem Kürzel anfangen, werden durch Events von der jeweiligen Userform ausgeführt. Wenn im Namen kein Kürzel von einem der Standardelemente enthalten ist, dann wird sie von allen Userformen ausgeführt. Dies ist möglich, da ähnliche Elemente gleich heißen.

UserFormInit

- die aktuelle Userform wird mit Standardwerten gefüllt. Diese Methode funktioniert nur bei den Userformen für die Standardelemente (frmOPCConfig_*)

VariantArrToStringArr

- wandelt ein Array aus Variant-Variablen, das mit Strings gefüllt ist, in ein Array aus Strings um. Dies funktioniert nur mit eindimensionalen Arrays.

3.7 Modul VDBA

*eda_**

- dies sind Funktionen der EDA-Schnittstelle. Die Funktionsbeschreibungen sind am Besten der Dokumentation dieser Schnittstelle zu entnehmen, da dort der aktuelle Stand hinterlegt ist.

FnGetProfile

- gibt einen einzelnen Wert in eine Initialisierungsdatei, bei Angabe von Dateiname, Sektion und Schlüssel, als String zurück.

FnSetProfile

- schreibt einen einzelnen Wert in eine Initialisierungsdatei, bei. Angabe von Dateiname, Sektion und Schlüssel.

INI_getSec

- gibt den Inhalt einer Sektion bei Angabe von Dateiname, Sektion und Anzahl der Zeichen als Array zurück.

INI_getSecNames

- gibt alle Sektionsnamen einer Initialisierungsdatei bei Angabe von Dateiname als Array zurück.

INI_GetString

- gibt einen einzelnen Wert in eine Initialisierungsdatei bei Angabe von Dateiname, Sektion, Schlüssel und Anzahl von Zeichen zurück.

INI_setSec

- schreibt den Inhalt eines Arrays bei Angabe von Dateiname und Sektion in eine Initialisierungsdatei.

INI_SetString

- schreibt einen einzelnen Wert in eine Initialisierungsdatei bei, Angabe von Dateiname, Sektion und Schlüssel.

3.8 Einschränkungen

Dieses Programm wurde möglichst variabel geschrieben, dennoch gibt es gewisse Einschränkungen bezüglich der Erstellung der Tags. Da sich Bedienerfreundlichkeit und ein möglichst hoher Freiheitsgrad bei der Parametrierung disproportional zueinander verhalten, muss der Benutzer bei der Bedienung der Oberfläche eingeschränkt werden. Ähnlich verhält es sich bei der Kompatibilität und Komplexität des Programms, da aus einer erhöhten Variabilität zwar eine bessere Anpassbarkeit an bestehen Systeme resultiert, jedoch auch die Fehlerquote steigt.

- Es können nur Standardelemente erzeugt werden, da ein Teil der Parameter direkt im Programmcode hinterlegt wurde. Dann kann dieses Programm nur innerhalb der iFIX™-Umgebung arbeiten, da elementare Funktionen von dieser Software genutzt werden.
- Da unterschiedliche Steuerungen im OPC-Treiber und in der Datenbank unterschiedlich angesprochen werden, funktioniert dieses Programm nur mit Steuerungen von Siemens, sprich S7 und S5.
- Bei S5en können keine analogen Inputs erstellt werden, da für diese Art von Steuerungen keine eigenen Bausteine vorgesehen sind.
- Sollte ein Typ von den Standardelementen auf mehr als ein Datenbaustein verteilt sein, so kann nur ein Baustein unterstützt werden.
- Der SIX-Treiber wird nicht parametriert, da dieser keine Schnittstelle für eine externe Software bietet. Dies kann zu unerwarteten Fehlern führen.
- Die Tageigenschaften sind nicht für einzelne Automatisierungsgeräte einstellbar, abgesehen vom "Offset" und "Create".

4 Validierung, Verifikation und Evaluation

Validierung, Verifikation und Evaluation fanden während der gesamten Entwicklungsphase statt. Dadurch konnte schon früh dem größten Teil eventueller Unzulänglichkeiten entgegengewirkt werden. Nach Abschluss der Arbeit wurde das Tool im Ganzen auf eventuelle Fehler und das allgemeinen Verhalten überprüft.

4.1 Validierung und Verifikation

4.1.1 Definition Validierung

Der Prozess der Beurteilung eines Systems oder einer Komponente während oder am Ende des Entwicklungsprozesses, mit dem Ziel, festzustellen, ob die spezifizierten Anforderungen erfüllt sind.

Validierung beantwortet also die Frage: Entwickeln wir das richtige System?

4.1.2 Definition Verifikation

Der Prozess der Beurteilung eines Systems oder einer Komponente mit dem Ziel, festzustellen, ob die Resultate einer gegebenen Entwicklungsphase den Vorgaben für diese Phase entsprechen.

Verifikation beantwortet die Frage: Entwickeln wir das System richtig?

[IEEE 610.12].

Dementsprechend wird der Tagvergleich (S.50) der Verifikation zugeordnet und die Implementierung (S.51) bzw. die Evaluation (S.52) eher der Validierung.

4.1.3 Durchführung

Um schon während der Entwicklung alle Funktionen, Methoden und Softwareschnittstellen (API) testen zu können, wurde ein Projekt mit den Konfigurationen aus der Brauerei in Żywiec (Polen) aufgesetzt und mit einer SPS verbunden. Die Anlage läuft original nur mit einem OPC-Client. Deshalb wurde hier noch zusätzlich ein SIX-Treiber installiert, um eventuelle Störungen durch diese Softwarekomponente frühzeitig zu erkennen. Durch die angeschlossene SPS konnte eine Kopplung bis in die Steuerungsebene überprüft werden.

4.1.4 Tag Vergleich

Nr.	Feldname	Feld-Kurzbezeichung	Excel Makro	Beispiel Project	VBA-Script	Vergleich EXCEL - VBA	Vergleich Project- VBA
1	[BLOCK TYPE	!A_NAME	AI	AI	AI	1	1
2	TAG	A_TAG	PID703_SCALE_MAX	PID703_SCALE_MAX	PID703_SCALE_MAX	1	1
3	NEXT BLK	A_NEXT				1	1
4	DESCRIPTION	A_DESC	Upper Limit of Scale	Upper Limit of Scale	Upper Limit of Scale	1	1
5	INITIAL SCAN	A_ISCAN	ON	ON	ON	1	1
6	SCAN TIME	A_SCANT	1	1	1	1	1
7	SMOOTHING	A_SMOTH	0	0	0	1	1
8	I/O DEVICE	A_IODV	OPC	OPC	OPC	1	1
9	H/W OPTIONS	A_IOHT	Server	Server	Server	1	1
10	I/O ADDRESS	A_IOAD	PID;PID703;DB6,D572IEEE;FLTS7	PID;PID703;DB6,D572IEEE;FLTS7	PID;PID703;DB6,D572IEEE;FLTS7	1	1
11	SIGNAL CONDITIONING	A_IOSC	NONE	None	None	1	1
12	LOW EGU LIMIT	A_ELO	-99999,99	-99999,99	-99999,99	1	1
13	HIGH EGU LIMIT	A_EHI	99999,99	99999,99	99999,99	1	1
14	EGU TAG	A_EGUDESC				1	1
15	INITIAL A/M STATUS	A_IAM	AUTO	AUTO	AUTO	1	1
16	ALARM ENABLE	A_IENAB	DISABLE	DISABLE	DISABLE	1	1
17	ALARM AREA(S)	A_ADI	NONE	NONE	NONE	1	1
18	LO LO ALARM LIMIT	A_LOLO	-99999,99	-99999,99	-99999,99	1	1
19	LO ALARM LIMIT	A_LO	-99999,99	-99999,99	-99999,99	1	1
20	HI ALARM LIMIT	A_HI	99999,99	99999,99	99999,99	1	1
21	HI HI ALARM LIMIT	A_HIHI	99999,99	99999,99	99999,99	1	1
22	ROC ALARM LIMIT	A_ROC	0	0	0	1	1
23	DEAD BAND	A_DBAND	0	10000	10000	0	1
24	ALARM PRIORITY	A_PRI	L	L	L	1	1
25	ENABLE OUTPUT	A_EOUT	YES	YES	YES	1	1
26	SECURITY AREA 1	A_SA1	NONE	NONE	NONE	1	1
27	SECURITY AREA 2	A_SA2	NONE	NONE	NONE	1	1
28	SECURITY AREA 3	A_SA3	NONE	NONE	NONE	1	1
29	ALARM AREA 1	A_AREA1	ALL	SCADA3	SCADA3	0	1
30	ALARM AREA 2	A_AREA2		FLTS7	FLTS7	0	1
31	ALARM AREA 3	A_AREA3				1	1
32	ALARM AREA 4	A_AREA4				1	1
33	ALARM AREA 5	A_AREA5				1	1
34	ALARM AREA 6	A_AREA6				1	1
35	ALARM AREA 7	A_AREA7				1	1
36	ALARM AREA 8	A_AREA8				1	1
37	ALARM AREA 9	A_AREA9				1	1
38	ALARM AREA 10	A_AREA10				1	1
39	ALARM AREA 11	A_AREA11				1	1
40	ALARM AREA 12	A_AREA12				1	1
41	ALARM AREA 13	A_AREA13				1	1
42	ALARM AREA 14	A_AREA14				1	1
43	ALARM AREA 15	A_AREA15				1	1
44	USER FIELD 1	A_ALMEXT1				1	1
45	USER FIELD 2	A_ALMEXT2				1	1
46	ESIG TYPE	A_ESIGTYPE	NONE	NONE	NONE	1	1
47	ESIG ALLOW CONT USE	A_ESIGCONT	YES	YES	YES	1	1
48	ESIG XMPT ALARM ACK	A_ESIGACK	NO	NO	NO	1	1
49	ESIG UNSIGNED WRITES]	A_ESIGTRAP!	REJECT	REJECT	REJECT	1	1
					Summe:	46	49
					Prozentualle Übereinstimmung:	94	100

