Softwareplattform Spezifikation

www.copadata.com
sales@copadata.com

zenon
by COPA-DATA

Chronologie

Letztes Update:	Features zenon 15 Release, 2025
Update	Funktionalitäten für Energy, zenon 15

Inhaltsverzeichnis

Einleitung

Dieses Dokument beschreibt die Anforderungen an eine Softwareplattform für den Einsatz in der Automatisierung, als HMI- oder SCADA-System für die Fertigungs- und die Energiebranche. Für diesen Zweck beziehen sich die Spezifikationen auf allgemeine Merkmale der Systemarchitektur und der Systemdesign-Philosophie sowie auf spezifische funktionale Details. Außerdem werden die Systemanforderungen sowohl aus der Perspektive der Systementwicklung als auch des Systembetriebs beschrieben. Spezielles Augenmerk liegt dabei auf der Systemintegration (vertikale Integration, Kopplung). Das Dokument wird ergänzt durch Spezifikationen der für die Automatisierung benötigten Sicherheitsmerkmale.

Einschränkungen bezüglich der Nutzung des Inhalts dieses Dokuments:

Dieses Dokument stellt keinen Anspruch auf Vollständigkeit. Es stellt nicht den Anspruch, alle benötigten technologischen Merkmale im Detail zu erklären, da sich die Anforderungen und Lösungen (Produkt und Produktkomponenten) dynamisch weiterentwickeln. Stattdessen repräsentiert es die generische Spezifikation einer Softwareplattform für den Einsatz im Bereich der Automatisierung, vor allem im Moment der Erstellung des Dokuments. Pflichtfunktionen und architektonische Notwendigkeiten werden angeführt, um die Anwendungsanforderungen im Bereich der Prozess Automatisierung umfassend abzudecken. Daraus folgt, dass beim vollständigen oder teilweisen Wiederverwenden der Informationen in diesem Dokument große Sorgfalt in Bezug auf die jeweilige Zielanwendung nötig ist. Es soll klargestellt werden, dass die Bearbeiter dieser Informationen für die technische Validität ihrer weiteren Verwendung nicht verantwortlich sind.

Definitionen

Da dieses Dokument die Spezifikation einer Softwareplattform für den Einsatz in Automatisierungsstrukturen (als HMI oder SCADA-System) repräsentiert, werden einige Definitionen gebraucht, die für alle folgenden Abschnitte gelten.

Terminologie Anforderungsdefinition

- Aussagen mit **„Muss“** markieren Pflicht-Anforderungen. Diese müssen befolgt werden und benötigen eine Verifizierung.

- Aussagen mit **„Wird“** stehen für geplante Anwendungsfälle. Darum beziehen sie sich auch auf Pflicht-Anforderungen. Sie identifizieren Aspekte von zukünftigen Systemen, die möglicherweise nicht leicht zu verifizieren sind (z.B. Systemerweiterbarkeit, Versionskompatibilität, Konnektivität).

- Aussagen mit **„Sollte“** markieren Empfehlungen. Die darin beschriebenen Systemeigenschaften werden dringend empfohlen.

Geplantes Systemdesign

Dieses Dokument bietet einen Vorschlag für eine Aufteilung der Gesamtlösung in verschiedene funktionale Komponenten. Grundsätzlich besteht die Softwareplattformaus einem Entwicklungssystem (zenon Engineering Studio) und einem Runtime-System (zenon Service Engine). Darüber hinaus kann die Runtime entweder Server- oder Client-Funktionalität übernehmen. Ein Server „leitet“ üblicherweise den Prozess und bietet verschiedene Mechanismen für die Prozesssteuerung, die Datenverwaltung und die Kommunikation mit Drittsystemen. Bei kritischen Infrastrukturen muss der Server auf ein redundantes System (Hot Standby) dupliziert werden. Hauptaufgabe des Clients ist es, dem Bediener Zugriff auf das Prozessleitsystem zu ermöglichen, und zwar mithilfe von HMI, z.B. einer visuellen Prozessanzeige.

Ausdrücke und Abkürzungen

ASCII	American Standard Code for Information Interchange
COM	Component Object Model
CSV	Comma Separated Values
ERP	Enterprise Resource Planning
HMI	Human Machine Interface
HTML	Hypertext Markup Language
IED	Intelligent Electronic Device
IKT	Informations- und Kommunikations-Technologie
KPI	Key Performance Indicator
LAN/WAN	Local Area Network / Wide Area Network
MIS	Manufacturing Information System
OLEDB / ODBC	OLE Database / Open Database Connectivity
OLE	Open Linking and Embedding
OPC (UA)	Open Platform Communication (Unified Architecture)
SPS	Speicherprogrammierbare Steuerung
RTU	Remote Terminal Unit
SCADA	Supervisory Control and Data Acquisition
SCL, SCD, ICD, CID	Verschiedene Dateiformate von IEC 61850 Konfigurationsdateien, die auf der Substation Configuration Language basieren.
SMTP	Simple Mail Transfer Protocol
SNMP	Simple Network Management Protocol
SOE	Sequence of Events
SVG	Scalable Vector Graphics
VBA	Visual Basic for Applications
VPN	Virtual Private Network
WPF	Windows® Presentation Foundation
WYSIWYG	What-You-See-Is-What-You-Get
XAML	Extensible Application Markup Language
XML	Extensible Markup Language

1. Allgemein

1.1. Betriebssystem-Umgebung

Die Softwareplattform (in der Anwendung als SCADA-System) und alle wichtigen Entwicklungs- und Runtime-Komponenten müssen unter folgenden Betriebssystemen und zugehörigen Netzwerkumgebungen funktionieren:

Betriebssystem Edition / Version	Entwicklung und Runtime	Nur Runtime
Windows® 10 (Pro, Enterprise) x64 ab 21H2	●	
Windows® 10 (IoT) Enterprise LTSC x64 ab Version 2021	●	
Windows® 11 (Pro, Enterprise) ab 21H2	●	
Windows® 11 (IoT) Enterprise LTSC ab Version 2024	●	
Windows® Server 2019 (ausgenommen Core Versionen) ab Version 2019	●	
Windows® Server 2022 (ausgenommen Core Versionen) ab Version 2022	●	
Windows® Server 2025 (ausgenommen Core Versionen) ab Version 2025	●	

Linux-Betriebssysteme:

Service Engine für Linux und Logic Service sind als 64-Bit-Docker-Image für amd64- und arm64-Plattformen verfügbar.

Achtung: Diese Komponenten erfordern eine funktionierende Docker-Installation sowie einen separaten Lizenzserver.

Info: Die Service Engine ist und wird nicht für 32 bit (= x86 und armhf) Linux Betriebssysteme auf jeder Prozessorarchitektur verfügbar sein.

Die Hauptkomponenten des SCADA-Systems müssen als native 64-Bit-Anwendungen ausführbar sein, um die Performance auf dem jeweiligen Betriebssystem zu optimieren.

Das SCADA-System muss auch in virtualisierten Umgebungen lauffähig sein. Es muss sowohl einen nativen Hypervisor als auch einen gehosteten Hypervisor unterstützen.

Im SCADA System kann die Qualität der dargestellten Grafik auf die Ressourcen des Systems abgestimmt werden. Die Einstellung erfolgt Projekt spezifisch. Diese Einstellungen wirken sich nur auf die Darstellung / Belastung in der Runtime aus.

Folgende Einstellungen sind möglich:

Windows Basis: Grundlegende Grafikeigenschaften. Für ressourcenschwache Hardware empfohlen.

DirectX Software: Grafikberechnungen werden von der CPU erledigt. Es können, abhängig von den darzustellenden Grafiken, mehrere CPUs verwendet werden. Die Verwendung von DirectX Software kann eine hohe Auslastung der CPU erzeugen.

DirectX Hardware: Ein Teil der Grafikberechnungen wird auf die Grafikkarte ausgelagert, wodurch die Performance gesteigert werden kann. Wird diese Einstellung vom verwendeten System nicht unterstützt, muss das System automatisch auf DirectX Software zurückschalten. Prinzipiell ist DirectX Hardware zu bevorzugen und DirectX Software nur zu verwenden, wenn unbedingt benötigt.

1.2. Systemanforderungen auf höchster Ebene

Ein voll ausgerüstetes und umfassendes Visualisierungssystem mit einem integrierten Entwicklungssystem wird benötigt. Das System muss das Konzept der Dezentralisierung unterstützen und somit die freie Zuweisung von Entwicklungs- und Runtime-Systeminstanzen (Anwendungsserver, Bedienerstationen) innerhalb eines gegebenen IT-Netzwerks (LAN, WAN) erlauben. Neben der Prozessvisualisierung für Steuerungs- und Überwachungszwecke werden folgende Funktionen benötigt:

- Protokollierung von Prozessdaten auf Drucker und Monitor
- Spontane, zyklische und Ereignis gesteuerte Archivierung von Daten in Dateien oder Datenbanken
- Alarmierung im Störungsfall
- Verarbeitungsfunktion für binäre und analoge Daten

Außerdem muss das System erweiterte Funktionen bieten, wie z.B. Datenaggregation und Reporting, Ereignisprotokollierung und Anzeige von Prozessdaten, statistische Leistung und Systeminformation.

Das System muss die Möglichkeit bieten, als Prozessleitsystem zu funktionieren. Darum muss eine vollständige IEC 61131-3 Programmierschnittstelle in das System integriert sein. Die Programmierschnittstelle muss alle fünf standardisierten IEC-Programmiersprachen unterstützen

- Sequential Function Chart (SFC)
- Function Block Diagram (FBD)
- Ladder Diagram (LD)
- Structured Text (ST)
- Instruction List (IL)

Das zugehörige Runtime-System muss als Soft-SPS und auch als Hard-SPS verfügbar sein. Die SPS-Komponente muss unter denselben Betriebssystemen lauffähig sein wie alle anderen SCADA-Komponenten (siehe Abschnitt 1.1). Das System muss die modulare Anwendung dieser Steuerkomponente in einer integrierten sowie einer dezentralen Umgebung ermöglichen (Integriert: SCADA und Soft-SPS in derselben Betriebssystem-Umgebung; Dezentralisiert: Soft-SPS läuft auf einem separaten physischen Gerät). In beiden Fällen muss ein voll integrierter Datenfluss innerhalb des Systems garantiert werden. Darum müssen Entwicklungsmethoden für jede gewählte Systemtopologie auf dieselbe Art und Weise angewendet werden können.

Das Prozessleitsystem soll modular aufgebaut sein. Es muss auf jeder Lösungs- und Anwendungsebene erweiterbar sein. Das System muss den Betrieb von einem oder mehreren Visualisierungsbildschirmen (HMI) erlauben. Die Erweiterung auf eine Mehrstationen-Konfiguration sowie die Anbindung eines Arbeitsplatzes über eine Onlineverbindung (Intranet oder Internet mit angemessener Sicherheit) muss möglich sein.

Wenn ein System aus Komponenten von verschiedenen Teillösungen und verschiedenen hierarchischen Ebenen besteht (Aggregierung verschiedener Teilprojekte), so müssen dieselben Bilder überall anwendbar sein. Generell muss beachtet werden, dass Datenelemente konsistent gehalten werden, basierend auf einzelnen Dateneinträgen (Vermeidung von mehrfachen, redundant definierten Datenpunkten).

Es muss für das Bereitschaftspersonal möglich sein, das System von einem externen Zugriffspunkt zu bedienen (z.B. Fernzugriff von privatem Wohnsitz). Über das Remote-Terminal muss der Zugriff auf dieselben Bilder und Funktionen wie auf dem Hauptsystem möglich sein. Ein Systemzugriff muss über gewöhnliche Methoden der WAN- oder Internetkommunikation möglich sein, z.B. über analoges Modem, Breitbandkabel, ISDN, oder auch mobile Kommunikationsnetzwerke. Ein Fernsteuerungs-Terminal muss über eine native System-Clientstation oder über eine webbasierte Clientvisualisierung verfügbar sein, die mit einem Standard-Internetbrowser verwendbar ist (siehe Abschnitt 12.2). Verbindungen für den Remote Zugriff sollten zumindest eine Authentifizierung und eine Verschlüsselung der Daten unterstützen.

Das verwendete Produkt muss auf dem Markt frei verfügbar sein. Die grundlegende Systemeinrichtung (Systementwicklung) muss ohne umfassende IT- und Programmierkenntnisse machbar sein. Dies muss für eine Menge von klar definierten SCADA- und Automatisierungsaufgaben realistisch ausführbar sein, wie z.B. Design von Visualisierung und Bildnavigation, Rezeptverwaltung, Schaltfolgen, Alarmierung, Datenarchivierung und Variablenauszeichnung, Befehlsgabe und topologische Einfärbung. Auf dieselbe Art und Weise muss die Herstellung von Datenverbindungen zu Geräten oder Stationen (IEDs) von Drittherstellern durch verfügbare Kommunikationstreiber umfassend unterstützt werden (siehe auch Abschnitt 5). Eine Erweiterung einer bestehenden Lösung muss mithilfe derselben Entwicklungsmechanismen möglich sein, die für die anfängliche Entwicklung genutzt wurden (Runtime- und Entwicklungssystemkompatibilität über verschiedene Versionen hinweg).

Um auf dem neuesten Stand der Technologie zu bleiben, müssen neue oder verbesserte Funktionen, die aus einer kontinuierlichen Produktentwicklung (neue Versionen der Softwareplattform) resultieren, in Projektrevisionen anwendbar sein. Darum muss die Kompatibilität von Systemkomponenten und -versionen für zumindest fünf Jahre garantiert werden bzw. für zehn Jahre im Falle eines Service Level Agreement. Das System muss dedizierte Mechanismen für die Entwicklungsverwaltung anbieten, wie z.B. Versionierung, Projektsicherung, Änderungsverfolgung und selektiver Projektzugriff (Schutz von kritischen Modulen/Teilen).