Tabelle 1: Tagvergleich Beispielprojekt - Otas_Dialog_Tool - Programm

Um die Ergebnisse des VB-Skriptes zu validieren, wurden alle erzeugten Tags zum einen mit den erzeugten Tags von OTAS_DIALOG_TOOL und zum anderen mit dem Beispielprojekt verglichen (Tabelle 1). Es wurden Testobjekte mit dem OTAS_DIALOG_TOOL erstellt und ähnliche Objekte aus dem Beispielprojekt exportiert Nach dem Erzeugen der Database Wizard-Vergleichsobjekte, wurden auch diese exportiert.

Dabei wird deutlich, dass sich schon das bestehende Projekt und die Excel-Tabelle an einigen Stellen unterschieden. Das rührt daher, dass OTAS_DIALOG_TOOL nicht auf dem aktuellen Standard ist. Nach Rücksprache mit einigen Mitarbeitern wurde beschlossen, dass das Beispielprojekt als Standard anzusehen ist.

4.1.5 Praxistest

Etwa zur Mitte der Diplomarbeit wurde der Database Wizard in Tychy (Polen) bei der Brauerei Tysky an einer realen Anlage getestet. Dort wurde zu dem Zeitpunkt eine Inbetriebnahme durchgeführt. Nach der Integration des Database Wizard in die bestehende Anlage und einigen Tests, kam die Software auch gleich zum Einsatz. Es wurde ein digitaler Input für einen Tank nachgepflegt. Da die Software noch nicht vollständig war, konnten nur grundlegende Funktionen getestet werden. Nach einigen Anpassungen haben alle getesteten Funktionen des Tools funktioniert, bis auf das Löschen von Items im OPC-Client. Bei späteren Versuchen am Testsystem im Büro konnte auch für dieses Problem eine Lösung gefunden werden.

Anfang Juni, gab es einen neuen Auftrag von der Brauerei Tysky. Bei dem bestehenden System sollen die S5en durch S7en ausgetauscht werden. Durch die erhöhte Rechenkapazität der S7, reduziert sich auch die Anzahl der Steuerungen von drei auf zwei. Da sich nichts an der Anlage ändert außer der Steuerung, kann man die alten Fließbilder beibehalten. Sie müssen einzig neu verknüpft werden.

Für die Projektierung heißt das, dass die komplette Datenbank sowie der OPC-Client neu angelegt werden müssen und dass die Bilder neu zu verknüpfen sind. Für eine so hohe Anzahl an neu zu erstellenden Datenpunkten ist der Database Wizard nicht zu empfehlen, da OTAS_Dialog_Tool deutlich schneller ist. Bei der Verknüpfung der Bilder oder bei späteren Änderungen stellt jedoch der Database Wizard eine gute Alternative dar.

Deshalb wurde dieser in das neue Projekt integriert und getestet. Nach einigen Anpassungen in der INI-Datei funktionierte das Tool problemlos.

Festzuhalten ist, dass der VBA-Code möglichst variabel konzipiert wurde. Alle wichtigen Parameter sind direkt über die INI-Datei änderbar und die Software kann relativ schnell und verständlich in das bestehende System implementiert werden.

4.2 Evaluation

Die Software-Ergonomie ist aus der Einsicht entstanden, dass aus den vielfältigen Gestaltungsmöglichkeiten, die moderne Technologien bereithalten, stets Systeme entwickelt werden müssen, die menschlichen Bedürfnissen und Anforderungen entsprechen. Neben Fragen der technischen Leistungsfähigkeit und Perfektion stellt die Anpassung von Computersystemen an den sie benutzenden Menschen eine entscheidende Gestaltungsaufgabe für alle an einer Entwicklung beteiligten Parteien dar. Die Umsetzung des Gestaltungsziels "Benutzerfreundlichkeit" von Softwaresystemen wird einerseits durch ein Angebot an Richtlinienkatalogen sowie nationalen und internationalen Standards unterstützt, andererseits zeigt die Praxis, dass die Entwicklung einer hochwertigen Benutzersoberfläche einen so komplexen Prozess darstellt, dass Evaluationsmaßnahmen in verschiedenen Entwicklungsstadien unverzichtbar sind. Die ergonomische Bewertung von Software kann dabei mit vielen Methoden erfolgen. Das Testen eines Produktes ist eine zentrale Methode im Rahmen der Qualitätssicherung bzw. benutzerorientierter Produktgestaltung. [Hegner 03]

4.2.1 Nutzen

Die Evaluation wird heute hauptsächlich als Mittel der Informationssammlung mit gestaltungsunterstützender Rolle innerhalb eines iterativen Software-Entwicklungsprozesses gesehen. Durch Evaluationsmethoden sollen Fehler und Schwachstellen des zu fertigenden Systems in einer frühen Entwicklungsphase aufgedeckt werden. Dabei wird das Ziel verfolgt, die Software an die Bedürfnisse der Nutzer anzupassen. [Hegner 03]

4.2.2 Definition

Wottawa definiert Evaluation "als das Sammeln und Kombinieren von Daten mit einem gewichteten Satz von Skalen, mit denen entweder vergleichende oder numerische Beurteilungen erlangt werden sollen" [Wottawa 01]

Auch für Görner und Ilg bedeutet die Evaluation ein systematisches Sammeln, Auswerten und Interpretieren von Daten, um eine reliable und valide Bewertung der Benutzungsschnittstelle zu ermöglichen. Dabei wird aus den Ergebnissen der Evaluation abgeleitet, ob ein vorab definiertes Ziel erreicht ist bzw. ob und wo weitere Verbesserungsmöglichkeiten ausgeschöpft werden können. [Görner & Ilg 93]

4.2.3 Durchführung

Um alle Funktionen der erstellten Software möglichst praxisnah zu testen, wurde ein Aufgabenblatt erstellt (Abbildung 35).

Aufgabenstellung

Aufgabe 1.
Ein Kunde hat einen neuen Lagertank bestellt. Die Hardwareplanung und die Implementierung in die SPS sind bereits geschehen. Nun muss der Tank noch in IFIX™ dargestellt werden.

Das Bild "Ubungsbild.grf" soll um den Tank erweitert werden. An diesen Tank sind eine Füllstandanzeige, eine Vollmeldung, eine Durchflussregelung zum Umwälzen des Inhalts und ein Befüll-/Auslassventil nachzupflegen. Der neue Tank ist baugleich zuTank1, abgesehen vom Motor, dem Durchflusssensor und natürlich den Eingängen an der SPS.
(Alle Aufgabenstellungen sollen mit dem Database Wizard gelöst werden) 

Daten:
Informationen zu den Komponentennamen entnehmen Sie bitte dem RI-Schema.

Füllstandanzeige:	an der 12. Stelle im DB30
Durchflussanzeige:	an der 15. Stelle im DB30
Befüllventil :	angeschlossen an Byte 30 Bit 1 abgelegt im DB52
Umwälzpumpe:	für die Umwälzpumpe existieren schon die entsprechenden Tags, sie müssen nur entsprechend verknüpft werden.
Vollmeldung:	Angeschlossen an Byte 10 Bit 2 Abgelegt im DB 10
Regler:	an der 218. Stelle im DB55 Maximale Durchflussmenge 60hl pro Stunde. Die Durchflussmenge soll mit zwei Nachkommstellen angezeigt werden. (größere Pumpe und besserer Sensor als Tank 1)

Aufgabe 2.
Aus finanziellen Gründen sollen die Pumpe und der Sensor von Tank 2 nun doch nur die Standardausführung bekommen. Das heißt für sie, dass der Durchflussregler genauso konfiguriert werden muss wie der Regler von Tank 1 (Nachkommstelle, maximaler Durchfluss).

Aufgabe 3.
Durch den geringeren Durchfluss wird jetzt eine Regelung überflüssig.
Löschen Sie den soeben erzeugten Regler und alle zugehörigen Tags.

Abbildung 35: Evaluation Aufgabenblatt

Damit dieser Test möglichst Praxisnah ist, wurde zusätzlich ein RI Schema erstellt. Da diese Schemata meist die Grundlage für die Fließbilder sind(Abbildung 36).

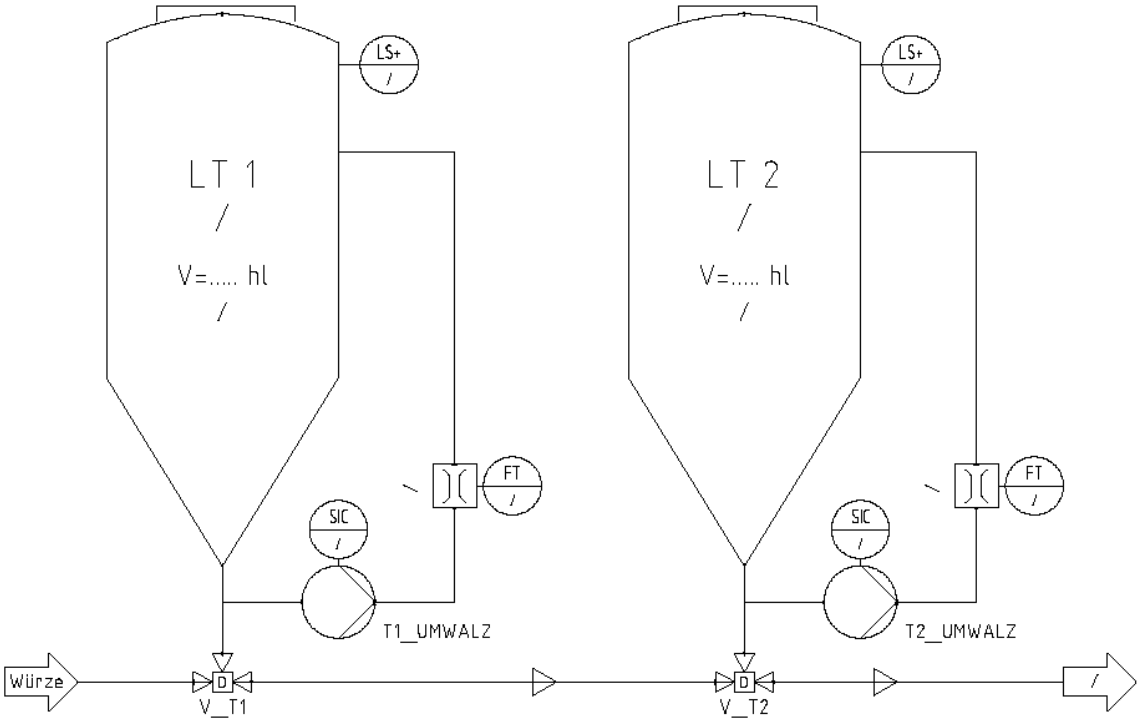


Abbildung 36: RI Schema

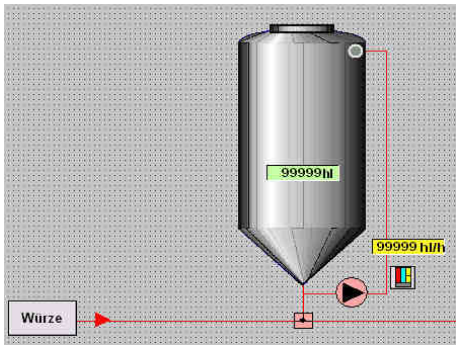


Abbildung 38: Fließbild vorher

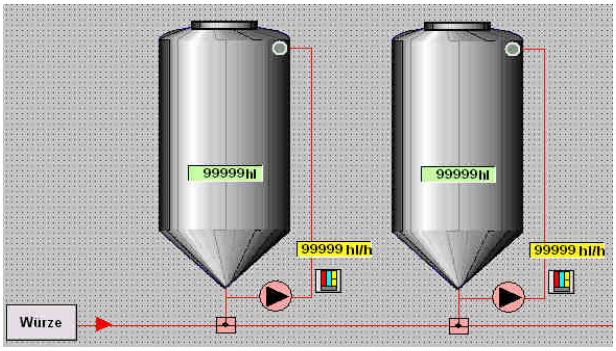


Abbildung 37: Fließbild nachher

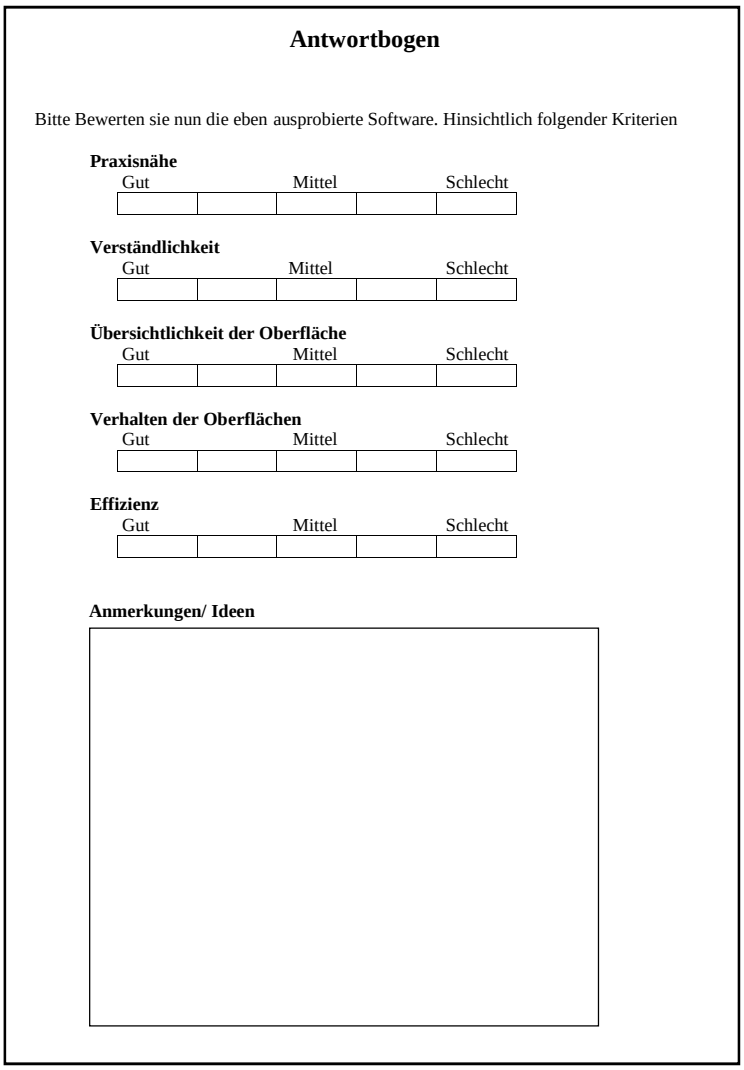
Es wurde ein Fließbild vom Tank1 vorbereitet (Abbildung 38). In dieses Fließbild wurde dann der Tank2 eingefügt (Abbildung 37). Bei diesem Beispiel sind alle Standardelemente vertreten. V_T1 und T1_Umwalz stehen für die Ventile und Motoren. Die Vollmeldung als binäres Signal für einen digitalen Input, der Füllstand und die Durchflussanzeige für einen analogen Input. An dem Durchflussregler kann die Konfiguration eines Reglers demonstriert werden.

Innerhalb der Übung zur Parametrierung der Bilder, sollen die unterschiedlichen Möglichkeiten des Database Wizard getestet werden. Aufgabe des Testers soll nicht sein, zu überprüfen, ob alle Verknüpfungen richtig erstellt wurden. Während des Tests wurden

die Probanden beobachtet, um eventuelles Fehlverhalten der Software zu erkennen und wenn möglich zu beseitigen. So war zum Beispiel in einem der Fenster eine Schalfläche an einer anderen Stelle als in allen anderen Fenstern, was dann für ein wenig Verwirrung sorgte. Durch geringfügige Änderungen konnte dieser Mangel beseitigt werden.

4.2.4 Evaluationsergebnis

Nachdem die Übung abschlossen war, wurde von den Mitarbeitern ein Evaluationsantwortbogen ausgefüllt, der zum einem die Software bewerten, zum anderen Platz für Anregungen und Ideen geben sollte (Abbildung 39).



Antwortbogen

Bitte Bewerten sie nun die eben ausprobierte Software. Hinsichtlich folgender Kriterien

Praxisnähe

Gut		Mittel		Schlecht

Verständlichkeit

Gut		Mittel		Schlecht

Übersichtlichkeit der Oberfläche

Gut		Mittel		Schlecht

Verhalten der Oberflächen

Gut		Mittel		Schlecht

Effizienz

Gut		Mittel		Schlecht

Anmerkungen/ Ideen

Abbildung 39: Evaluation Antwortbogen

Die Evaluation wurde von zehn Mittarbeitern durchgeführt, wobei die Gruppe sehr gemischt war. Alle Mitarbeiter der PC-Abteilung, einige aus dem Verkauf und der Hardwareplanung sowie aus dem Management und auch ein Subunternehmer waren an dem Test beteiligt.