1.3. Integrierte Entwicklung

Ein integriertes Softwarepaket für die Erstellung der im Vorabschnitt erwähnten Funktionen wird benötigt. Die Bedienung der Software darf keine vertiefenden Programmierkenntnisse voraussetzen. Das gesamte System muss Unicode basierend sein, die Verwendung beliebiger Zeichensätze muss ohne zusätzlichen Aufwand möglich sein. Es muss möglich sein, lediglich mithilfe von Konfigurations-Wizards, Vorlagen und Parametereinstellungen ein Projekt zu erstellen. Für die erweiterte Anpassung des Systems muss es möglich sein, Erweiterungen für die Runtime-Funktionalität zu entwickeln, die auf Standard-Skriptsprachen basieren, wie z.B. .NET oder C#. Gleichzeitig muss eine Integration von WPF / .NET Steuerelementen von Drittherstellern sowie aus eigener Entwicklung möglich sein.

Das Projektierungstool muss die parallele Arbeit von mehreren Personen an einem Projekt unterstützen. Das System muss Mechanismen anbieten, die vermeiden, dass mehr als eine Person zur selben Zeit an demselben Teil des Projekts arbeitet und somit einen inkonsistenten Projektzustand verursacht. Darum müssen Projektteile, die gerade bearbeitet werden, unzugänglich (gesperrt) für andere Entwickler sein. Das Projektierungstool muss Informationen über den Ort (Workstation-ID) und den Benutzer anzeigen können, der gerade exklusiv an einem bestimmten Projektteil arbeitet.

Um die Projekterstellung zu vereinfachen, muss es möglich sein, wiederkehrende Projektteile automatisch zu generieren. Dafür müssen die betreffenden Mechanismen der Entwicklungsumgebung über Skriptsprachen wie .NET oder C# steuerbar sein. Es muss möglich sein, den programmierten Code ausgelöst durch bestimmte Ereignisse innerhalb des Entwicklungssystems auszuführen, sodass Projektdefinitionen automatisch generiert und entsprechend angeordnet werden. Gemäß den Anforderungen bezüglich Kompatibilität und Wiederverwendbarkeit muss der Skriptcode in Folgeprojekten wiederverwendbar sein (Kompatibilität von Schnittstellen).

Die Visualisierungsbilder müssen unabhängig von der endgültigen Bildschirmauflösung des Zielsystems (Bildschirmabmessungen) entwickelt werden. Als Option muss es möglich sein, die Bildabmessungen des Zielsystems voreinzustellen.

Das System muss die Möglichkeit bieten, Projekte für mehrere Bedienerstationen zu entwerfen, die auf einzelnen Computersystemen laufen (Server und Remote-Clients).

Das System muss die Möglichkeit einer schablonenbasierten Projektierung anbieten. Global definierte Schablonen müssen für den Einsatz über mehrere Projekte hinweg verfügbar sein (siehe Abschnitt 2).

Schriften, Sprachtabellen für die Online-Sprachumschaltung, Bitmaps, Hilfedateien, Multimediadateien, Textdateien sowie benutzerdefinierte Dateien müssen sowohl zentral als auch global in einem Projekt verfügbar sein.

Zur Standard-Funktionalität des Entwicklungstools muss eine Import-/Exportschnittstelle gehören, die es erlaubt, Projektdefinitionsdaten wie z.B. Bilder, Variablen, Archivserver-Definitionen etc. im XML-Format für die externe Bearbeitung zu exportieren und die Daten wieder zu importieren. Genauso muss es möglich sein, extern erstellte Daten zu importieren, z.B. aus CAD/CAE-Programmen. Für Variablen muss es zusätzlich zu XML eine Funktion zum Import/Export im CSV-Format geben.

Das System muss Online-Projektierung sowie eine Übertragung von Änderungen per Mausklick während der Runtime unterstützen. Die Systemprojektierung und -entwicklung muss auf jedem Arbeitsplatz möglich sein, der die oben genannten Betriebssystem-Anforderungen erfüllt (siehe Abschnitt 1.1). Projektänderungen müssen selektiv an jene Station übertragbar sein, auf der die betroffene Systemkomponente liegt (ein ausgewählter Server oder Client). Die Dateien für das Runtime-System müssen auf das Zielsystem über eine serielle oder eine TCP/IP-Verbindung übertragbar sein. Idealerweise sollte das Entwicklungssystem einen integrierten Transportmechanismus bieten, der alle notwendigen Daten auf das Zielsystem überträgt.

Ein gleichzeitiger Betrieb des Entwicklungssystems und der Runtime auf einem PC muss möglich sein.

Das System muss eine konsistente Online-Sprachumschaltung sowie Schriftumschaltung während der Runtime unterstützen. Alle Texte (inklusive Dialoge), die für den Benutzer sichtbar sind, müssen während der System-Runtime austauschbar sein.

Eine Ansicht mit Baum- sowie Listenstruktur, wie sie in gängigen Dateiverwaltungstools umgesetzt ist (Elemente ein- und ausklappen) wird für eine schnelle und einfache Navigation bei der Projektierung benötigt. Projekte müssen über Copy & Paste sowie Drag & Drop angelegt und gewartet werden können. Es muss möglich sein, individuelle Objekte über ein Eigenschaftenfenster zu bearbeiten, das Mehrfachauswahl sowie Mehrfachänderungen erlaubt.

Verweise im System (z.B. beim Variableneinsatz in Bildern) müssen im Standardpaket über ein kontextsensitives Menü rück verfolgbar sein. Es muss die Möglichkeit geben, eine Querverweisliste für die Projektdokumentation auszudrucken. Außerdem muss es möglich sein, nach unverknüpften Variablen und Funktionen zu suchen.

Es muss in jeder Phase der Projektentwicklung möglich sein, eine Sicherung eines Entwicklungsprojekts zu speichern und zu erzeugen, inklusive aller Unterverzeichnisse und benötigten Ressourcen, wie Grafiken, Variablendefinitionen etc. Eine Projektsicherungsdatei darf grundsätzlich nicht größer als 20 MB sein (außer bei umfangreichem Einsatz von Mediendateien), um einen Versand per E-Mail zu ermöglichen.

Das System muss die Möglichkeit eines Rollbacks auf eine ältere Projektsicherung bieten und einen Versionierungsmechanismus anbieten, der automatisch eine neue Unterversionsnummer vergibt, wenn eine Sicherung angelegt wird.

Das Projektierungstool muss eine integrierte Änderungshistorie anbieten, die eine konsistente Aufzeichnung von Bearbeiter (Benutzeridentität), Datum, Zeit und Inhalt einer Änderung in einem Projekt ermöglicht. Der Genauigkeitsgrad der aufgezeichneten Daten muss konfigurierbar sein.

Das System muss die Möglichkeit bieten, eine Projektdokumentation automatisch zu erstellen. Das automatisch erzeugte Dokument muss Informationen über die Entwicklung der Projektstruktur, Module, Objekte und Einstellungen sowie der Verweise enthalten. Die automatisch erzeugte Projektdokumentation muss in einem offenen Dokumentformat erstellt werden, das in übergeordnete Dokumente integriert werden kann. Zudem muss es den Einsatz von Navigationslinks (e.g. HTML) erlauben.

1.4. Netzwerk, topologisches Design und Datenaustausch

Um eine Integration von Informationen aus verschiedenen Datenquellen aus dem Bereich der Prozess Automatisierung mit vertretbarem Aufwand zu ermöglichen, muss das System spezifische Netzwerk- und Verbindungsfunktionen enthalten. Das System muss in einer frei definierbaren Anordnung von Komponenten (z.B. baumförmige Topologie, bestehend aus Servern, Clients und Systemen von Drittanbietern) verwendbar sein:

- Automatische Verwaltung von Datenfluss und Kommunikation über die Originalkomponenten des Systems (Server, Clients): Es darf kein Entwicklungsaufwand für den Austausch von einzelnen Variablen, Datasets oder Datenstrukturen notwendig sein. Der Datenaustausch muss automatisch definiert und eingerichtet werden, abhängig von der Struktur des Anwendungsprojekts. Mechanismen wie Sichtbarkeitsbereiche für Variablen und Teilprojekte (modulares Projektdesign) müssen enthalten sein, um eine beschränkten Datenzugriff umzusetzen.

- Trennung von logischer Projektstruktur und physischer Zieltopologie: Die im Entwicklungsprojekt definierte Systemfunktionalität darf nicht an eine vordefinierte Anordnung von Runtime-Komponenten (Hardware sowie Software) gebunden sein. Stattdessen muss das System die beliebige Anordnung von Host-Systemen ermöglichen, gemäß der in Abschnitt 1.1 angeführten Betriebssystem-Anforderungen. Der notwendige Datenaustausch zwischen den physischen Komponenten (Host-Systeme, aus denen die physische Topologie besteht) muss wie oben erwähnt automatisch und transparent ablaufen (also im Grunde unsichtbar für den

Anwendungsentwickler). Idealerweise sollte die Parametrisierung der Adressen der Host-Stationen (IP-Adressen) innerhalb des gegebenen IT-Netzwerkes ausreichen, um die jeweiligen Systemkomponenten anzuwenden (Auszeichnung von Servern und Clients). Im Weiteren muss das Kommunikationsprotokoll, welches für die Verbindung zwischen Servern, Standby Servern sowie Clients verwendet wird, mittels TLS 1.3 Verschlüsselung geschützt werden können.

- Ausreichende Unterstützung von Kommunikationsrichtlinien: Das System muss einsatzbereite Treiber zur Verfügung stellen, um den Datenaustausch mit Systemen von Drittanbietern zu ermöglichen die üblicherweise in der Automatisierung zum Einsatz kommen. Der Einsatz dieser Treiber muss auf konsistente Weise in den allgemeinen Datenfluss des Systems integriert sein. Bei der Einrichtung der jeweiligen Kommunikationsschnittstellen muss der Benutzer durch die Entwicklungsumgebung geleitet werden (z.B. durch Konfigurations-Wizards). Wenn eine Kommunikationsrichtlinie dies erlaubt, sollte dieser Einrichtungsvorgang lediglich aus der Parametrisierung der notwendigen Attribute bestehen (z.B. Knotennummern, Zeitintervalle, Protokollbesonderheiten etc.).

1.5. Systemredundanz

Für hochverfügbare Anlagen werden redundante Server benötigt. Im Falle eines Serverausfalls muss sofort ein redundanter Server aktiv werden und die Funktion der ausgefallenen Komponenten übernehmen. Der Systembetrieb muss konsistent weitergeführt werden. Das betrifft unter anderem die Weiterführung von Datenaufzeichnung und -archivierung sowie von Steuerungs-, Überwachungs- und Kommunikationsaufgaben. Das Backupsystem muss folglich zu jedem Zeitpunkt eine exakt gespiegelte Kopie des Primärsystems darstellen.

Die Einrichtung des redundanten Systems muss selektiv erfolgen, indem jene Serverstation definiert wird, die gesichert werden soll. Es dürfen keine besonderen Netzwerk- oder Programmierkenntnisse notwendig sein, um eine redundante Systemtopologie realisieren zu können. Der Aufwand für die Umsetzung eines redundanten Systems muss so gering wie möglich gehalten werden. Idealerweise sollte dazu eine Parametrisierung des Fallback-Servers (Stationsadresse) sowie allgemeiner Verhaltensparameter ausreichen.

Das System muss dazu fähig sein, sowohl eine Software- wie auch eine Hardwareredundanz anzuzeigen. Softwareredundanz impliziert den nahtlosen Ersatz einer SCADA-Systemkomponente (Server). Hardwareredundanz impliziert die konsistente Verwaltung einer redundanten Einrichtung von aufeinanderfolgenden Steuergeräten (z.B. doppelte SPS-Hardware) für den Fall eines Komponentenausfalls.

Der Standby Server muss den Ausfall eines Servers automatisch erkennen und sofort zum Server werden. Alle angeschlossenen Clients müssen außerdem den Ausfall des Servers erkennen und sofort auf den neuen Server umschalten. Während der Zeit eines Ausfalls bzw. seiner Erkennung darf kein Datenverlust erfolgen.

Das System muss eine kreisredundante Struktur herstellen können, die wie folgt aufgebaut ist: Computer A ist der Server für Projekt A und Standby Server für Projekt B. Computer B ist der Server für Projekt B und Standby Server für Projekt C. Computer C ist der Server für Projekt C und Standby Server für Projekt A. Somit wird der Kreis geschlossen. Wenn ein Computer ausfällt, muss das System mit den verbleibenden Stationen konsistent weiterlaufen können. Dieses Modell muss durch eine beliebige Anzahl an verbundenen Stationen erweiterbar sein.

Zudem muss die Aktivierung des Servers auf einer benutzerdefinierten Metrik basieren. Dies ermöglicht die Nominierung des optimalen Servers aufgrund vorgegebener Grenzbedingungen (z.B. Verbindungsqualität, verbleibende Rechen- und Speicherkapazität etc.), inklusive einer flexiblen Serverumschaltung.

Bei der Hardware-Redundanz in einem Bewerteten Netzwerk entscheiden Bewertungskriterien, welcher Rechner die Rolle des Prozessführenden Servers und welcher die Rolle des Standby Servers übernimmt.

Diese Bewertung ist frei konfigurierbar und kann mehrere unterschiedliche Kriterien beinhalten. Jedem Kriterium sind Bewertungspunkte - die sogenannte Gewichtung - zugeordnet. Die Summe der Gewichtungspunkte entscheidet dann über die jeweilige Serverrolle.

Bezüglich der Serveraktivierung muss sowohl dominantes als auch nichtdominantes Redundanzverhalten verfügbar sein.

1.6. Modularität und Erweiterbarkeit

Um auf wechselnde Anforderungen, technologische Neuerungen oder grundlegende Systemänderungen (z.B. Erweiterungen) zu reagieren, muss das System beliebige Änderungsszenarios unterstützen. Dies muss auf den Prinzipien einer modularen Systemarchitektur und strukturierten Systementwicklung basieren, unterstützt durch Funktionen für die Wiederverwendung von bereits entwickelten Projektteilen. Das System muss die folgenden Strukturierungsmöglichkeiten bieten:

- Modulare Anordnung von grundlegenden Systemkomponenten, Anbieten gekapselter Technologiemodule.
- Hohes Maß an Kompatibilität zwischen diesen Modulen, vor allem was neu erscheinende Releases betrifft (Weiterentwicklung des Grundsystems).
- Hierarchische Organisation von Projekten und Projektteilen (Haupt- und Unterprojekte)
- Hierarchische Organisation von Projektkomponenten (z.B. objektorientierte Repräsentation von Lösungselementen)
- Orientierung an wohldefinierten Schnittstellen und Datenformaten, auf Seite der Anwendung, z.B. COM (Component Object Model) sowie des Umfeldes, z.B. OPC UA, XML, HTML etc.
- Modulare Updatemöglichkeiten für aktive Runtime (siehe "Hot Reload" in Abschnitt 1.8)

1.7. Überwachung und Steuerung von dezentralisierten Anlagen

Das System muss die Erstellung einer dezentralisierten Kopplung von Client (Visualisierung) und Server (Anwendungsserver) ermöglichen. Die Methoden zur Dezentralisierung müssen es erlauben, große Infrastruktur-Installationen aufzuteilen und in vernünftig dimensionierten Teilprojekten zu verwalten. Darüber hinaus muss es möglich sein, Teilprojekte zusammenzufügen, um eine zentralen Leitstand zu schaffen, der Zugriff auf alle untergeordneten Projekte hat. Dazu müssen Möglichkeiten zur hierarchischen Projektanordnung geboten werden. Die Daten der untergeordneten Projekte (Bilder, Alarme, chronologische Daten) müssen über das Integrationsprojekt (übergreifendes Projekt) direkt erreichbar sein.

Der zentrale Computer muss gleichzeitig mehrere individuelle Projekte in der Gesamtumgebung starten können.

Die Alarmstatuszeile des Integrationsprojekts muss ebenso alle Alarme der untergeordneten Projekte anzeigen. Es muss möglich sein, beliebige Variablen untergeordneter Projekte in der Prozessdarstellung des Integrationsprojekts anzuzeigen. Dadurch sollte sichergestellt sein, dass der Bediener alle wichtigen Informationen aus untergeordneten Stationen auf einen Blick sehen kann.

Um ausreichende Flexibilität zu gewährleisten, muss eine Definition von mehrfachen Clients und Servern möglich sein, die miteinander vernetzt sind. Die logische Strukturierung und Integration von Projekten entsprechend der existierenden Überwachungs- und Steuerungsaufgaben sollte zu einer höheren Projektierungseffizienz beitragen. Die Dezentralisierungsfunktionen des Systems müssen das Anwenden von Projektkomponenten (Server, Clients) auf den gewünschten Netzwerkstandort (designierte Hoststation) erlauben. Dies muss letztlich die Erstellung des benötigten topologischen Schemas ermöglichen.

Es muss möglich sein, einzelne Projekte unabhängig vom Betriebszustand anderer Stationen selektiv zu starten oder zu pausieren (Runtime-Betrieb). Solange logische Abhängigkeiten nicht durch die projektierte Anwendungsfunktionalität eingeführt werden, dürfen sich Entwicklungsprojekte nicht gegenseitig beeinflussen.

In einer dezentralisierten Infrastruktur-Installation muss es möglich sein, Informationen in einer wohldefinierten und homogenen Art und Weise zu liefern, unabhängig von geografischen oder organisatorischen Grenzen (d.h. „Horizontale Transparenz“). Abhängig von der Verfügbarkeit der benötigten IT-Infrastruktur und Netzwerke (LAN, WAN) muss jeder Computer grundsätzlich in der Lage sein, auf das Runtime-System (HMI) der benachbarten Computer zuzugreifen. Damit wird sichergestellt, dass mit jedem individuellen Computer, der in die Gesamtanwendung integriert ist, die gesamte Anlage überwacht und gesteuert werden kann.

1.8. Systemupdates und Kompatibilität

Die weitreichende Anwendung von Systemkomponenten in einer vorwiegend verteilten Art und Weise führt zu gesteigerten Anforderungen, was die Versionskompatibilität des Systems betrifft (Systemkomponentenversionen). Diese Tatsache ist eng verknüpft mit dem modularen Systementwurf und steht im Einklang mit der Notwendigkeit einer „minimal-invasiven“ Wartung (z.B. kein komplettes Herunterfahren erlaubt, modulare Projektupdates, Hot Reload).

Darum muss das System den folgenden Anforderungen bezüglich Kompatibilität entsprechen:

- Projektkompatibilität (Engineering Kompatibilität) im Falle einer neuen Version der Entwicklungsumgebung („älteres“ Entwicklungsprojekt kann in einer „neueren“ Entwicklungsumgebung geöffnet und gewartet werden).

- Eine neue Entwicklungsumgebung kann Runtime-Definitionen für ältere Runtime-Versionen mittels dedizierter Compiler Einstellungen erstellen (Entwicklungssystem und Runtime-System, Abwärtskompatibilität)

- Eine neue Runtime-Version kann ältere Runtime-Dateiendefinitionen ausführen (Runtime-System, Abwärtskompatibilität)

- Eine neue Runtime-Version eines Netzwerk-Clients kann mit einer älteren Runtime eines Netzwerk-Servers interagieren

- Modulares Update von projektierter Funktionalität während der Runtime (selektiver Runtime-Projekt-Reload, Hot Reload)

- XML-Import/Export Aufwärts Formatkompatibilität über verschiedene Versionen

- Die Kompatibilität von Programmierschnittstellen muss über verschiedene Versionen hinweg garantiert sein. Bzw. es müssen zumindest Migrationsstrategien und -tools vorhanden sein die den manuellen Adaptions- Aufwand auf ein Minimum reduzieren.

1.9. Benutzerverwaltung

Das System muss eine Benutzerverwaltung für die Runtime (Bedienerstation, HMI) sowie für das Entwicklungssystem (Projektierungsumgebung) unterstützen. In beiden Fällen muss das System einen Schutz vor unberechtigtem Zugriff und ungewollter Manipulation gewährleisten.

Folglich muss es mit den gebotenen Mechanismen möglich sein, registrierten Benutzern Zugriff auf bestimmte Bereiche zu gewähren und zu verweigern. Benutzerrechte zur Betrachtung und Manipulation von Daten (oder, im Falle eines Runtime-Systems, zum Setzen von Steuerhandlungen) müssen selektiv verwaltet werden können.

Das System muss fähig sein, bis zu 256 Benutzer zu verwalten. Es muss möglich sein, rollenbasierte Authentifizierungsmodelle zu definieren. Für jedes Kontrollelement auf dem Visualisierungsbild muss es möglich sein, keine, eine oder auch mehrere Berechtigungsebenen zuzuweisen. Dasselbe gilt für Befehlsgaben und die Aufhebung der Verriegelung von Befehlsgaben. Ein Benutzer darf nur auf ein Kontrollelement zugreifen bzw. dieses manipulieren, wenn seine Berechtigungsebene mit jener des Kontrollelements übereinstimmt. Darum muss es auch möglich sein, jedem registrierten Benutzer keine, eine oder mehrere voneinander unabhängige Berechtigungsebenen zuzuweisen.

Jeder Benutzer muss sein eigenes Passwort wählen können und es auch online ändern können. Wenn für eine definierte Zeitspanne keine Benutzeraktivität gegeben ist, muss dies zu einem automatischen Log-out führen.

Es muss die Möglichkeit geben, Benutzer vorübergehend zu blockieren, nachdem eine gewisse Anzahl von Anmeldeversuche fehlgeschlagen ist. Die Anzahl der möglichen Anmeldeversuch sowie die Zeitspanne, in der keine weiteren Anmeldeversuche möglich sind, muss konfigurierbar sein.

Darüber hinaus muss das System die Konfiguration von unsichtbaren Buttons erlauben (z.B. Bildumschaltung, spezifische Funktionsaufrufe etc.), abhängig von dem jeweils eingeloggten Benutzer.

Das System muss außerdem die folgenden Anforderungen erfüllen:

- Es muss die Möglichkeit bieten, dass nur ein einzelner Benutzer oder nur bestimmte Benutzer Administratorenrechte haben, also, dass nur diese Personen dazu autorisiert sind, neue Benutzer anzulegen, Benutzer oder das System zu sperren und Benutzer zu aktivieren oder zu deaktivieren.

- Es muss möglich sein, einen Benutzer zu deaktivieren und ihn zu einem späteren Zeitpunkt wieder zu reaktivieren. Der deaktivierte Benutzer darf sich nicht einloggen können. Es müssen jedoch alle in der Vergangenheit von ihm getätigten Aktionen rückverfolgbar sein.

- Wenn mehrfach (Anzahl im Projekt-Engineering parametrierbar) ein falsches Passwort eingegeben wird, muss der Benutzer automatisch gesperrt werden. Die Sperre des Benutzers kann nur von einem Administrator aufgehoben werden. Dieser Mechanismus muss auch bei einem Login Versuch über die API greifen.

- Wenn mehrfach (Anzahl im Projekt-Engineering parametrierbar) ein falscher Benutzername eingegeben wird, muss das gesamte System gesperrt werden. Um das Einloggen von Benutzern wieder zu ermöglichen, muss der Administrator das System wieder entsperren. Dieser Mechanismus muss auch bei einem Login Versuch über die API greifen.

- Es muss möglich sein, die Komplexität des Passworts festzulegen
  - Passwortlänge
  - Anzahl Sonderzeichen
  - Anzahl Ziffern

- Anzahl Großbuchstaben
- Anzahl Kleinbuchstaben
- Anzahl Sonderzeichen

- Ein Passwortverlauf muss gespeichert werden können. Anzahl zuvor verwendeter Passwörter, welche nicht erneut verwendet werden können, muss definierbar sein.

- Passwörter müssen nach einer gewissen Zeit ablaufen (konfigurierbarer Ablauf von Passwörtern in Tagen). Bei dem nächsten Login muss der Benutzer sein Passwort ändern.

- Der Administrator darf nur Benutzer mit einer Berechtigungsebene anlegen, die niedriger oder gleich der seinen ist. Es können bei anderen Benutzern keine höheren Berechtigungsebenen aktiviert werden.

- Benutzer können aufgefordert werden, das Passwort zu ändern.

- Es darf nicht möglich sein, dass der Administrator Informationen über das Passwort anderer Benutzer abfragen kann. Darum muss bei einem Benutzer mit einem neuen Konto oder mit einem Passwort, das vom Administrator festgelegt wurde (Passwort vergessen) beim nächsten Login eine Passwortänderung erzwungen werden.

- Benutzer mit einem vordefinierten Ablaufdatum müssen angelegt werden können. Eine Warnung für den Benutzer bei Login, wenn dieser abläuft, muss aktivierbar sein.

- Ein e-Signaturkonzept muss bei jedem Element und für Rezeptänderungen integriert sein, das Dateneinträge ermöglicht. Wenn eine Signatur erforderlich ist, muss auch ein bereits eingeloggter Benutzer zur erneuten Eingabe seines Benutzernamens und seines Passworts gezwungen werden (eine bis zu 3 Stufen Authentifizierung sollte möglich sein). Diese Signaturhandlung muss in einem Betriebsjournal bzw. Audit Trail aufgelistet werden, gemeinsam mit einem benutzerdefinierten Text (Beschreibung der Aktivität). Optional wird der Benutzer gezwungen, ein Kommentar zu hinterlassen.

- Die eSignature muss in der Entwicklungsumgebung einfach parametrierbar sein.

- Die eSignature muss an zentraler Stelle z.B. Datentyp oder Variable (Tag) konfigurierbar sein

- Die eSignatur unterstützt externe Systeme für biometrische Erkennung (z.B. Nymi)

- Wenn eine bestimmte, konfigurierbare Zeit lang keine Benutzerhandlung gesetzt wird, dann muss der Benutzer automatisch ausgeloggt werden. Dies kann je Benutzergruppe separat eingestellt werden.

Alternativ muss es möglich sein, die Definitionen der Windows® Benutzerverwaltung zu verwenden. Die Berechtigungsebenen für das Prozessleitsystem müssen im Microsoft Active Directory anpassbar sein (bei einer fehlenden Domäne muss alternativ eine Anbindung des Microsoft AD-LDS® möglich sein). Die gesamte Benutzerverwaltung sollte auf der Ebene des Betriebssystems definiert werden. Im Prozessleitsystem sollte nur der Windows®-Benutzer mit Benutzername und Passwort registriert sein. Die gesamte Aufzeichnung der Benutzeraktivität muss auf dieselbe Art und Weise erfolgen wie bei einem integrierten Benutzer.

Folgende 3 Szenarien sollten als Verhältnis von Anlagen Rechner und Windows Domain Controller möglich sein:

- Der Anlagen Rechner ist ein Mitglied der Windows Domäne. Am Anlagen Rechner wird mit einem gültigen User der Windows Domäne eingeloggt. Die Benutzer Rechte werden aus der verbundenen Domäne bezogen.

- Der Anlagen Rechner ist Mitglied einer speziellen Automatisierungs- Domäne. Am Anlagen Rechner wird mit einem gültigen User einer anderen (überlagerten) Windows Domäne eingeloggt. Die Benutzer Rechte werden aus der überlagerten Domäne bezogen.