Das Hauptaugenmerk lag dabei bei den späteren Anwendern, mit denen dann auch etwas tiefer in die Software eingestiegen wurde. Diese Übungsaufgaben hatten nicht nur den Zweck der Evaluation, sondern waren auch gleichzeitig eine kleine Schulung für die späteren Anwender.

Durch die geringe Anzahl der Probaten, die Durchführung und die Art der Evaluation, lässt sich keine sinnvolle statistische Aussage über diese treffen. Dennoch ist eine Tendenz zu erkennen (1 = gut, 5 = schlecht) und man kann sagen, dass die gesetzten Kriterien, zumindest subjektiv, erfüllt worden sind [Hegner 03].

Kriterien	Antworten										Ø
Praxisnähe	1	1	1	1	1	1	1	1	1	2	1,1
Verständlichkeit	1	1	1	2	1	1	1	1	1	2	1,2
Übersichtlichkeit der Oberfläche	1	1	1	1	2	1	1	1	1	1	1,1
Verhalten der Oberflächen	1	1	1	2	2	1	1	1	1	1	1,2
Effizienz	1	1	1	1	2	1	1	1	1	1	1,1

Tabelle 2: Auswertung Evaluation

Antworten aus dem Anmerkungen-/Ideen-Fenster:

- Bei analogen Inputs die zusätzlichen Tags deaktivierbar machen ✔
- Text für Start-Button ✔
- Sehr ordentliche Arbeit, super Design, alles in Ordnung.
- Möglichkeit für Laden und Speichern der INI-File (z.B. SIX und OPC)
- Gut intuitiv beschrieben
- Positionsänderung des Knopfes bei analogen Inputs ✔
- Nachkommastellen einstellbar bei analogen Inputs.

Die abgehakten Stichpunkte wurden bereits erledigt und alle anderen bedürfen größerer Änderungen und sind für später geplant.

5 Zusammenfassung

In der Vergangenheit gab es diverse Möglichkeiten, einzelne Prozesskomponenten in einem bestehenden Prozessvisualisierungssystem nachzupflegen. Jedoch setzte das nicht nur gute Kenntnisse über die Visualisierungssoftware und deren Komponenten voraus, sondern auch über die vor Ort herrschenden Normen und Standards. Das hatte zur Folge, dass nur Projektierer, die sich mit der Anlage auskannten, diese neuen Komponenten einfügen konnten oder sie mussten zumindest einen nicht unerheblichen Support leisten.

Deshalb sollte eine Software entwickelt werden, die das Parametrieren von Verknüpfungen erleichtert. Außerdem muss es möglichst unkompliziert an die meisten Anlagenkonfigurationen anpassbar sein. Als Programmiersprache wurde Visual Basic for Application gewählt, da diese zum einen einfach und effizient in bestehende Projekte eingebunden werden kann und zum anderen, weil schon Projekte in dieser Sprache realisiert wurden.

Die entstandene Software erzeugt alle Tags für die Standardelemente innerhalb der geforderten Parameter, inklusive der nötigen OPC-Items. Zudem werden die entsprechenden Einträge in die jeweiligen Templates getätigt, so dass der Benutzer sich auf die Eingabe der objektspezifischen Eigenschaften konzentrieren kann und sich nicht um die normgerechte Erzeugung der Verknüpfungen kümmern muss.

Während und nach der Programmierung des Werkzeugs wurde die Software validiert und evaluiert, um möglichst dicht an den Anforderungen zu sein. Die abschließende Evaluierung hatte nicht nur den Sinn der Bewertung des Produkts, sondern sollte auch gleichzeitig eine Schulung für spätere Nutzer darstellen.

Abgesehen von den Verbesserungsvorschlägen, die während der Evaluation gemacht wurden, könnte man sich vorstellen ein neues, variables "Standardelement" zu erzeugen. Als Beispiel könnte hier der Tank dienen. Dieser hat innerhalb einer Anlage, bzw. eines Anlagenbereiches, ähnliche Prozessdaten, ist aber von Kunde zu Kunde sehr unterschiedlich. Deshalb wäre die Möglichkeit vorstellbar, ein variables Objekt zu kreieren, welches nicht einem bestimmten Template zugordnet ist, sondern nur zur automatischen Generierung von Tags genutzt wird.

6 Fazit

Nach der Evaluierung und dem Praxistest kann ich sagen, dass die Aufgabenstellung: "Entwicklung eines Software-Werkzeugs, zur intuitiven Projektierung von Datenpunkten der Prozessvisualisierung" im Rahmen der Diplomarbeit erfüllt wurde. Eine endgültige Aussage über die Qualität lässt sich sicherlich erst nach einem längeren Praxistests treffen. Abgesehen davon bin ich überzeugt, dass die Entwicklung einer Software niemals ganz abgeschlossen ist und es stets etwas zu verbessern gibt. Dies muss grundsätzlich im Dialog mit dem Kunden bzw. dem späteren Anwender geschehen.

7 Glossar

AccesPath(OPC)

Der Zugriffspfad "AccesPath" ist eine optionale Information für Objekte der Klasse OPC-Item. Der Bedarf und die Interpretation eines Zugriffspfad spezifiziert, falls erforderlich, auf welchem Weg der Server die Daten einer Prozessvariablen lesen bzw. schreiben soll. Der Zugriffspfad kann zum Beispiel ein Pfad- oder Verbindungsname sein.

Client

Als Client bezeichnet man ein Programm, das auf einen Server zugreifen kann, also auf ein fremdes Programm, um dessen Dienste in Anspruch zu nehmen oder um Routine-aufgaben in der Serverapplikation zu automatisieren. [WEKA 03]

CSV-Datei

Dies ist eine einfache Textdatei, deren Werte durch einen Separator getrennt sind. Meist ist dies ein Semikolon. Diese Dateien werden meist als Parameter- oder Wertespeicher benutzt.

DCOM

Steht für Distributed Component Object Model und beschreibt ein Objektmodell für das Implementieren verteilter Anwendungen, entsprechend dem Client-Server-Paradiagram. Ein Client kann gleichzeitig mehrere Server nutzen und ein Server kann seine Funktionalität mehren Clients gleichzeitig zur Verfügung stellen. [Iwanitz&Lange 02]

Mit DCOM kann man Prozesse auf effiziente Weise über mehrere Computer verteilen, so dass die Client-Server-Komponenten einer Anwendung an optimalen Stellen im Netzwerk gespeichert werden können. Die Prozesse verlaufen transparent, so dass der Benutzer auf Informationen zugreifen und diese gemeinsam nutzen kann, ohne zu wissen, wo die Komponenten der Anwendung gespeichert sind. [Microsoft 98]

Factory Acceptance Tests

Der Factory Acceptance Test (FAT) dient der Qualitätssicherung. Es ist ein kompletter Softwarecheck, der schon beim Hersteller durchgeführt wird. Dabei wird die Anlage mit allen Ein- und Ausgängen simuliert.

iFIX™

IFIX™ ist eine Softwarelösung für industrielle Automatisierungsaufgaben und fasst die Funktionalität für die Prozessvisualisierung, Datensammlung und Überwachungssteuerung der Prozessumgebung zusammen. Damit deckt IFIX™ innerhalb der Intellution Dynamics-Produktfamilie, die Anforderungen für Visualisierungs- und Leitsystemaufgaben ab und stellt die gesammelten Produktionsinformationen unternehmensweit zur Verfügung [Tech 00] (Siehe S. 13).

INAT OPC-Server TCPIPH1

Ist ein All-in-One OPC-Server, der die wichtigsten, auf Ethernet aufbauenden, Industrieprotokolle beherrscht und dadurch für unterschiedlichste Systeme verwendet werden kann. Neben den Siemensprotokollen S7 und S5 über TCP/IP und H1, beherrscht er das EtherNet/IP-Protokoll, das bei Allen-Bradley-Steuerungen zum Einsatz kommt. Weiterhin kommuniziert der Server mit Ethernet-Anschaltungen, von Wago, Beckhoff, Phoenix Contact usw. über das Modbus on TCP-Protokoll. Zu Geräten, die keines der genannten Protokolle unterstützen, ist es möglich, über Send/Receive-Kommunikation Daten zu übertragen. [INAT 05]

OPC

"**O**LE for **P**rocess **C**ontrol" ist ein IndustrieStandard, der von verschiedenen Firmen der Automatisierungstechnik, in Zusammenarbeit mit der Firma Microsoft, geschaffen wurde, um einen standardisierten Datenaustausch unterschiedlicher Automatisierungssysteme zu gewährleisten. [Iwanitz&Lange 02]

OPC-Client

Ein OPC-Client als Nutzer der Dienste ist nicht auf einen Server beschränkt, sondern kann im Rahmen der Leistungsfähigkeit des Systems beliebig viele OPC-Server nutzen. Da die Art des Datenzugriffs für alle OPC-Server gleich ist, kann mit vergleichsweise geringem Aufwand ein OPC-Server gegen ein Produkt eines anderen Herstellers ausgetauscht werden.[Langmann 07] In der Vorliegenden Diplomarbeit stellt "Powertool" den OPC Client dar.

OPC-Gruppe

Eine OPC-Gruppe ist eine logische Einheit zur Strukturierung der von einem Client genutzten OPC-Items. OPC-Gruppen werden von einem Client angelegt und können ein oder mehrere OPC-Items enthalten. Auf die OPC-Items einer Gruppe können Mengenaufrufe angewendet werden.

OPC-Item-ID

Die OPC-Item-Id ist ein String, der eine Prozessvariable eindeutig identifiziert. Die Syntax der ID ist vom Server abhängig. Die OPC-Item-ID gibt dem Server eindeutig an, welche Prozessvariable dem OPC-Item zugeordnet wird.

OPC-Server

Ein OPC -Server wird von einem Hersteller als "Dienst-Erbringer" zum Zugriff auf Daten bereitgestellt. Verschiedene Hersteller bieten unterschiedliche OPC-Server mit unterschiedlichen Eigenschaften für verschiedene Einsatzgebiete an. Ein OPC-Server könnte beispielsweise die über den PROFIBUS erreichbaren Daten anbieten, ein anderer könnte den Zugang zu einer speicherprogrammierbaren Steuerung herstellen. [Langmann 07]

OLE

OLE ist die Abkürzung für **O**bject **L**inking and **E**mbedding. Dieser Begriff bezeichnete bis in die Mitte der 90er Jahre die gesamte Microsoft-Technologie für Realisierung objektorientierter Systeme. [Iwanitz&Lange 02]

OTAS_Dialog_TOOL

OTAS_Dialog_TOOL ist eine Excel-Datei die automatisch CSV-Dateien generiert. Dies CSV-Dateien können dann in den OPC-Client und in die Datenbank importiert werden.

Prozessdatenbank

Eine Datei die alle Prozessdaten beinhaltet. Sie ist die Hauptquelle für die meisten iFIX™ Anwendungen. [iFIX 99]

SCADA Server

Ein SCADA (Supervisory, Control, and Data Acquisition) Knoten ist ein Computer, der eine Prozessdatenbank geladen hat und Prozessinformationen von einer oder mehreren Steuerungen sammelt. Ein SCADA Server kann Daten in das Automatisierungsgerät zurückschreiben, um den Prozess zu automatisieren und zu steuern. [iFIX 99]

Server

Ein Server ist ein Computer in einem Netzwerk, der einen Dienst oder Ressourcen für einen Client bereitstellt[Gupta 95]

Tag

In der Datenverarbeitung und Informatik steht das englische Wort Tag [tæg] (Etikett, Anhänger, Aufkleber, Marke, Auszeichner) für die Auszeichnung eines Datenbestandes mit zusätzlichen Informationen. Die darin enthaltenen Informationen dienen je nach Verwendungsgebiet sehr unterschiedlichen Zwecken. [WIKI 07a]

In dieser Diplomarbeit meint ein Tag einen speziellen Eintrag in der Datenbank.

UserForm

Die UserForm ist ein Objekt von VBA und repräsentiert die Fenster. Dies Dialogfenster hilft bei der täglichen Arbeit mit den unterschiedlichsten Anwendungsprogrammen. Mit ihrer Hilfe wird man in die Lage versetzt mit den Anwendungen zu "kommunizieren". Man kann Einstellungen vornehmen, Werte eingeben, Fragen beantworten usw., um sich danach die gewünschten Informationen anzeigen zu lassen oder entsprechende Aktionen auszuführen.

VARIANT

Ein VARIANT ist ein spezieller Datentyp, der vorwiegend in OLE-Automation verwendet wird. Eine Variable vom Typ VARIANT kann Werte verschiedener Variablentypen sowiedie speziellen Werte Empty, Error und Null repäsentieren.

WinCC

PC-basiertes Bedien- und Beobachtungssystem für das Visualisieren und Bedienen von Prozessen, Fertigungsabläufen, Maschinen und Anlagen in allen Branchen, mit einfachen Einplatz- bis hin zu verteilten Mehrplatzsystemen, mit redundanten Servern und standortübergreifenden Lösungen mit Web-Clients(siehe auch Seite 15).[Siemens 07]

8 Literaturverzeichnis

[Lauer&Göher 99a] Lauber, R. und Göhner, P.: Prozessautomatisierung 1, 3. Auflage Berlin-Heidelberg: Springer 1999

[Lauer&Göher 99b] Lauber, R. und Göhner, P.: Prozessautomatisierung 2, Berlin-Heidelberg: Springer 1999

[Bitzer 91] Bitzer, B.: Prozessvisualisierung mit dem Industrie-PC: Überwachen und Steuern mit dem Industrie PC-Grafik, Würzburg: Vogel 1999

[Iwanitz&Lange 02] Iwanitz, F. und Lange, J.: OPC- Grundlagen, Implementierung und Anwendung, 2.Auflage Heidelberg: Hüthing, 2002

[WEKA 03] WEKA MEDIA Gmbh & Co. KG: MS Office Troubleshooting VB-Script Referenz, Königsbrun: Offsetdruckerei Aldo Marzorati 2003

[Tuch 07] http://www.gea-brewery.com , 25.07.2007

[GEAG 07] http://www.geagroup.com, 30.06.2007

[Hegner 03] Hegner,M.: Methoden zur Evaluation von Software (IZ-Arbeitsbericht Nr. 29), Bonn: Informationszentrum Sozialwissenschaften 2003

[Wottawa 01] Evaluation. In Weidemann, B., Krapp, A.: Pädagogische Psychologie (4., vollst. überarb. Aufl). Weinheim : Beltz PVU 2001

[Tech 00] Werbeprospekt PN122DEv1 – IFIX Tech – 02/2000

[INAT 05] Datenblatt: INAT OPC-Server TCPIPH1 Ethernet 01/2005

[Siemens 07] Katalogdaten-Version Siemens AG, DE.3564.DE.102 , 02.07.2007

[WIKI 07a] http://de.wikipedia.org/wiki/Tag_%28Informatik%29, 31.07.07

[Gupta 95] Gupta Corporation: SQL Application Programming Interface Reference, Version 6.0, 20-2111-1003 1995

[Microsoft 98] Microsoft Press Deutschland: Microsoft Windows NT Server – Netzwerk, Version 4 Kempten: Kösel 1998

[iFIX 99] Intellution®, Inc.: Using VBA With iFIX, V2.1-7.99, Norwood, MA 1999

[Langmann 07] Prof. Dr.-Ing. Langmann, Einführung in die OPC-Technik, "http://tww.fh-duesseldorf.de/et/langmann/seminarprojekte/ si-ham/einf%FChrung_in_die_opc.htm" 10.06.2007

[iFIX 03] GE Fanuc International, Inc.: Elektronisches Handbuch: "Ein Leitsystem aufbauen", iFIX Version 3.5 - 07.2003

[IEEE 610.12] IEEE Std 610.12-1990 IEEE Standard Glossary of Software Engineering Terminology

[Görner & Ilg 93] Görner, C., Ilg, R.: Evaluation der Mensch-Rechner-Schnittstelle. Oldenbourg 1993

9 Abbildungsverzeichnis

10 Abkürzungsverzeichnis

AG	Automatisierungsgerät
AI	Analog Input Block
API	Application Program / Programming Interface
ASCII	American Standard Code for Information Interchange
CSV	Comma Separated Values
DB	Datenbaustein
DCOM	Distributed Component Object Model
DI	Digital Input Block
DLL	Dynamic Link Library
EDA	Easy Database Access
FAT	Factory Acceptance Tests
FIX	Fully Integrated Control System
GEA	Global Engineering Alliance
I/O	Input / Output
LAN	Local Area Network
ODBC	Open Database Connectivity
OPC	OLE for Process Control
OTAS	Offene Tuchenhagen Automatisierungssoftware
PDB	Prozessdatenbank
PC	Personal Computer
PID	Proportional-Integral-Differential Regler
S7/S5	Step 7 / Step 5
SAC	Scan, Alarm and Control Program
SCADA	Supervisory Control and Data Acquisition
SPS	Speicherprogrammierbare Steuerung
TBS	Tuchenhagen Brewery Systems
TLB	Type Library
UDT	Anwenderdefinierter Datentyp
VM	Ventile und Motoren