- Der Anlagen Rechner ist kein Mitglied einer Domäne. Am Anlagen Rechner wird mit einem gültigen User einer frei zu definierenden Windows Domäne eingeloggt. Die Benutzer Rechte werden aus dieser nicht verbundenen Domäne bezogen.

Das System muss die Einbindung in einen RADIUS (Remote Dial in User Service) Authentifizierungsservice unterstützen.

1.10. Anlagenmodell

Mit dem System muss es möglich sein das gesamte Modell einer Anlage abzubilden und beliebige Maschinen, Gebäude und Abläufe zu erstellen. Zur Runtime und im Engineering können damit Daten gruppiert und gefiltert werden. Anlagenmodelle können auch als Bild zur Runtime angezeigt und als Filter für andere aufgeschaltete Bilder benutzt werden. Eine hierarchische Filterung muss unterstützt werden. Dies bedeutet: Eine Variable muss nur auf einer Ebene zugeordnet werden und wird dann automatisch in einem Filter für übergeordnete Ebenen eingebunden.

Anlagenmodelle können projektübergreifend oder lokal in einzelnen Projekten erstellt werden.

Verwendung:

Engineering: Im Engineering werden Anlagenmodelle zum Filtern benutzt. So kann bei der Projektierung auf bestimmte Anlagen eingeschränkt werden.

Runtime: Zur Runtime erfolgt die Verknüpfung mit Anlagenmodellen bei der Ausführung von Funktionen, um Informationen zu bestimmten Anlagen zu erhalten und online zu filtern.

Beispiel Variablen: Bei der Projektierung im Engineering werden Variablen für bestimmte Anlagen ausgefiltert. Zur Runtime werden über Anlagenfilter Alarme für bestimmte Variablen zusammengefasst.

Es muss möglich sein, mit dem Anlagenmodell eine Alarmhierarchie aufzubauen. Dabei sollen Alarme von Variablen, aufgeteilt auf Alarmklassen, über die Hierarchie des Anlagenmodells aggregiert (summiert) werden. Die Darstellung der Alarmsummen muss einerseits in der Darstellung des Anlagenbaumes, andererseits auf eigenen Bildelementen möglich sein. Dabei soll nicht nur der Alarmstatus, sondern auch die Anzahl der aktiven und unquittierten Alarme als Zahl dargestellt werden.

1.11. Interaktive Benutzerführung

Es müssen Kontextinformationen in das Konzept der Benutzerschnittstelle integriert sein. Bestimmte Visualisierungen (z.B. Prozessbilder), Betriebszustände oder Alarmzustände können mithilfe von unterstützenden Informationen erklärt werden. Diese Informationen sollten angemessen in das Visualisierungskonzept integriert sein und für den Bediener leicht zugänglich sein.

Das System muss deshalb eine Methode zur Integration und Anzeige von unterstützenden Informationen je nach betrieblichem Kontext bieten.

- Zuweisung von Hilfekapiteln (HTML-Dokumente) zu individuellen Visualisierungsbildern

- Zuweisung von Hilfekapiteln (HTML-Dokumente) zu individuellen Alarmen

- Hilfetexte, die über ein Kontextmenü zur Verfügung gestellt werden (basierend auf HTML Help)

- Bereitstellung von Hilfetexten in der ausgewählten Betriebssprache

1.12. Bedienoberfläche

Das System muss eine mehrsprachige Bedienoberfläche für den Anwender unterstützen. Dabei darf die Anzahl der Anzeigesprachen nicht limitiert werden. Die Anzeigesprachen muss jederzeit im Laufzeitsystem ohne beenden des Systems umschaltbar sein. Dabei sollen ebenfalls verknüpfte Dokumenten (z.B. Online-Hilfetexte) ebenfalls in der Sprache umgeschaltet werden. Mit der Sprachumschaltung sollen ebenfalls die Anzeigeschriftarten gewechselt werden, damit die Unterstützung von Zeichen der jeweiligen Sprache möglich ist.

Optional sollen mit der Sprachumschaltung ebenfalls die angezeigten Einheiten gewechselt werden. Dabei muss die Umrechnung der Einheiten durch Standardfunktionen des Systems ohne zusätzliche Programmierung möglich sein.

1.13. Produktlizenzverwaltung

Computerspezifische Softlizenzen sollen unterstützt werden. Alternativ gibt es die Möglichkeit, die Softwareplattform durch einen USB-Dongle zu schützen. Die unterschiedlichen Lizenzformen sollen im Rechnerverbund beliebig kombiniert werden können.

Lizenzpflichtige Komponenten des Softwaresystems (Module, Programme, Tools) sollen über eine zentrale Lizenzinfrastruktur verwaltet werden können. Mithilfe eines entsprechenden Softwaretools soll der Anwender Aufgaben wie Lizenzbezug, -aktivierung, -verwaltung oder -updates erledigen können.

Die Aktivierung einer Lizenz muss aus Sicht des Produktiv-Rechners, welcher die lizenzpflichtige Komponente betreibt, sowohl online als auch offline erfolgen können. „Online" bedeutet, dass mittels aktiver Verbindung zu einem zentralen Lizenzdepot über das Internet eine Lizenz bezogen und aktiviert werden kann. „Offline" bedeutet, dass eine Lizenz auch manuell auf das Zielsystem übertragen werden kann, wenn dieses keine Internetverbindung hat.

Lizenzen sollten für alle relevanten Komponenten als Software-Dongle oder Hardware-Dongle bezogen werden können. Die Konsumation einer Lizenz kann lokal (am Rechner) sowie über ein Netzwerk erfolgen. Die aktuell lizenzierten Funktionen und Module müssen an jeder Station überprüfbar sein. Die Zuordnung einer Lizenz zu einem Zielsystem muss explizit erfolgen können. D.h., wenn eine Lizenz am Zielsystem aufliegt (z.B. Dongle mit Lizenz ist angeschlossen), wird diese nicht automatisch konsumiert. Erst bei expliziter Zuweisung wird eine Lizenz verwendet.

Der Umfang einer Lizenz wird durch einen spezifischen Code (Serial Nummer) definiert. Das System muss die Vormerkung zweier oder mehrerer Serial Nummern unterstützen (Liste von Serial Nummern). Bei Ausfall einer übergeordneten Serial Nummer wird auf die jeweils nachfolgende Serial Nummer (falls vorhanden) zurückgegriffen.

Stellvertretend für den tatsächlichen Zielrechner kann der Lizenzierungsvorgang von einem Vermittlungsrechner aus durchgeführt werden (sog. Remote-Lizenzierung).

Es muss ebenfalls möglich sein, die Abarbeitung grundlegender Lizenzierungsabläufe in einem Batch-Job abzubilden. Eine entsprechende Funktion oder Kommando muss die Aktivierung einer Einzellizenz, das Update aller Lizenzen eines Rechners sowie die Eintragung einer Lizenz in die Liste der Serial Nummern ermöglichen. Das System unterstützt des Weiteren die „Massenaktivierung" von

Lizenzen. Mithilfe eines definierten Konfigurationsformats können mehrere Lizenzierungen für dedizierte Zielrechner automatisch durchgeführt werden.

2. Funktionen des Projektierungssystems

Die Projektentwicklung und -wartung stellt einen großen Teil der Investition beim Einsatz von HMI/SCADA-Systemen in der Automatisierung dar. Dabei geht es auf der einen Seite um die anfängliche Projekteinrichtung und Inbetriebnahme. Auf der anderen Seite geht es dabei um Wartungstätigkeiten sowie Änderungen und Erweiterungen des Systems. Diese Aufgaben können von verschiedenen Technikern, Bedienern und Wartungsbediensteten ausgeführt werden.

Das Engineering System sollte einen globalen Ansatz unterstützen. So lassen sich zum Beispiel mit einem Globalprojekt Standards projektübergreifend festlegen. Damit ist es möglich, bestimmte Module oder Elemente für alle Standorte zu definieren, während lokale Projekte individuelle Parameter festlegen.

In globalen Komponenten sollten enthalten sein:

- Alarmierung: Deklaration der global zu verwendeten Alarmgruppen, Alarmklassen und Alarmbereiche.

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

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

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

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

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

- Stile: Stile im Engineering fassen grafische Eigenschaften von Bildelementen zusammen. Das bedeutet, grafische Parameter von Elementen wie Linienstärke, Größe, Farbe etc., werden für die benötigten Elemente in einem Projekt vordefiniert. Die Stile werden zentral in einem Globalprojekt gespeichert und können dann auf alle weiteren Elemente übertragen werden. Das stellt sicher, dass das Design in einem Projekt (oder auch projektübergreifend) durchgängig bleibt.

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

Darum muss das System klar strukturierte Methoden für die Projektentwicklung sowie konsistente Möglichkeiten für die Definition, Wartung und Erweiterung von Funktionalität bieten, wie in der Folge beschrieben wird.

2.1. Anwendungsentwicklung

Das System muss grundsätzlich aus einem Entwicklungssystem und einer Runtime-Komponente bestehen, die über eine Online-Verbindung zusammenarbeiten müssen. Diese Verbindung muss entweder lokal oder über eine Netzwerkverbindung erfolgen, das heißt eine Projektierungsstation muss sich mit jeder verfügbaren Runtime-Station über das Netzwerk verbinden können. Dies bedeutet auch, dass ein Entwicklungssystem unabhängig von einem Runtime-System installiert werden kann. Die Mehrbenutzerentwicklung muss durch passende Methoden unterstützt werden. Das System muss einen konsistenten Zustand zwischen Entwicklungsprojekt („Definition der Funktionalität“) und Runtime-Projekt („instanziierte Funktionalität“) garantieren.

Es muss möglich sein, unterschiedliche Abschnitte und Funktionalitäten eines Projektes in einer modularen Weise zu verwalten und somit getrennte Unterprojekte in einem gemeinsamen Entwicklungskontext (also „Entwicklungs-Arbeitsbereich“) zu generieren. Es muss möglich sein, Ressourcen zwischen verschiedenen Unterprojekten zu teilen, wie z.B. Variablen, Alarmnachrichten-Definitionen oder grafische Symbole. Es muss eine Möglichkeit geben, verschiedene Projektentwicklungsstadien zu verwalten (Projekt Versionierung etc.) sowie Änderungen in einem Entwicklungsprojekt nachzuverfolgen. Eine gewisse Art von Projektdokumentation oder Projektzusammenfassung muss zu jeder Zeit abrufbar sein. Das kann z.B. ein HTML-Dokument oder ein Compiled-HTML-Dokument sein.

Alle funktionalen Komponenten müssen in einer objektorientierten Art und Weise organisiert sein. Folglich müssen auch alle zugehörigen Attribute in aussagekräftigen Kategorien organisiert sein. Die Konfiguration muss, soweit Standard-Funktionalitäten betroffen sind, durch Wizards geleitet werden. Das Vererbungsprinzip muss anwendbar sein. Es muss eine Möglichkeit geben, die Wertevererbung zwischen über- und untergeordneten Objektinstanzen selektiv zu entkoppeln und wieder zu verbinden.

Die Projektierung der Visualisierung und des Bildlayouts muss durch einen WYSIWYG-Engineering (What-You-See-Is-What-You-Get) unterstützt werden, der die einfache Anordnung, Konfiguration und Verbindung von Visualisierungskomponenten ermöglicht. Gängige Usability-Funktionen wie Drag & Drop oder die Ausrichtung von Komponenten untereinander müssen unterstützt werden. Im Zuge der Bildbearbeitung muss es möglich sein, einzelne Editierungsschritte rückgängig zu machen. Bei übereinanderliegenden Bildelementen muss eine effektive Selektion des gewünschten Elements durch geeignete Mechanismen unterstützt werden.

Es ist von höchster Wichtigkeit, dass Standardfunktionen im Bereich HMI/SCADA und Reporting von Nichtprogrammierern entwickelt und gewartet werden können. Darum muss das System konfigurierbare Module für Standardaufgaben bieten, die lediglich durch das Konfigurieren der jeweiligen Modulattribute eingerichtet werden können. Standardaufgaben wie die Alarm- oder Rezeptverwaltung müssen deshalb durch vordefinierte Datenverarbeitungsmodule (Ereignislisten etc.) und die zugehörige Visualisierung unterstützt werden. Diese Lösungen müssen den Benutzern als Vorlagen in der Projektierungsumgebung angeboten werden. Falls benötigt, müssen die jeweiligen Implementierungskomponenten der Lösung (Visualisierungsbild, Datenverarbeitungsroutinen, Kommunikationsroutinen etc.) automatisch eingerichtet und angeordnet werden. Der Entwickler muss anschließend dazu imstande sein, die Funktionalität mit spezifischen Parametern anzupassen oder die gegebene Funktionalität voll zu adaptieren, z.B. indem er auf die verfügbaren Schnittstellen zugreift.

Das System muss eine Möglichkeit bieten, grafische Symbole zu erstellen und zu verwalten sowie dementsprechende Symbolbibliotheken zu verwalten. Dadurch muss die Erstellung von

prozessspezifischen Illustrationen und Animationen ermöglicht werden. Das System muss grundlegende Mechanismen bieten, um das Aussehen und die Anordnung von grafischen Elementen zu ändern und einzelne Symbole in ein größeres Schema zu gruppieren.

Das System bieten eine Möglichkeit zur Erstellung und Verwaltung von Engineering Vorlagen für voll funktionale Prozessbausteine. Diese bündeln Funktionen auf unterschiedlichen Ebenen, etwa das Datenmodell (Variablen, Treiber) die zugehörige Steuerungslogik und Algorithmen sowie sämtliche grafischen Ansichten des Bausteins (Übersichtssymbole, Detailbilder). Es können beliebig viele Instanzen aus diesen Vorlagen erstellt werden, wobei jede Instanz über eine vordefinierte Zusammenstellung an Parametern individuell konfiguriert werden kann. Im Weiteren muss der Bezug zwischen Vorlage und Instanz erhalten bleiben, sodass Änderungen and der Vorlage auch systematisch in dessen abhängige Instanzen übertragen werden. Die Zuweisung von Variablen einer Vorlage zu den tatsächlichen Variable im Automatisierungsprojekt muss im Zuge der Instanziierung auf effektive Weise erfolgen können. Das System muss die Sicherung (Export) bzw. den Austausch von Prozessbaustein Vorlagen zwischen unterschiedlichen Arbeitsplätzen unterstützen.

Das System muss absolut abwärtskompatibel sein. Das bedeutet, dass Projekte, die in Vorgängerversionen entwickelt wurden, auch in Nachfolgeversionen der Projektierungsumgebung ohne Datenverlust geöffnet und gewartet werden können. Außerdem müssen Runtime-Projekte in neueren Versionen der Runtime-Software ohne Einschränkungen lauffähig sein.

Das System muss eine Funktion zur Projektsicherung bieten, die alle zugehörigen Definitionen und Ressourcen in einer einzelnen ZIP-Datei verpacken kann. Abhängig von den benutzten Mediendateien (Bilder, Audiodateien, Videos etc.) muss die Sicherungsdatei eine begrenzte Größe haben, sodass sie per E-Mail versendet werden kann.

Das System muss/sollte Funktionen bieten Projektbackups zu vergleichen, um Versionsunterschiede und Änderungen sichtbar zu machen. Änderungen und Unterschieden zwischen Versionen werden in einer übersichtliche und nachvollziehbaren weise dargestellt.

2.2. Variablenhandhabung

2.2.1. Variablenerstellung für die Kommunikation

Es muss möglich sein, Variablen einzelner, einfacher Datentypen zu erstellen sowie von strukturierten Typen. Die Erstellung von Arrays muss möglich sein. Die Erstellung von Arrays muss mit einfachen Variablen, mit Struktur-Variablen sowie mit individuellen Struktur-Elementen funktionieren.

Alle Variablen müssen auf definierten Datentypen basieren, von denen sie alle Eigenschaften übernehmen. Einfache Variablen müssen auf einfachen Datentypen basieren, Struktur-Variablen müssen auf Struktur-Datentypen basieren, die wiederum aus mehreren einfachen Datentypen und/oder Struktur-Datentypen zusammengesetzt sind.

Bei Variablen muss das Konzept der Vererbung unterstützt werden. Es muss also möglich sein, alle Eigenschaften von einem übergeordneten Datentyp zu erben. Änderungen an den Definitionen übergeordneter Datentypen müssen dementsprechend auch an alle untergeordneten (abgeleiteten) Variablen weitergegeben werden. Außerdem muss es eine Möglichkeit geben, diese Vererbungsbeziehung für jede Eigenschaft einzeln zu trennen und wieder zu verbinden. Eigenschaften, die aus der Vererbungsbeziehung ausgeschlossen wurden, dürfen nicht von untergeordneten Variablen übernommen werden.

Die Erstellung von Variablen und deren spezifischen Vererbungsbeziehungen darf nicht zahlenmäßig begrenzt sein.

Zusätzlich zum Projektierungstool muss die Erstellung von Variablen auch durch entsprechende Export-/Import-Mechanismen unterstützt werden. Es muss also möglich sein, Variablendefinitionen (Listen) aus Steuergeräten von Drittherstellern zu importieren. Insbesondere muss es möglich sein, Variablen direkt von IEDs zu importieren, die auf IEC 61850, IEC 60870, OPCUA, BacNet, Siemens S7 (TIA) oder Codesys basieren.

In der Projektierungsumgebung müssen in Variablenlisten entsprechende Filteroptionen (vor allem nach Name, Datentyp, Funktionsverwendung, zugehörige Alarme, Datenquelle) angeboten werden, um den Entwickler zu unterstützen.

Die Massenbearbeitung von Variablen, im Sinne einer Mehrfachselektion und synchroner Änderung gemeinsamer Eigenschaften muss unterstützt werden. Das System muss dabei eine angemessene Reaktionszeit, auch bei großen Variablenmengen bis zu mehreren hunderttausend Variablen gewährleisten, Auch der Wechsel zwischen unterschiedlichen Anordnungen zur Ansicht der Variablenliste im Engineering, wie das Sortieren- oder die Gruppierung anhand von Eigenschaften, muss in angemessener Zeit erfolgen. Vorausgesetzt werden kann hierbei die Verwendung eines Rechnersystems (Engineering PC) mit einer Leistungsspezifikation gemäß den Herstellerangaben.

Individuelle Variablen müssen durch einen eindeutigen Namen voneinander unterscheidbar sein. Die Definitionsinformationen von Variablen müssen dauerhaft im System gespeichert werden. Wenn also eine Variable gelöscht und später unter demselben Namen neu angelegt wird, müssen alle Verknüpfungen wiederhergestellt werden. Zusätzlich muss es möglich sein, für jede Variable mehrere textbasierte Beschreibungsattribute festzulegen.

Informationen über die Anzeige und das Verhalten (z.B. Farbinformationen) der Anwendung sowie Informationen für die weitere Verarbeitung (z.B. Alarme, Druck) müssen auf dem Status/Wert von individuellen Variablen basieren.

Die Variablen müssen der Mittelpunkt für die Definition des Anwendungsverhaltens sein. Dementsprechend müssen Farbumschaltung, Wertoperationen (z.B. Skalierung) oder die Auslösung von Funktionen und Alarmen (z.B. wertabhängig) implizit mithilfe von variablenbezogenen Eigenschaften konfigurierbar sein.

Funktionen für arithmetische und logische Verknüpfungen müssen im Basissystem angeboten werden. Zusätzlich zu den grundlegenden arithmetischen Operationen und Klammerfunktionen müssen komplexe Berechnungen, wie z.B. trigonometrische Funktionen, statistische Berechnungen und Datenverdichtung über parametrisierte Zeitintervalle, unterstützt werden. Es muss möglich sein, berechnete und abgeleitete Informationen weiterzuverarbeiten.

Die Verwendung zusätzlicher Ressourcen und Definitionen wie Datentypen, Grenzwerte oder SPS-Programme muss an der jeweiligen Variable angezeigt werden, um ggf. einen direkten Wechsel der jeweiligen Einstellung zu ermöglichen.

2.2.2. Variablenanzeige

Es muss möglich sein, Variablen von verschiedensten Datenquellen (Leitstand, SPS) in einem Bild anzuzeigen. Die Qualität (online oder gestört) jeder Variable muss sofort ersichtlich sein. Wenn eine Variable gestört ist (z.B. keine Onlineverbindung), muss das System automatisch einen definierten Backupwert einstellen.

Jeder Variable muss neben dem Wert einen Zeitstempel (intern oder extern) und Statusinformationen bereitstellen, die das System analysieren kann und auf die es entsprechend reagieren kann.

Das Bediensystem sollte über eine Online Variablen Diagnose verfügen, die es einem Anlagen Betreiber erlaubt die Qualität seiner Daten Kommunikation zu überprüfen. In diesem Management Screen können unter anderem auch Datenpunkte aktiviert oder deaktiviert werden.

Es muss möglich sein, einzelne Variablen oder Gruppen von Variablen im Online-Modus zu aktivieren und zu deaktivieren.

Um die allgemeine Netzwerkbelastung zu reduzieren, müssen numerische Werte mit einem Schwellwert versehen werden, der definiert, ab welcher Differenz Änderungen zu dem Prozessleitsystem übertragen werden.

Lineare oder nichtlineare Wertanpassungen müssen für jeden numerischen Wert möglich sein.

Es muss einen Zugriffschutz für Variablen geben. Es muss also möglich sein, selektiv festzulegen, ob es auf eine Variable nur Lesezugriff oder Lese- und Schreibzugriff gibt.

2.2.3. Variablenkommunikation

Das Prozessleitsystem muss als Daten Server z.B. OPC UA-Server oder IEC 60870 Server agieren können, um den Transfer von Variablenwerten zu den entsprechenden Protokoll Clients zu ermöglichen.

Zusätzlich muss das System die Möglichkeit bieten, als Modbus-RTU-Slave (Remote Terminal Unit, seriell oder TCP/IP) zu agieren, damit Modbus-Master Daten lesen und schreiben können, die auf dem Runtime-System gehalten werden.

Das Prozessleitsystem muss die Möglichkeit bieten, ein Prozessabbild von frei wählbaren Variablen zyklisch in eine SQL-Datenbank zu schreiben (Archivierung, siehe Abschnitt 11).

2.2.4. Variablenablage

Es muss genau eine zentrale und konsistente (logische) Datenquelle geben. Alle Variablen müssen in einer frei zugänglichen Standard-SQL-Datenbank abgelegt sein. Die benötigte IEC 61131-3 Programmierumgebung muss ebenfalls auf diese Variablen zugreifen. Änderungen bei den Variablen müssen in der IEC 61131-3 Programmierumgebung sowie in der Entwicklungsumgebung des Prozessleitsystems angezeigt werden. Wenn eine Änderung an einem Systemteil gemacht wird, muss der andere Teil verständigt werden, damit die Systeme konsistent zueinander bleiben. Deswegen müssen beide Systeme mit einem bidirektionalen, ereignisgesteuerten Datenaustausch-Mechanismus ausgestattet sein, um mit der Datenbank zu kommunizieren.

2.3. Mehrplatzfähige Projektierung

Das Projektierungstool muss die parallele Arbeit von mehreren Personen an einem Projekt unterstützen. Das System muss Mechanismen anbieten, die vermeiden, dass mehr als eine Person zur selben Zeit an demselben Teil des Projekts arbeitet und somit einen inkonsistenten Projektzustand verursacht. Darum müssen Projektteile (Module), die gerade bearbeitet werden, unzugänglich (gesperrt) für alle anderen Entwickler sein. Das Projektierungstool muss Informationen über den Ort (Workstation-ID) und den Benutzer anzeigen können, der zu einem bestimmten Zeitpunkt exklusiv an einem bestimmten Projektteil arbeitet. Änderungen durch einen Entwickler an einem Projekt dürfen sich nicht sofort auf das Master-Projekt auswirken. Das System muss einen Vor-Test auf einer unabhängigen Projektkopie (z.B. lokale Arbeitskopie) unterstützen. Dementsprechend muss es möglich sein, dass die Entwickler Änderungen, die bereits getestet wurden, Schritt für Schritt ins Master-Projekt übernehmen.

Außerdem muss das System die Möglichkeit bieten, einzelne Projektmodule selektiv zu schützen, sodass nur autorisierte Nutzer auf bestimmte Module zugreifen und diese modifizieren können.

2.4. Versionierung und Änderungen

Das System muss die Möglichkeit bieten, Projekte und untergeordnete funktionale Module mit Versionen zu versehen. Es muss möglich sein, alle Projektänderungen umfassend zu protokollieren, um volle Nachverfolgbarkeit zu garantieren und einen möglichen Revalidierungs-Aufwand zu reduzieren. Die Änderungshistorie muss aus Informationen bestehen wie z.B. geändertes Objekt, Benutzer, Datum und Zeit, Wert vor und nach der Änderung. Außerdem muss es die Option geben, zwei verschiedene Projektversionen miteinander zu vergleichen und die bestehenden Unterschiede hervorzuheben.

Eine Anbindung an eine externe Versionierungssoftware – wie z.B. VersionDog von Auvesy – muss ohne zusätzlichen Projektier- oder Programmieraufwand möglich sein.

2.5. Automatische Projektierung

Um spezifisches Anwendungswissen zu bewahren und mehrfachen Implementierungsaufwand für einzelne, jedoch wiederkehrende Entwicklungsaufgaben zu verhindern, muss das System Möglichkeiten für die Wiederverwendung von Code und die automatische Projektierung anbieten. Darum muss es möglich sein – zumindest für fortgeschrittene Benutzer – gewisse Lösungskomponenten, die standardisiert werden können, in wiederverwendbare Lösungspakete zu übertragen. Andere Entwickler müssen die Möglichkeit haben, diese Vorlagen zu verwenden. Die Umsetzung von automatischer Projektierung muss von der Kapselung einfacher Datenverarbeitungsaufgaben in Programmierbibliotheken über die Vorbereitung bestimmter Visualisierungsbilder (homogenes Visualisierungsdesign) bis hin zur Erstellung umfassender Projekterstellungs-Wizards reichen. Projekterstellungs-Wizards müssen beim schrittweisen Einbeziehen (bzw. Ausschließen) sowie bei der Parametrisierung bestimmter Funktionen und Bilder (z.B. Alarmverwaltung, Ereignisprotokollierung, Benutzerverwaltung etc.) für die Erstellung einer Lösungsvorlage assistieren.

Die Erstellung von Programmen für die softwaregestützte Projekterstellung und -verwaltung erfolgt auf Basis einer integrierten Schnittstelle (API / Objektmodell), welche die Methoden und Parameter des Engineerings umfassend widerspiegelt. Diese Schnittstelle kann für die Entwicklung in gängige Softwareentwicklungsumgebungen eingebunden und dort bedient und diagnostiziert werden. Insbesondere wird die Microsoft Visual Studio Umgebung unterstützt. Programme und Tools können mit jeder .NET Programmiersprache erstellt werden. Eine IDE-Unterstützung steht für die Programmiersprache C# zur Verfügung.