11 Anhang

Auszug (1 von 2) INI-Datei - Abschnitt Common

[COMMON]								
OPC	=	True						
VM_Info	=	**name**	**archive only**	**suffix**	**Close Disc**	**Open Disc**	**Category**	
VM_0	=	error	DISABLE	_ALM	Error	Ok	Alarm	
VM_1	=	manual activation	ENABLE	_MAN	On	Off	Manual	
VM_2	=	automatic activation	ENABLE	_AUTO	On	Off	Automatic	
VM_3	=	manual interlock	ENABLE	_MANLCK	On	Off	Man_interlock	
VM_4	=	automatic interlock	ENABLE	_LOCK	On	Off	Aut_Interlock	
VM_5	=	maintenance activation	ENABLE	_MAINT	On	Off	Main_Act	
VM_6	=	maintenance interlock	ENABLE	_MAINTLCK	On	Off	Main_Int	
VM_7	=	overwrite errors	ENABLE	_ERR_OVR	On	Off	ALL	
VM_8	=	active feedback	ENABLE	_FBA	On	Off		
VM_9	=	inactive feedback	ENABLE	_FBI	On	Off		
AI_Info	=	**name**	**archive only**	**suffix**	**Close Disc**	**Open Disc**	**Category**	
AI_0	=	Broken Wire	DISABLE	_BW	BROKEN WIRE	WIRE OK		
AI_1	=	Over Flow	DISABLE	_OV	OVERFLOW	OK		
AI_2	=	Simulation	ENABLE	_SM	SIMULATION ON	SIMULATION OFF		
PID_AI	=	**TYP**	**A_TAG**	**A_NEXT**	**A_DESC**	**A_ISCAN**	**A_SCANT**	**A_SMOTH**
PID_0	=	AI	%Name_W	%Name_W_NORM	Effective Setpoint			
PID_1	=	AI	%Name_W_INT		Setpoint from PC			
PID_2	=	AI	%Name_W_EXT		Setpoint from PLC			
PID_3	=	AI	%Name_X	%Name_X_NORM	Current Value			
PID_4	=	AI	%Name_Y_RESULT	%Name_Y_NORM	Effective Output Value			
PID_5	=	AI	%Name_SCALE_MIN		Lower Limit of Scale			
PID_6	=	AI	%Name_SCALE_MAX		Upper Limit of Scale			
PID_AR	=	**TYP**	**A_TAG**	**A_DESC**	**A_IODV**	**A_IOHT**	**A_NUMS**	**A_IOAD**
PID_7	=	AR	%Name_CMDB	Commandbyte	%Driver			%IOADB%OffSet
PID_8	=	AR	%Name_CYCLE	Cycle Time (sample time)	%Driver			%IOADD%OffSetKF
PID_9	=	AR	%Name_DEADBAND	Deadband Width	%Driver			%IOADD%OffSetIEEE
PID_10	=	AR	%Name_KP	Gain	%Driver			%IOADD%OffSetIEEE
PID_11	=	AR	%Name_PARAB	Parameterbyte	%Driver			%IOADB%OffSet
PID_12	=	AR	%Name_SPLITRANGE_I	Lower Splitrange Value	%Driver			%IOADD%OffSetIEEE
PID_13	=	AR	%Name_SPLITRANGE_II	Upper Splitrange Value	%Driver			%IOADD%OffSetIEEE
PID_14	=	AR	%Name_STATB	Statusbyte	%Driver			%IOADB%OffSet
PID_15	=	AR	%Name_TD	Derivative Time (Tv)	%Driver			%IOADD%OffSetKF
PID_16	=	AR	%Name_TI	Reset Time (Tn)	%Driver			%IOADD%OffSetKF
PID_17	=	AR	%Name_Y_HAND	Output Value direct from PC	%Driver			%IOADD%OffSetIEEE
PID_18	=	AR	%Name_Y_MAX	Upper Limit Output Value	%Driver			%IOADD%OffSetIEEE
PID_19	=	AR	%Name_Y_MIN	Lower Limit Output Value	%Driver			%IOADD%OffSetIEEE
PID_20	=	AR	%Name_Y_NF	Output Value direct from PLC	%Driver			%IOADD%OffSetIEEE
PID_21	=	AR	%Name_Y_OFF	Output Value if PID OFF	%Driver			%IOADD%OffSetIEEE
PID_ETR	=	TYP	A_TAG	A_NEXT	A_GET1	A_ELO	A_EHI	A_EGUDESC
PID_22	=	ETR	%Name_Y_TRND		%Name_Y_NORM.F_CV			
PID_23	=	ETR	%Name_W_TRND		%Name_W_NORM.F_CV			
PID_24	=	ETR	%Name_X_TRND		%Name_X_NORM.F_CV			
PID_DR	=	TYP	A_TAG	A_DESC	A_IODV	A_IOHT	A_NUMS	A_IOAD
PID_25	=	DR	%Name_PARA	Bitarray Parameter	%Driver			%IOADB%OffSet.1BA
PID_26	=	DR	%Name_STAT	Bitarray Status	%Driver			%IOADB%OffSet.1BA
PID_27	=	DR	%Name_CMD	%Discription	%Driver			%IOADB%OffSet.1BA
PID_CA	=	TYP	A_TAG	A_NEXT	A_DESC	A_GET2	A_GET3	A_GET4
PID_28	=	CA	%Name_X_NORM	%Name_X_TRND	Current Value 0-100%	%Name_SCALE_MIN.F_(	%Name_SCALE_M	100
PID_29	=	CA	%Name_W_NORM	%Name_W_TRND	Setpoint 0-100%	%Name_SCALE_MIN.F_(	%Name_SCALE_M	100
PID_30	=	CA	%Name_Y_NORM	%Name_Y_TRND	Output Value 0-100%	%Name_STAT.F_6	100	-100
AlarmArea1	=	Nodename						
AlarmArea2	=	Accesspath						
AlarmArea3	=	Category						
AlarmArea4	=							
Alarm_DI	=	Message						

Auszug (2 von 2) INI-Datei - Abschnitte Steuerungen

[GKS1]																				
AGTyp	= S7																			
Driver	= OPC																			
NrSPS	= 1																			
DI_Server	= IO																			
DI_Group	= DI																			
DI_DB	= 10																			
VM_info	= OPC Server	OPC Group	DB	VM nr:Offset	VM nr:Offset	VM nr:Offset														
VM_Common	= VM	GKS1	500	0:0.0	1:0.1															
AI_info	= OPC Server	OPC Group	DB	Start Offset	Offset															
AI_Common	= IO	AI	30	0	6	0:4.0	1:4.7	2:4.1												
PID_info	= OPC_Server	QPC_Group	DB	Start Offset	Offset	PID_Nr:Offset														
PID_Common	= PID	%Name	6	20	200	0:6	1:136	2:132	3:10	4:156	5:148	6:152	7:130	8:2	9:36					
[GKS2]																				
AGTyp	= S7																			
Driver	= OPC																			
NrSPS	= 2																			
DI_Server	= IO																			
DI_Group	= DI																			
DI_DB	= 10																			
VM_info	= OPC Server	OPC Group	DB	VM nr:Offset	VM nr:Offset	VM nr:Offset														
VM_Common	= VM	GKS2	500	0:0.0	1:0.1															
AI_info	= OPC Server	OPC Group	DB	Start Offset	Offset															
AI_Common	= IO	AI	30	0	6	0:4.0	1:4.7	2:4.1												
PID_info	= OPC_Server	OPC_Group	DB	Start Offset	Offset	PID_Nr:Offset														
PID_Common	= PID	%Name	6	20	200	0:6	1:136	2:132	3:10	4:156	5:148	6:152	7:130	8:2	9:36					

Auszug Datenbaustein VM

SIMATIC 7_41dt_engl\VM\...\DB50 - <offline> 09.08.2007 11:38:24

DB50 - <offline> - Deklarationssicht

"VM_LOCK_DB"
Globaler Datenbaustein DB 50
Name: VM_AL Familie: TBS
Autor: TBS Version: 1.0
Bausteinversion: 2
Zeitstempel Code: 05.08.2005 13:35:43
Interface: 04.08.2005 10:01:19
Längen (Baustein / Code / Daten): 24474 20258 00000

Objekteigenschaften:
S7_language 7(1) Deutsch (Deutschland) 28.06.2007 11:18:17

Baustein: DB50

Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	STAT0	BOOL	FALSE	
+0.1	STAT1	BOOL	FALSE	
+0.2	STAT2	BOOL	FALSE	
+0.3	STAT3	BOOL	FALSE	
+0.4	STAT4	BOOL	FALSE	
+0.5	STAT5	BOOL	FALSE	

UDT Block für VM Statusvariable

SIMATIC 7_41dt_engl\VM\...\UDT50 - <offline> 09.08.2007 11:37:36

UDT50 - <offline>

"VM_UDT"
Name: Familie:
Autor: Version: 0.0
Bausteinversion: 2
Zeitstempel Code: 14.10.2006 10:01:15
Interface: 05.08.2005 08:53:26
Längen (Baustein / Code / Daten): 00000 00000 00000
UNLINKED