Hilfsprogramme können für einen expliziten Durchlauf als Assistent oder Wizard konzipiert werden, sowie für den dauerhaften Betrieb als Dienst, etwa für Validierungen oder integrierte Erstellungshilfen. Das System verfügt über integrierte Funktionen für das Einrichten und Verwalten dieser Programme.

3. Funktionen des Runtime-Systems

3.1. Optimale Nutzung der Rechnerleistung

Das Runtime System soll die am Zielrechner zur Verfügung stehende CPU-Leistung bestmöglich nutzen. Dies betrifft einerseits die optimale Nutzung der Rechnerarchitektur, also gegebenenfalls auch die Fähigkeit zur Aufteilung bestimmter Teilprozesse auf unterschiedliche Rechnerkerne. Andererseits muss durch die verteile Berechnung von Standardaufgaben eine gegenseitige Beeinflussung (sprich Verlangsamung) vermieden werden.

Als Standardaufgaben in diesem Sinne sind insbesondere zu verstehen:

- Runtime Visualisierung (HMI-Updates bzw. Bedienbarkeit)
- Betrieb einer Archivierung, sowohl zyklisch als auch spontan
- Kommunikation und Live-Abgleich von Prozessdaten zwischen vernetzten Runtime Instanzen
- Werteaufnahme und interne Werteverteilung für Prozesskommunikation (Direkttreiber)
- Erforderliche Kommunikation und erforderliche Abgleichsoperationen bei Redundanzumschaltungen

3.2. Bedienung des Runtime-Systems

Eine Bedienung des Runtime-Systems über (Multi-)Touchscreen, Keyboard und Maus muss unterstützt werden. Für die einfache Touch-Bedienung muss das System frei definierbare virtuelle Keyboards unterstützen. Multi Touch sollte für die wichtigsten Operationen (z.B. zoomen im Trend) nativ vom System ohne zusätzlichen projektier oder Kodier-Aufwand unterstützt werden.

Das System muss auf über Standardmenüs und Kontextmenüs bedienbar sein. Um ein Kontextmenü auf einem Touchscreen aufrufen zu können, muss es eine Funktion für die Emulierung eines Rechtsklicks geben.

Es muss möglich sein, ein externes Programm mit Parametern zu starten, entweder über die Benutzerschnittstelle oder über den Aufruf einer Funktion. Das Basis-System muss Datei-Operationen wie das Kopieren von Dateien erlauben.

Das Runtime System muss eine Aktualisierung der zugehörigen Projektinformationen ohne Beenden und Neustart des Systems unterstützen. Diese Aktualisierung muss sowohl vom Bediener des Laufzeitsystems als auch remote über das Projektierungssystem möglich sein.

3.3. Systeminformation

Das Basis-System muss die Möglichkeit bieten, bestimmte Systemzustände zu überwachen. Es muss möglich sein, aktuelle Systemzustände, wie z.B. „Druck aktiv/inaktiv“, sowie andere systemspezifische Parameter wie Speicherauslastung oder Festplattenkapazität zu evaluieren. Die folgenden Informationskategorien müssen für eine Evaluierung während der Runtime verfügbar sein:

- **Alarme:** Anzahl der aktiven Alarme, anstehend oder quittiert
- **Archive**: Datenvolumen
- **Benutzerverwaltung**: Aktueller Benutzer, zugewiesene Berechtigungsebenen etc.
- **Drucker:** Druckername, Status ein/aus, Status druckt ja/nein, Anzahl anstehender Druckaufträge etc.
- **Hardware-Ressourcen des Host-Systems:** CPU-Auslastung, Freier Speicher in RAM, Datenbank oder auf Festplatte etc.
- **Netzwerk**: Servername, verbundene Clients, Redundanzkomponenten, Sicherungsbetrieb nach Serverausfall, Netzwerkperformance-Statistiken

- **Verbindungs- Statistik**: Information über Treiber Performance, Qualität der Daten Verbindung.

- **Rezepturen**: Zustands Informationen über die Rezepturverwaltung, zuletzt abgesetztes Rezept etc.

- **Allgemeine Projekt- und Systeminformationen**: Speicherort der Projektdefinitionen auf dem Host-System, allgemeiner Systemzustand (aktiv, Simulation etc.), Runtime-Version, Projektname, Protokollierung aktiv/inaktiv, verschiedene Statuswerte von funktionalen Modulen, Diagnoseinformationen von Kommunikationstreibern etc.

- **SNMP Traps:** Das System muss SNMP-Traps v1, v2 und v3 abfragen und empfangen können, um diese Informationen in ein zentrales Netzwerk Monitoring einbinden zu können.

- **SNMP Inform:** Das System muss SNMP-INFORM-REQUESTS v2c und v3 unterstützen.

3.4. Software-Betriebsmodi

Das System muss die folgenden Betriebsmodi hinsichtlich des zugrunde liegenden Betriebssystems (siehe auch Abschnitt 1.1) unterstützen.

- Start und Betrieb als reguläre Anwendung (Prozess). Auf der Client-Seite beinhaltet das die Anzeige der grafischen Schnittstelle (HMI) mit den zugehörigen Optionen wie Maximieren, Minimieren, Verstecken, Schließen etc.

- Start als Anwendung (Prozess) mit exklusiver Anzeige der grafischen Schnittstelle (HMI). Das System darf es dem Bediener nicht erlauben, den Anzeigekontext zu manipulieren oder auf darunterliegende Funktionen und Programme des Host-Systems zuzugreifen (wie bei modalen Fenstern, Anzeige als oberstes Element).

- Start der Anwendung auf einem Client im Simulations- Modus ohne Echtzeit Daten Kopplung.

- Betrieb von mehreren, unabhängigen Client-Instanzen im Kontext eines Terminal-Server-Setups (Einsatz von Thin Clients, siehe auch Abschnitt 12.1).

- Betrieb einer Runtime-Server-Instanz als Windows®-Dienst ohne aktives User Login am Betriebssystem.

3.5. Schwellwertverwaltung

Die Überwachung von digitalen und numerischen Werten muss über die Angabe eines Minimal- und eines Maximalwerts möglich sein. Es muss auch möglich sein, String Werte hinsichtlich bestimmter Texte oder Textabschnitte zu überwachen.

Im Falle einer Schwellwertverletzung muss das Auslösen einer Funktion (vordefinierte Systemfunktion oder benutzerdefinierter Skriptcode) möglich sein. Die entsprechenden Schwellwerte müssen direkt über Variablenattribute definiert werden. Alternativ muss die Sammlung von Grenzwerten unabhängig voneinander erstellt und verwaltet werden. Die entsprechenden Konfigurationselemente müssen zu verschiedensten Variablen zuordenbar sein. Als Option muss jede Wertänderung als Grenzwertverstoß interpretiert werden können. Außerdem muss es möglich sein, Statusänderungen von Variablen zu evaluieren.

Es muss möglich sein, jedem Schwellwert eine Hysterese zuzuweisen, um unnötige Schwellwertverletzungen bei flatternden Werten zu verhindern. Es muss auch möglich sein, eine

Schwellwertverzögerung zu konfigurieren sowie Schwellwerte dynamisch festzulegen, also über andere Wertvariablen definiert.

3.6. Wertstatusverwaltung

Es ist wichtig, den Status und die Qualität (Verbindungsqualität) einer Variablenaktualisierung zu evaluieren. Darum muss der Status einer Variable aufgelöst werden (Wert von IED empfangen, Ersatzwert, manueller Wert, Verbindung unterbrochen, nicht aktualisiert). Die weiterführende Verarbeitung wird dem Qualitätsmanagement dienen.

Jede Variable muss Statusinformationen anbieten, die für bedingte Reaktionen evaluiert werden können oder für statistische Zwecke (z.B. Erkennung einer abfallenden Verbindungsqualität) analysiert werden können. Diese Statusinformationen müssen archiviert werden, um die Rückverfolgbarkeit der jeweiligen Bedingung zu einem späteren Zeitpunkt sicherzustellen.

3.7. Alarmierung

Alarme müssen aus spezifischen Variablenänderungen resultieren. Darum müssen Alarme, die ausgelöst werden sollen, im Kontext der jeweiligen Variableneinstellungen konfiguriert werden.

Alarme müssen durch Schwellwertverletzungen ausgelöst werden. Es muss möglich sein, Alarme unabhängig voneinander zu klassifizieren und zu gruppieren.

Eine individuelle Alarmklassifikation muss möglich sein, um die Priorität (bzw. den Schweregrad) eines Alarms anzuzeigen Außerdem müssen unabhängig von der Alarmklassifikation logische Alarmgruppen definierbar sein. Diese Gruppen können z.B. eine Gruppierung nach miteinander gekoppelten Systemelementen (z.B. Kühlsystem) repräsentieren. Zusätzlich werden Alarme, unabhängig von den oben erwähnten Merkmalen, in standortbezogene Segmente gruppiert werden, wie z.B. Zellen, Hallen, Anlagen oder Regionen. Diese Art von Gruppierung muss eine konsistente und skalierbare Anzeige von Alarminformationen in Kombination mit umfassenden Schemata erleichtern, wie in Abschnitt 4.2 beschrieben.

Für jede(n) Alarmgruppe, -klasse oder -bereich muss ein(e) spezifische(r) Name, Nummer (ID), Farbe, Funktion, Statusvariable und Grafik zuweisbar sein.

Es muss möglich sein, Alarmgruppen selektiv abzuschalten, z.B. für Wartungsarbeiten oder um Sammelalarme in der Steuerung zu ermöglichen.

Hinweis zum Engineering von Alarmgruppen, -klassen oder -bereichen:
Die Definition von Alarmgruppen, -klassen oder -bereichen muss über ein strukturiertes Format erstellt oder übertragen werden können. Hierzu soll eine XML-basierte Export/Import Schnittstelle zur Verfügung stehen.

Für jeden Alarm muss eine Verpflichtung zum Quittieren oder Löschen des Alarms definierbar sein. Außerdem muss es möglich sein, beim Auftreten eines Alarms Variablen hervorzuheben (z.B. durch Blinken). Eine Quittierung dieses hervorgehobenen Status muss unabhängig möglich sein.

Alarme müssen online auf einem Listendrucker druckbar sein sowie – als Option – im Betriebsjournal aufzeichenbar sein.

Die Alarmbenachrichtigungszeile muss immer in das Visualisierungsbild eingebettet sein. Sie muss den ältesten und den letzten unquittierten Alarm anzeigen.

Alarme müssen in einer Alarmmeldeliste angeführt werden. Außerdem müssen sie die Alarmbenachrichtigungszeile im Prozessbild aktivieren (Alarmanzeige in den Vordergrund schalten).

Des Weiteren muss es möglich sein, anhand eines Anlagenmodells bzw. einer Anlagenstruktur die Anzahl aktiver Alarme eines Teilbereichs einer Anlage oder die (verdichtete) Gesamtanzahl von Alarmen auf übergeordneten Ebenen anzuzeigen.

Als Option müssen die Alarme auf dem Benachrichtigungsdrucker ausgegeben werden und innerhalb des Prozessbildes mit obligatorischer Quittierung kenntlich gemacht werden können (z.B. Hervorheben der betroffenen Komponente).

Die Quittierung von Alarmen muss in Listen in dedizierten Ansichten (Bildern) möglich sein. Die Quittierung von Alarmen in Listen muss individuell oder in einer Seitenansicht möglich sein. Eine Filteroption muss in den Listenansichten angeboten werden. Alle Alarmattribute müssen für die Alarmfilterung benutzbar sein. Für kritische Alarme muss ein Kommentar zwingend erforderlich sein bevor das System eine Quittierung des Alarmes erlaubt.

Für jeden Alarm müssen mindestens folgende Attribute in einer Listenansicht angezeigt werden:

- Alarmnummer (ID)
- Alarmstatus
- anstehender Zeit
- Zeitstempel von Auftreten, Quittieren und Löschen
- verbundene Variable
- Alarmtext
- Kommentare des Bedieners. Das System muss für den Anwender die Möglichkeit bieten aus einer vor selektierten (zentral verwalteten) Liste von Kommentaren auszuwählen, um Fehleingaben zu vermeiden und die Auswertung zu erleichtern.
- weitere Informationen wie Benutzer, Computername, Alarmgruppennamen und -Klassen Namen sowie die zugehörige Gruppe, Klassennummern, Alarmbereich

Die Zeitanzeige muss optional in Millisekunden oder Mikrosekunden möglich sein. Alle Informationen bezüglich eines Alarms müssen auf eine Zeile im Bild passen.

Bei Kommunikationsprotokollen mit externer Zeitstempelung muss es möglich sein, beide Zeitstempel, extern und intern, in jeweils einer Zeitspalte darzustellen.

Spaltentiteln in der Liste müssen frei definierbar sein und die Sprache muss anpassbar sein.

Es muss eine Möglichkeit geben, eine verdichtete Anzeige von Alarmen bei Reaktivierung zu erhalten. Das bedeutet, dass ein wiederholtes Auftreten desselben Alarms nicht mehrmals angezeigt wird. Stattdessen muss der letzte Eintrag gemeinsam mit der Anzahl des Auftretens des Alarms in der Alarmmeldeliste angezeigt werden.

Die Text- und Hintergrundfarbe eines Eintrags müssen – als Option – von den eingestellten Alarmklassenfarben abgeleitet werden. Es muss eine Möglichkeit geben, einzelne Alarmeinträge nach Status (Quittiert, Unquittiert) grafisch hervorzuheben.

Außerdem muss es möglich sein, Alarme nach frei definierbaren Kriterien zu filtern (z.B. stündlicher/wöchentlicher/täglicher Filter etc.) – Zeiträume, Prioritäten, Anlagenmodell Gruppen und Texte in beliebigen Kombinationen. Das Filtern nach Alarmen, die aktiv, nicht quittiert sind, eine Kommentar/Alarmursache erfordern oder quittiert wurden, muss möglich sein, genauso wie das

Filtern von historischen Alarmen. Die jeweiligen Filter müssen entweder während der Projektierung oder während der Runtime über das HMI anpassbar sein.

Der gefilterte Inhalt der Alarmmeldeliste muss mittels manueller Auswahl ausdruckbar sein. Der Ausdruck muss unabhängig definierbar sein, als Variation der Monitoranzeige.

Alarme, die quittiert oder gelöscht wurden, dürfen nicht aus dem Alarmprotokoll gelöscht werden, sondern müssen archiviert werden und somit für eine spätere Analyse verfügbar bleiben.

Außerdem muss für die archivierten Alarme ein „Filtern nach Zeit" möglich sein, um eine effiziente Überprüfung von Alarmen zu ermöglichen.

Die Quittierung von Alarmen muss alle alarmbezogenen Variablen zurücksetzen, einschließlich der Variablen von zugehörigen Steuermodulen.

Alarmunterdrückung (Alarm Shelving)

Das System unterstützt die temporäre Unterdrückung von Alarmen, in Anlehnung an ISA-18.2-2016 sowie IEC 62682 (Alarm Management Systeme für die Prozessindustrie). Als Alarm wird hierbei das wahlweise Über- oder Unterschreiten eines einzelnen Schwellwerts an einem einzelnen Prozesswert (Wert und/oder Status) verstanden, und somit dessen sämtliche vergangene, aktuelle wie auch künftigen Erscheinungen. Unterdrückte Alarme werden in herkömmlichen Alarmlisten ausgeblendet. Ebenfalls werden zu diesen Alarmen gehörende Funktionen, wie etwa akustische Meldungen, unterdrückt. Die Alarmunterdrückung bewirkt somit eine Entlastung des Bedieners in Situationen mit außergewöhnlich hohem Alarmaufkommen.

Die Auswahl und der Befehl für die Unterdrückung eines oder mehrerer Alarme kann durch befugte Bediener direkt in der Alarmmeldeliste erfolgen. Im Zuge des Befehls muss die Dauer der Alarmunterdrückung sowie ein Grund für die Unterdrückung angegeben werden. Entsprechende Gründe für die Alarmunterdrückung können im Engineering frei definiert werden.

Unterdrückte Alarme können in einer entsprechenden Alarmliste, analog zur regulären Alarmliste, eingesehen werden. Die Bewertung des Alarmzustands (fortlaufendes Unter/-Überschreiten von Schwellwerten, allfällige Quittierung durch einen befugten Bediener) läuft während aktiver Unterdrückung weiter. Der befugte Bediener kann die Unterdrückung von Alarmen vorzeitig auflösen. Nach abgelaufener Dauer der Unterdrückungszeit geht ein Alarm automatisch wieder in die reguläre Alarmbehandlung über.

Die Aktivierung/Deaktivierung der Alarmunterdrückung wird lückenlos in der Ereignisliste (Sequence of Events) protokolliert. Das System gibt Auskunft über die Anzahl der aktuell unterdrückten Alarme. Der Zustand der Alarmunterdrückung wird im gesamten Server-Client Netzwerk konsistent verteilt. Das heißt, unterdrückte Alarme werden auf sämtlichen Clients als solche angezeigt.

Export von Alarmdaten

Ein Konsistenter Export und Speicherung von Alarmdaten in einer externen SQL-Datenbank muss möglich sein, sowie die Rückführung ins System und deren Wiederverwendung.

Es besteht die Möglichkeit den Export von Alarmdaten anzupassen und eine Filterung für die zu exportierenden Daten durchzuführen. Es bestehen weitrechende Filtermöglichkeiten wie Zeitfilter, Filterung nach Alarmklassen,-gruppen und -bereiche, ...

3.8. Alarmstatistik

Das System muss die Möglichkeit bieten, umfassende Analysen der Alarmhistorie einer Anlage durchzuführen. Alarminformationen müssen deshalb in einer ODBC-kompatiblen (Open Database

Connectivity) Datenbank abgelegt werden. Abhängig von der ausgewählten Systemtopologie müssen die jeweiligen Dateneinträge aus den verschiedenen Stationen (Servern) gesammelt werden und in einen einzelnen Alarmeintrag integriert werden. Die Alarmanalyse-Software kann direkt in die Runtime integriert sein oder optional als separates Softwaretool zur Verfügung stehen.

Die Analyse muss zumindest die „häufigsten", „kürzesten" und „längsten" Alarme zeigen. Die Anzahl der anzuzeigenden Alarme (Filterergebnisse) muss frei anpassbar sein. Eine Analyse der Gesamt-Alarmdauer der Anlage sowie der Anlagekomponenten muss möglich sein.

Außerdem muss der Alarmanalyse-Mechanismus die Bestimmung der Netto-Ausfalldauer einer Prozesskomponente ermöglichen. Dazu muss eine Unterscheidung zwischen geplanten (z.B. Wartung) und ungeplanten Stillstands Zeiten möglich sein. Um dies zu erreichen, muss das System eine Scheduler-Komponente enthalten, die diese Informationen liefern kann.

Die Analyse-Software muss frei anpassbare Filter anbieten. Eine Ergebnisanzeige in tabellarischem Format sowie in grafischen Formaten (z.B. Balken- und Tortendiagramme) muss möglich sein.

3.9. Ereignisliste, Betriebsjournal, Sequence of Events list, Audit Trail

Eine Ereignisliste ist notwendig für die chronologische Anzeige aller Statuswerte, spezifischer Systemereignisse und spezifischer Mensch-Maschine-Interaktion. Die Informationen zu den Änderungsereignissen müssen umfassen: Datum, Zeit, Beschreibungstext, Statustext und Variablenstatus.

Die Zeitspalte muss Zeitstempel mit Millisekunden- und Mikrosekundenauflösung zulassen.

Bei Kommunikationsprotokollen mit externer Zeitstempelung muss es möglich sein, beide Zeitstempel, extern und intern, in jeweils einer Zeitspalte darzustellen.

Alle Bedieneraktivitäten (z.B. Eingabe eines Sollwerts) müssen zurück verfolgbar sein. Außerdem muss das Einloggen und Ausloggen von Benutzern dokumentiert werden. Es muss die Möglichkeit geben, bei Bedarf userspezifische Informationen aus Sicherheitsgründen auszublenden.

Optional muss es möglich sein, das System so zu konfigurieren, dass alle Rezeptänderungen, alle Rezeptstornierungen auf der Steuerung sowie alle Änderungen an Archivdaten protokolliert werden.

Ein Filter für das selektive Anzeigen von Events muss online anwendbar sein (Runtime, HMI). Es muss eine Möglichkeit geben, spezifische Filtereinstellungen zu speichern und zu laden, als Unterstützung für den Bediener.

Kategorien zur Anzeige bestimmter Informationen müssen definiert werden können. Somit können Einträge z.B. nur für signaturrelevante Wertänderungen von kritischen Parametern für das Audit Trail angezeigt werden.

Für jeden Event muss es möglich sein, einen Kommentar von mindestens 80 Zeichen einzugeben. Das System muss für den Anwender die Möglichkeit bieten aus einer vor selektierten (zentral verwalteten) Liste von Kommentaren auszuwählen, um Fehleingaben zu vermeiden und die Auswertung zu erleichtern.

Das System muss ein kontinuierliches Ausdrucken des Betriebsjournals erlauben. Alternativ muss es möglich sein, einen spezifischen (gefilterten) Zustand des Journals für ein späteres Ausdrucken zu speichern.

Die Events, die durch Filtern oder Stornierung aus dem Betriebsjournal entfernt worden sind, müssen in einem chronologischen Ereignisarchiv erhalten bleiben, um sie für eine spätere Auswertung zu bewahren.

Export von Ereignisdaten

Konsistenter Export und Speicherung von Eventdaten in einer externe SQL-Datenbank muss möglich sein, sowie die Rückführung ins System und deren Wiederverwendung.

Es besteht die Möglichkeit den Export von Ereignisdaten anzupassen und eine Filterung für die zu exportierenden Daten durchzuführen. Es bestehen weitrechende Filtermöglichkeiten wie Zeitfilter, Filterung nach Ereignisklassen,-gruppen und -bereiche, ...

Alle Events müssen nach SQL ausgelagert werden können.

3.10. Reports

Das System muss das Erstellen von Reports unterstützen. Die Reports müssen durch den Benutzer definierbar sein. Die Reports müssen die Anzeige von beliebigen historischen und aktuellen (zum Zeitpunkt der Reporterstellung) Prozesswerten erlauben. Insbesondere muss die Erstellung von betrieblichen Statistiken und die Berechnung von KPIs (Key Performance Indicators, z.B. Stromqualität) möglich sein. Um dies zu erreichen, muss der Reporting-Mechanismus Zugriff auf alle Archivdaten, zentral und dezentral, sowie auf Online-Daten (Runtime-Daten) haben. Ein zugehöriges Report-Einrichtungswerkzeug (z.B. Wizard, Report-Verwaltungssoftware) muss die Erstellung von Reports leiten und unterstützen sowie die verfügbaren Datenelemente (Variablen, Archive, Online-Daten, logische Gruppierungen) für die Informationsverdichtung anzeigen. Neben den Prozesswerten soll das Berichtswesen auch Statusinformationen zu den jeweiligen Datenpunkten (Variablen) auswerten können. Darauf aufbauend können z.B. Zeiträume mit auffälliger Datenqualität in den jeweiligen Berichten hervorgehoben werden.

Um die Erstellung von aussagekräftigen Betriebsstatistiken zu unterstützen, muss der Reporting-Mechanismus dazu fähig sein, die existierenden Konfigurationselemente, wie z.B. Alarmmodelle (Klassen, Gruppen und Bereiche), Zeitpläne, Rezepte, Betriebsjournal oder Anlagenelementgruppen, mit einzuschließen. Die Überlagerung der Prozessdaten mit entsprechenden Schichtdaten der Anlage oder Liegenschaft muss unterstützt werden. Außerdem muss der Status eines jeden Prozesswerts – wie durch die jeweilige Variablenstatus-Repräsentation definiert – zugänglich sein, um den Kommunikationsstatus auflösen zu können. Dazu muss das Reporting-System einen umfassenden historischen Vergleich sowie Echtzeitvergleich (Benchmarking) für verschiedene Teile der Anlage (physisch und logisch) unterstützen. Außerdem muss das Report-Einrichtungswerkzeug grundlegende Vorlagen für gängige, branchenspezifische Reports anbieten (z.B. Statistiken zur Stromqualität).

Um die Nachvollziehbarkeit, der angezeigten Daten, zu ermöglichen ist es notwendig die angewendeten Filterungen anzuzeigen.

Die Erstellung von Reports muss – abhängig von dem Datenvolumen (z.B. Extraktion eines historischen Zeitrahmens) und der Netzwerkgeschwindigkeit – eine angemessene Zeitdauer in Anspruch nehmen. Das Reporting-System muss vorhandene Optimierungsmöglichkeiten nutzen, wie z.B. Datenbankindizierung, gespeicherte Prozeduren oder benutzerdefinierte Funktionen.

Die Verwaltung (Einrichtung, Wartung, Scheduling, Anordnung und Zugriff) von Reports (Reportdefinitionen) muss auf strukturierte Art und Weise organisiert sein. Es muss eine Benutzerverwaltung geben, um unautorisierten Zugriff zu vermeiden. Reports sollten nach bestimmten Themen oder Adressaten verwaltet werden (z.B. vergleichbar mit einer Datei-

Ordnerstruktur). Die Strukturierung und Organisation der Reportdefinitionen müssen durch einen autorisierten Benutzer frei definierbar sein.

Alarm- und Ereignisreport

Es besteht die Möglichkeit mehrsprachige Alarm und Ereignisreports zu erstellen wobei die Sprachumschaltung, durch den Nutzer des entsprechende Reports durchgeführt werden kann.

3.11. Notizbuch

Eine Notizbuchfunktion muss die Erstellung von Textnotizen erlauben, wie z.B. Events, Telefonnummern, Arbeitsanweisungen etc. Je nach Berechtigungsebene des Benutzers muss der Text sichtbar und editierbar sein. Für voll berechtigte Benutzer muss es möglich sein, Texte während der Runtime zu erstellen, zu speichern, zu editieren und aufzurufen.

3.12. Wartungsmanagement

Das System muss eine integrierte Funktionalität zur Erfassung von Anlagen- oder Maschineneinheiten bieten, welche für Wartungen überwacht werden müssen. Die Erfassung muss anhand einzelner Einheiten (z.B. Geräte) erfolgen können. Die Angabe allgemeiner Stammdaten und Beschreibungen wie Gerätename, Typ, Kommentare und Grundeinstellungen für die Auslösung von Wartungen ist erforderlich.

Die Signalisierung von Wartungsaufgaben muss anhand der Parameter Betriebsdauer (seit letzter Wartung), Produktionstakt oder Schaltungen (Zählung des Schaltspiels) erfolgen können. Außerdem muss eine Wartung direkt ausgelöst werden können, beispielsweise durch eine entsprechende Trigger-Variable. Dies wird benötigt, um bei festgestellten Abweichungen im Zuge von externen Analysen Wartungen direkt anfordern zu können.

Manuelle Wartungsaufgaben dürfen unter Umständen nur durch Personen mit geeigneter Autorisierung durchgeführt werden. Daher darf das Wartungssystem eine formelle Wartungsdurchführung (Wartungsabschluss) nur bei Erfüllung einer definierten (optionalen) Benutzerebene zulassen.