Objekteigenschaften:
S7_language 7(1) Deutsch (Deutschland) 28.06.2007 11:18:17

Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	Alarm_verzoegert	BOOL	FALSE	Alarm verzögert
+0.1	Hand_ein	BOOL	FALSE	Steuerung und Status VM/ESG Hand - ein
+0.2	Automatik_ein	BOOL	FALSE	Steuerung und Status VM/ESG Automatik - ein
+0.3	Manuell_verriegelt	BOOL	FALSE	Steuerung und Status VM/ESG Manuell verriegelt
+0.4	Automatik_verriegelt	BOOL	FALSE	Steuerung und Status VM/ESG Automatik verriegelt
+0.5	Wartung_ein	BOOL	FALSE	Steuerung und Status VM/ESG Wartung - ein
+0.6	Wartung_verriegelt	BOOL	FALSE	Steuerung und Status VM/ESG Wartung verriegelt
+0.7	Ausgangsansteuerung	BOOL	FALSE	Ausgangsansteuerung VM/ESG - Ansteuerung
+1.0	Alarm_manuell_gesperrt	BOOL	FALSE	Alarm manuell unterdrückt
+1.1	Alarm_angesteuert	BOOL	FALSE	Alarm im angesteuerten Zustand
+1.2	Alarm_abgesteuert	BOOL	FALSE	Alarm im abgesteuerten Zustand
+1.3	Alarm	BOOL	FALSE	Alarm
+1.4	Alarmspeicher	BOOL	FALSE	Gespeicherter alarm
+1.5	aktive_Rueckmeldung	BOOL	FALSE	Punktionsbit VM/ESG ist angesteuert
+1.6	inaktiv_Rueckmeldung	BOOL	FALSE	Punktionsbit VM/ESG ist inaktiv
+1.7	Hand_Wartung_aktiv	BOOL	FALSE	VM/ESG ist unter Hand- oder Wartungskontrolle
+2.0	SW_Ueberwachungszeit	BYTE	B#16#F	Sollwert Überwachungszeit VM/ESG - Bewegung ein und aus
+3.0	IW_Ueberwachungszeit	BYTE	B#16#F	istwert Überwachungszeit VM/ESG - Bewegung ein und aus
+4.0	SW_Ein_Verzoegerung	BYTE	B#16#0	Sollwert Einschaltverzögerung
+5.0	SW_Aus_Verzoegerung	BYTE	B#16#0	Sollwert Ausschaltverzögerung
+6.0	IW_E_A_Verzoegerung	BYTE	B#16#0	Aktualwert Ein- oder ausschaltverzögerung
+7.0	Speicher_VM_ein	BOOL	FALSE	Speicher Ein-Zustand fürAusschaltverzögerung
+7.1	Speicher_Status	BOOL	FALSE	Speicher Status
+7.2	VM_ein_ohne_Ausgabe_DB	BOOL	FALSE	1= Ausgabe auf Ausgang direkt ohne Digitalausgabe - DB
+7.3	QUIT_WINCC	BOOL	FALSE	Quittiersignal für Alarmsynchronisation WinCC
+7.4	Umlauf_Kurzstoerung_1	BOOL	FALSE	Speicher für Kurzzeit - Fehlerunterdrückung
+7.5	Umlauf_Kurzstoerung_2	BOOL	FALSE	Speicher für Kurzzeit - Fehlerunterdrückung
+7.6	Verzoegerung_ein	BOOL	FALSE	1= Einschaltung der Kurzzeit - Fehlerunterdrückung (Möglich keit Vermischung!)
+7.7	Profibusfehler	BOOL	FALSE	Profibusfehler zur zeit ohne Punktion
+8.0	VM_Schaltspielzaehler	WORD	W#16#0	Schaltspielzähler
+10.0	VM_Betriebszeit	DWORD	DW#16#0	Speicher Betriebszeit
=14.0		END_STRUCT		

UDT Block für Regler

SIMATIC 7_41dt_engl\Regler\...\UDT6 - <offline> 09.08.2007 11:29:49

UDT6 - <offline>

"Regulator UDT for DB6"
Name: Familie:
Autor: tp Version: 1.0
Bausteinversion: 2
Zeitstempel Code: 14.10.2006 15:32:59
Interface: 14.02.2006 22:53:09
Längen (Baustein / Code / Daten): 00000 00000 00000

Objekteigenschaften:
S7_language 7(1) Deutsch (Deutschland) 28.06.2007 11:18:17

Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	COM_RST	BOOL	FALSE	[kompletter Neustart] Parameter in [] NICHT für unseren G ebrauch
+0.1	MAN_ON	BOOL	TRUE	[manueller Wert AN] Parameter mit einem * sind für unsere n Gebrauch
+0.2	PVPER_ON	BOOL	FALSE	Istwert Peripherie aktiv]
+0.3	P_SEL	BOOL	TRUE	*P-Anteil einschalten
+0.4	I_SEL	BOOL	TRUE	*I-Anteil einschalten
+0.5	INT_HOLD	BOOL	FALSE	[Integralwert einfrieren]
+0.6	I_ITL_ON	BOOL	FALSE	[Initialisierung des integralen Anteils]
+0.7	D_SEL	BOOL	FALSE	*D-Anteil einschalten
+2.0	CYCLE	TIME	T#1S	*Abtastzeit
+6.0	W	REAL	0.000000e+000	*interner Sollwert (Name bei Siemens: SP_INT)
+10.0	X	REAL	0.000000e+000	*Istwert Eingang (Name bei Siemens: PV_IN)
+14.0	PV_PER	WORD	W#16#0	[Istwert Peripherie]
+16.0	Y_hand	REAL	0.000000e+000	*Hand(Manual-)wert (Name bei Siemens: MAN)
+20.0	Kp	REAL	2.000000e+000	*Proportional Wert (Verstärkung) (Name bei Siemens: Gain)
+24.0	TI	TIME	T#20S	*Integrationszeit
+28.0	TD	TIME	T#10S	*Differenzierzeit
+32.0	TM_LAG	TIME	T#500MS	[Verzögerungszeit des D-Anteils]
+36.0	DEADBAND	REAL	0.000000e+000	*Totzonenbreite (Name bei Siemens: DEAD_W)
+40.0	Y_max	REAL	1.000000e+002	*Stellwert obere Begrenzung (Name bei Siemens: LMN_HLM)
+44.0	Y_min	REAL	0.000000e+000	*Stellwert untere Begrenzung (Name bei Siemens: LMN_LLM)
+48.0	PV_FAC	REAL	1.000000e+000	[Istwertfaktor]
+52.0	PV_OFF	REAL	0.000000e+000	[Istwertoffset]
+56.0	LMN_FAC	REAL	1.000000e+000	[Stellwertfaktor]
+60.0	LMN_OFF	REAL	0.000000e+000	[Stellwertoffset]
+64.0	I_ITLVAL	REAL	0.000000e+000	[Initialisierungswert für I-Anteil]
+68.0	DISV	REAL	0.000000e+000	[Störgröße]
+72.0	Y	REAL	0.000000e+000	[Stellwert (Name bei Siemens: LMN)]
+76.0	LMN_PER	WORD	W#16#0	[Stellwert Peripherie]
+78.0	QLMN_HLM	BOOL	FALSE	[Obere Begrenzung des Stellwertes angesprochen]
+78.1	QLMN_LLM	BOOL	FALSE	[Untere Begrenzung des Stellwertes angesprochen]
+80.0	LMN_P	REAL	0.000000e+000	[P-Anteil]
+84.0	LMN_I	REAL	0.000000e+000	[I-Anteil]
+88.0	LMN_D	REAL	0.000000e+000	[D-Anteil]
+92.0	PV	REAL	0.000000e+000	[Istwert]
+96.0	ER	REAL	0.000000e+000	[Regeldifferenz]
+100.0	sInvAlt	REAL	0.000000e+000	
+104.0	sIanteilAlt	REAL	0.000000e+000	
+108.0	sRestInt	REAL	0.000000e+000	
+112.0	sRestDif	REAL	0.000000e+000	
+116.0	sRueck	REAL	0.000000e+000	
+120.0	sLmn	REAL	0.000000e+000	
+124.0	sbArwHLmOn	BOOL	FALSE	
+124.1	sbArwLLmOn	BOOL	FALSE	
+124.2	sblLimOn	BOOL	TRUE	
+126.0	reserve_dword_126	REAL	0.000000e+000	ENDE des SIEMENS INSTANZE DB PARAMETER
+130.0	CMD_OFF	BOOL	FALSE	*Kommando AUS
+130.1	CMD_MANUAL	BOOL	FALSE	*Kommando MANUAL (HAND)
+130.2	CMD_AUTO	BOOL	FALSE	*Kommando AUTOMATIK
+130.3	CMD_W_ext	BOOL	FALSE	*Kommando W_ext (externer Sollwert)
+130.4	CMD_W_int	BOOL	FALSE	*Kommando W_int (interner Sollwert)
+130.5	reserve_bit_5	BOOL	FALSE	
+130.6	reserve_bit_6	BOOL	FALSE	
+130.7	reserve_bit_7	BOOL	FALSE	
+131.0	STAT_OFF	BOOL	FALSE	[Status AUS]
+131.1	STAT_MANUAL	BOOL	FALSE	[Status Manual (Hand)]
+131.2	STAT_AUTO	BOOL	FALSE	[Status Automatik]
+131.3	STAT_W_EXT_ON	BOOL	FALSE	*wähle W_ext oder W_int
+131.4	STAT_TRACKING_ON	BOOL	FALSE	*Nachführbetrieb AN/AUS für Y_nf; gesetzt von der S7
+131.5	STAT_REVERSE	BOOL	FALSE	*Reversebetrieb
+131.6	STAT_SPLITRANGE	BOOL	FALSE	*Splitrangebetrieb
+131.7	STAT_HITCHLESS	BOOL	FALSE	*Stoßfreie Umschaltung von MANUAL zu AUTO und zurück
+132.0	W_ext	REAL	0.000000e+000	*externer Sollwert für W von der PLC
+136.0	W_int	REAL	0.000000e+000	*interner Sollwert für W vom PC

SIMATIC 7_41dt_engl\Regler\...\UDT6 - <offline> 09.08.2007 11:29:49

Adresse	Name	Typ	Anfangswert	Kommentar
+140.0	Y_off	REAL	0.000000e+000	*zum Schutz der Maschine nach dem Ausschalten
+144.0	Y_nf	REAL	0.000000e+000	*Nachführwert; z. B. bei der der Rohrreinigung
+148.0	Scale_min	REAL	0.000000e+000	*Skalierungswert Minimun für X und W
+152.0	Scale_max	REAL	0.000000e+000	*Skalierungswert Maximum für X und W
+156.0	Y_result	REAL	0.000000e+000	[Y_result: Wert für den Ausgang; letzendliche Ausgabestellgröße]
+160.0	Splitrange_I	REAL	0.000000e+000	[für Splitrangewert -100% - 0 %]
+164.0	Splitrange_II	REAL	0.000000e+000	[für Splitrangewert 0 - 100 %]
+168.0	HH_LIMIT	REAL	0.000000e+000	
+172.0	H_LIMIT	REAL	0.000000e+000	
+176.0	L_LIMIT	REAL	0.000000e+000	
+180.0	LL_LIMIT	REAL	0.000000e+000	
+184.0	ALM_LL	BOOL	FALSE	
+184.1	ALM_L	BOOL	FALSE	
+184.2	ALM_H	BOOL	FALSE	
+184.3	ALM_HH	BOOL	FALSE	
+184.4	ALM_LL_WinCC	BOOL	FALSE	synchronisierbit WinCC Quittierung
+184.5	ALM_L_WinCC	BOOL	FALSE	synchronisierbit WinCC Quittierung
+184.6	ALM_H_WinCC	BOOL	FALSE	synchronisierbit WinCC Quittierung
+184.7	ALM_HH_WnCC	BOOL	FALSE	synchronisierbit WinCC Quittierung
+186.0	W_NORM	REAL	0.000000e+000	0.0-100.0%
+190.0	X_NORM	REAL	0.000000e+000	0.0-100.0%
+194.0	Y_NORM	REAL	0.000000e+000	0.0-100.0%
+198.0	SPARE_BY	BYTE	B#16#0	
+199.0	Last_AUTO	BOOL	FALSE	[zum Speichern des letzens Status von STAT_AUTO]
=200.0		END_STRUCT		

Auszug Regler Datenbaustein

SIMATIC 7_41dt_engl\Regler\...\DB6 - <offline> 09.08.2007 11:28:52

DB6 - <offline> - Deklarationssicht

"PID Interface" for all regulatorparameter
Globaler Datenbaustein DB 6
Name: Familie:
Autor: tp Version: 1.0
Bausteinversion: 2
Zeitstempel Code: 14.02.2006 22:54:08
Interface: 14.02.2006 22:54:08
Längen (Baustein / Code / Daten): 41424 20020 00000

Objekteigenschaften:
S7_language 7(1) Deutsch (Deutschland) 28.06.2007 11:18:17

Baustein: DB6

Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	regulator_total	INT	64	*default 64 regulators (Vorgabe 64 Regler)
+2.0	temp_reg_counter	INT	0	[temp. Zähler der bearbeiteten Regler]
+4.0	temp_call_counter	INT	0	[temp. Zähler der Reglerаufrufe]
+6.0	reserve_word_6	INT	0	
+8.0	reserve_dword_8	REAL	0.000000e+000	
+12.0	reserve_dword_12	REAL	0.000000e+000	
+16.0	reserve_dword_16	REAL	0.000000e+000	
+20.0	PID_000	"Regulator UDT for DB6"		
+220.0	PID_001	"Regulator UDT for DB6"		
+420.0	PID_002	"Regulator UDT for DB6"		
+620.0	PID_003	"Regulator UDT for DB6"		
+820.0	PID_004	"Regulator UDT for DB6"		

Auszug Analog Input Datenbaustein

SIMATIC 7_41dt_engl\AnalogS5_S7_Fix\...\DB30 - <offline> 09.08.2007 11:21:59

DB30 - <offline> - Deklarationssicht

"Analog Werte" Ablage Fertigwerte
Globaler Datenbaustein DB 30
Name: ANA_IN Familie: Analog
Autor: Daatec Version: 1.1
Bausteinversion: 2
Zeitstempel Code: 17.11.2006 12:44:41
Interface: 17.11.2006 12:44:41
Längen (Baustein / Code / Daten): 05592 00768 00000

Objekteigenschaften:
S7_language 7(1) German (Germany) 28.06.2007 11:18:17

Baustein: DB30 Analogwerte_Eingang
Funktion..: Speicherung Fertigwerte analog copyrihgt.: © 2004 Daatec GmbH Datum.....: 01.01.2004 Info.....: Begin=DW 000 - Offset nächster Wert=6

Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	PE11011	"UDT AE"		KG1 pressure operating air
+6.0	PE11012	"UDT AE"		KG1 pressure filtrate
+12.0	PE11018	"UDT AE"		KG1 pressure unfiltrate
+18.0	LE11113	"UDT AE"		KG1 level Dosimat
+24.0	TE11502	"UDT AE"		CIP 1 temperature supply before heat echanger
+30.0	TE11503	"UDT AE"		CIP 1 temperature supply after heat exchanger
+36.0	PE12017	"UDT AE"		line 1 pressure pre and finalrinse tank bottom
+42.0	PE12018	"UDT AE"		line 1 pressure pre and finalrinse tank top
+48.0	LE12213	"UDT AE"		line 1 level pre and finalrinse tank
+54.0	PE12218	"UDT AE"		line 1 pressure BT2
+60.0	spare1	"UDT AE"		

UDT Block für Analoge Inputs

SIMATIC 7_41dt_engl\AnalogS5_S7_Fix\...\UDT30 - <offline> 09.08.2007 11:23:46

UDT30 - <offline>

"UDT AE"
Name: Familie: Analog
Autor: Daatec Version: 1.1
Bausteinversion: 2
Zeitstempel Code: 04.12.2006 15:09:35
Interface: 04.12.2006 15:09:35
Längen (Baustein / Code / Daten): 00000 00000 00000
UNLINKED

Objekteigenschaften:
S7_language 7(1) Deutsch (Deutschland) 28.06.2007 11:18:17

Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	X	REAL	0.000000e+000	Ablage berechneter Wert
+4.0	Broken	BOOL	FALSE	Drahtbruch
+4.1	Status_Simulation_ein	BOOL	FALSE	Status Simulation ist ein
+4.2	b4_2	BOOL	FALSE	
+4.3	b4_3	BOOL	FALSE	
+4.4	b4_4	BOOL	FALSE	
+4.5	b4_5	BOOL	FALSE	
+4.6	b4_6	BOOL	FALSE	
+4.7	Overflow	BOOL	FALSE	Überlauf
+5.0	b5_0	BOOL	FALSE	
+5.1	b5_1	BOOL	FALSE	
+5.2	b5_2	BOOL	FALSE	
+5.3	b5_3	BOOL	FALSE	
+5.4	b5_4	BOOL	FALSE	
+5.5	b5_5	BOOL	FALSE	
+5.6	b5_6	BOOL	FALSE	
+5.7	b5_7	BOOL	FALSE	
=6.0		END_STRUCT		

Datenbaustein Digital Input

SIMATIC 7_41dt_engl\Binaer\...\DB10 - <offline> 09.08.2007 11:26:55

DB10 - <offline> - Deklarationssicht

"DI_DB"
Globaler Datenbaustein DB 10
Name: Familie:
Autor: Version: 0.1
Bausteinversion: 2
Zeitstempel Code: 28.07.2005 13:24:58
Interface: 28.07.2005 13:24:45
Längen (Baustein / Code / Daten): 02150 02048 00000

Objekteigenschaften:
S7_language 7(1) Deutsch (Deutschland) 28.06.2007 11:18:17

Baustein: DB10

Adresse	Name	Typ	Anfangswert	Kommentar
0.0		STRUCT		
+0.0	DB_DI	ARRAY[0..2047,0..7]		
*0.1		BOOL		
=2048.0		END_STRUCT		

12 Erklärung zur Diplomarbeit

Name: Sperling

Stand: 03/2007

Vorname: Andre

Matr.-Nr.: 1156597

Studiengang : Angewandte Automatisierungstechnik

An den Prüfungsausschuss

der Universität Lüneburg

Fakultät III

Bereich Automatisierungstechnik Volgershall 1

21339 Lüneburg

Erklärung zur Diplomarbeit

Ich versichere, dass ich diese Diplomarbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Lüneburg, den 20.08.2007

_____

Unterschrift