Die Verwaltung der Stammdaten von Geräten sowie mögliche Zuordnungen zu den im Projekt befindlichen Anlagengruppen muss über eine strukturierte Baumansicht erfolgen können. Die Anzeige fälliger Wartungen kann in Listenform erfolgen. Zur gezielten Darstellung von anstehenden Wartungsaufgaben muss eine gefilterte Darstellung der Wartungen durch Angabe einer Zeitspanne oder anhand einer Selektion der Ebenen im Anlagenmodell unterstützt werden.

Änderungen im Wartungsbereich, wie etwa neu anstehende Wartungen oder durchgeführte Wartungen, müssen protokolliert werden.

Das System muss die kombinierte Darstellung von wartungsspezifischer Information und den dazugehörigen Auswertungen, wie etwa Trenddarstellungen, Meldungslisten oder Alarmlisten unterstützen.

3.13. Modellbasierte Prognose

Das System kann mittels eines mathematischen Prognosemodells von historischen Wertverläufen sowie aktuellen Werten auf prognostizierte, zukünftige Werteverläufe schließen. Das System muss über Mechanismen zur Unterstützung bei der Ermittlung einer geeigneten Prognosefunktion, wie etwa der Periodizität des Werteverlaufs, verfügen.

Der prognostizierte Werteverlauf kann zur Erstellung von Berichten verwendet werden, wobei beispielsweise der historische Verlauf durch die Prognose ergänzt wird. Des Weiteren können einzelne Werte aus dem prognostizierten Verlauf in Echtzeit im Runtime System in Form von Variablenwerten zur Verfügung gestellt werden. Diese werden dementsprechend kontinuierlich vorberechnet.

Der prognostizierte Werteverlauf muss in sekundengenauer Auflösung zur Verfügung stehen.

3.14. Benachrichtigungsdienst

Das System muss einen ereignisgesteuerten Benachrichtigungsdienst bieten. Wen ein vordefinierter Event auftritt, muss eine Nachricht selektiv an eine oder mehrere Personen geschickt werden. Mögliche Übertragungskanäle für diese Nachrichten sind:

- E-Mail (über Microsoft Outlook oder über SMTP). Das System muss für das SMTP Protokoll eine TLS 1.3 Verschlüsselung unterstützen.
- SMS (Short Message Service) an ein Mobiltelefon
- Telefon (Voice Mail, Text to Speech)
- VoIP: (Voice-over-IP) Audio Datei Wiedergabe
- Text-To-Speech Wiedergabe

Der Benachrichtigungsdienst muss komplett in das System integriert sein, um volle Kontrolle sicherzustellen und eine Manipulation durch Unberechtigte (Personen, externe Programme etc.) zu vermeiden.

Die Empfänger der Nachrichten müssen in Gruppen organisiert sein. Wenn eine Nachricht nicht an eine bestimmte Person übermittelt werden kann, muss es eine Option geben, automatisch eine geeignete Ersatzperson zu informieren. Dies muss durch die Verwaltung von Empfangsbestätigungen unterstützt werden. Nach dem Versenden der Nachricht muss ein Eintrag in dem Betriebsjournal erstellt werden.

Es muss möglich sein, den Benachrichtigungsdienst mit einem Arbeitsschichten-Zeitplan zu verknüpfen, um sicherzustellen, dass Nachrichten nur an Personen gesendet werden, die gerade im Dienst sind.

Bei Versand einer E-Mail muss das System den Anhang beliebiger Dateien unterstützen.

3.15. Rezeptverwaltung

Die Verwaltung von Rezepten muss sowohl in der Entwicklungs- als auch in der Runtime-Umgebung möglich sein. Unabhängig davon, ob ein Rezept in der Projektierungs- oder Runtime-Umgebung erstellt bzw. editiert wird, muss ein bidirektionaler Datentransfer-Mechanismus verfügbar sein, um selektiv synchronisieren zu können.

Rezeptmanagement-Operationen müssen unter strikter Beachtung von Benutzerberechtigungen erfolgen (siehe Abschnitt 1.9). Jede Manipulation oder Rezeptnutzung muss im Betriebsjournal (SOE oder Audit Trail) protokolliert werden.

Das System muss ein Minimum der folgenden Funktionen bezüglich der Rezeptverwaltung bieten:

- Freie Erstellung und Definition der Datensammlung (Variablen) für ein Rezept

- Sofortige Erstellung eines Prozessabbilds (Datensicherung) durch Auslesen tatsächlicher Prozesswerte (Live-Werte) in die definierten Variablen eines Rezepts.

- Sofortige Anwendung eines Rezepts durch Setzen von Prozesswerten (Live-Werte) auf das definierte Niveau

- Duplizierung eines Rezepts für die Weiterbearbeitung

- Export und Import von Rezeptdefinitionen unter Einsatz generischer Datenformate (ASCII, XML)

- Validierung von Rezepten im Echtzeit Betrieb

- Inkludierte Versionierung für die Erstellung von Vorlagenrezepte oder zur Absicherung vor Manipulation.

- Konfigurierbare eSignature bis zu drei Stufen für Rezeptwertänderungen inkl. vollständiger Protokollierung im Audit Trail

Außerdem muss es möglich sein, Rezepte in Gruppen zu organisieren, um das Gesamt-Datenvolumen zu strukturieren und um generische Operationen anzuwenden.

Es muss möglich sein, Rezepte automatisch anzuwenden, also gemeinsam mit einer Änderung des Betriebsmodus einer Prozesskomponente. Das Rezeptverwaltungssystem muss die Möglichkeit bieten, rezeptbezogene Operationen (erstellen, kopieren, aktivieren, exportieren etc.) zu steuern und zu automatisieren. Rezeptbezogene Operationen werden meistens mit visuellen Elementen verknüpft werden (z.B. Schaltflächen) oder durch Steuerprogramme (Automatisierungsskripts, Soft-SPS-Aufgaben) ausgelöst werden.

Das Projektierungssystem muss Vorlagen für die Anzeige von Rezepten bieten (z.B. im Tabellenformat). Außerdem muss die Rezeptverwaltung einen einfachen Soll-/Istwert-Vergleich bieten, um dem Benutzer einen unmittelbaren Vergleich von Runtime-Werten mit Rezeptwerten zu ermöglichen. Wenn die Werte voneinander abweichen, muss dies grafisch hervorgehoben werden.

Es muss möglich sein, die Normal-Position von Schalter-Variablen über Rezepte zu ändern.

3.16. Fernwartung und -betrieb

Durch den dezentralisierten Charakter von Industrie 4.0-Installationen müssen Möglichkeiten für Fernwartung und -betrieb unterstützt werden. Darum muss es eine Möglichkeit zum externen Login in das System geben. Unabhängig von der tatsächlichen Zugangsmethode muss der Zugriff auf das Prozessleitsystem (Visualisierungsclient) durch die in das System integrierte Benutzerverwaltung gesteuert werden (siehe Abschnitt 1.9).

In Abschnitt 12 werden die benötigten Varianten des Fernzugriffs auf das System aufgelistet.

3.17. Prozesssimulation

Das System muss die integrierte Simulation von Prozessverhalten unterstützen. Im Allgemeinen müssen prozess- und projektbezogene Simulationsoptionen so gestaltet sein, dass folgendes erleichtert wird:

- Entwicklungstests, um die Funktionalität beliebiger projektierter Komponenten (Anwendungsentwicklung und Debugging) zu verifizieren

- Teilweise Simulation (Virtualisierung) von Prozesskomponenten, im Fall von Teilinbetriebnahme-Szenarien (teilweise Verbindung zur Anlage, virtueller Anlauf)

- Aufzeichnung von Prozessverhalten und Wiederverwendung in simulationsbasierten Tests

- Volle Simulation der Systemumgebung, z.B. für Schulungszwecke oder funktionale Vorführungen

Das System muss diese Anforderungen erfüllen, indem es simulationsbezogene Funktionen auf verschiedenen Ebenen anbietet, z.B.

- Modellierung des Verhaltens einzelner Variablen, über einfache simulationsbezogene Parameter

- Selektive Anbindung und Abtrennung von Variablen an/von externen Stationen (Dritthersteller)

- Einsatz von Soft-SPS-basierten Funktionen und auf IEC 61131-3 basierender Programmierung für die Simulation von komplexeren Systeminteraktionen und -verhalten

Alle Definitionen (Parametrisierung, Programmierung etc.), die für die Simulationsfunktionalität benötigt werden, müssen in einer modularen Weise organisiert sein und dürfen die Konzeption des originalen Entwicklungsprojekts nicht beeinflussen. Es darf also beispielsweise kein zusätzlicher Entwicklungsaufwand (für die Erstellung und Wartung von zusätzlichen Schnittstellen) benötigt werden, um eine selektive Überführung von der physischen Infrastruktur in einen Simulationskontext herzustellen.

Generell müssen alle gängigen Entwicklungsmechanismen auf die Erstellung und Wartung von Simulationskomponenten anwendbar sein (Projektstrukturierung, Wiederverwendung von teilweisen Definitionen, Vorlagenerstellung etc.).

3.18. Prozess-Rekorder

Ein im System integrierter Prozess-Rekorder muss die Möglichkeit bieten, Prozessdaten der produktiven Runtime aufzuzeichnen. Zu einem späteren Zeitpunkt können diese aufgezeichneten Daten in der Runtime auf einem Projektsimulations-Client erneut abgespielt werden.

In der Regel besteht ein solches Modul aus zwei Komponenten:

1. Aufzeichnung von Prozessabläufen

Bei der Projektierung im System werden Variablen für die Aufzeichnung aktiviert. Im produktiven Prozessablauf in der Runtime werden diese Variablen aufgezeichnet.

Der Aufzeichnungsort der Prozessablaufdaten soll flexibel gewählt werden können.

2. Wiedergabe der Aufzeichnung

Die aufgezeichneten Daten werden in der Runtime in Form einer Prozess Simulation visualisiert. Die Wiedergabe der aufgezeichneten Werte wird in bestehenden Prozess Bildern visualisiert. Die Steuerung der Wiedergabe erfolgt über ein eigenes Bedien-Menü. Die Wiedergabe wird über ein separates Betriebsmenü gesteuert und bietet eine langsame und schnelle Wiedergabegeschwindigkeit. Es muss möglich sein, die Wiedergabegeschwindigkeit in diskreten Schritten zwischen dem 1/8-fachen (langsamer als Echtzeit) und dem 8-fachen (schneller als Echtzeit) einzustellen.

Projektänderungen müssen bei der Wiedergabe von Aufzeichnungen berücksichtigt werden.

3.19. Last-Management

Ein System muss angeboten werden, um den Einsatz von Elektrizität als Energiequelle zu verwalten. Einsparungspotenzial muss sich durch das Vermeiden teurer Leistungsspitzen ergeben. Dies muss durch einen Lastabwurf von schaltbaren Geräten und eine Einspeisung von eigenen Generatoren erreicht werden. Das System muss die mittlere Leistung einer Messperiode vorhersehen (Prognose). So lässt sich eine drohende Überschreitung der festgelegten Bezugsleistungsgrenze rechtzeitig erkennen und man kann entsprechend regelnd eingreifen. Das System muss dazu fähig sein, sowohl in geschlossenen (automatischen) als auch in offenen Schleifenzuständen zu arbeiten.

3.20. SAP-Anbindung

Das SCADA System muss einen bidirektionalen Datenaustausch mit einem SAP-System unterstützen. Dabei sollen Standardschnittstellen von SAP genutzt werden, um Inhalte von Wartungsmeldungen und Messbelegen etc. auszutauschen. Die Systeme unterstützen das SAP Netweaver Interface.

3.21. Diagnose des Systems

Das SCADA System muss für eine externe Diagnose automatisch im Hintergrund Logging-Informationen protokollieren. Für die Auswertung dieser LOG-Dateien muss ein zugehöriges Programm zur Verfügung stehen, das unabhängig vom System eingesetzt werden kann. Eine Remote Diagnose über Netzwerkverbindung muss unterstützt werden.

4. HMI

4.1. Prozessbilder

Prozessbilder müssen frei aus einer Menge an Illustrationen und Steuerkomponenten zusammengesetzt werden. Sie müssen hauptsächlich aus einem statischen Hintergrund bestehen. Die jeweiligen Variablen (Nachrichten, Werte, Alarme etc.) werden benutzt, um ein dynamisches Verhalten auf den Prozessbildern zu erzeugen. Die Erzeugung eines Bildes (Bildumschaltung) muss weniger als 2 Sekunden in Anspruch nehmen. Es muss möglich sein, mindestens 500 dynamische Elemente auf einem Bild zu verwalten.

Für Systeme, die in mehrere hierarchische Ebenen gegliedert sind, müssen dieselben Bilder auf jeder Ebene verwendbar sein. In Übereinstimmung mit der generellen Organisation des Datenflusses müssen Dateneinträge eindeutig verwaltet werden.

Änderungen des Prozessstatus werden angezeigt über:

- Änderung eines einzelnen Zeichens
- Symboländerung
- Farbänderung
- Einblenden von Text
- Automatische Bildumschaltung
- Beliebige Kombinationen der genannten Optionen

Werte werden dargestellt als:

- Numerische Werte
- Bargrafen
- Anzeige von Messinstrumenten
- Drossel- und Steuerhebel
- Kurvendarstellung mit dedizierter Wertachse pro gemessenem Wert und Zoom
- Eine Anzeige von dezimalen, hexadezimalen, oktalen und binären Werten muss möglich sein

Zusätzlich werden die folgenden Wertzustände (Variablenverbindung zustandsabhängig) angezeigt werden, z.B. mithilfe von Piktogrammen:

- Fehlfunktion
- Spontan
- Abgeschaltet
- Ersatzwert
- Selektierte
- Gesperrt/Verriegelt

Erweiterte grafische Elemente, wie z.B. WPF-Elemente (Windows® Presentation Foundation, Abschnitt 4.12) sowie die Einbettung von .NET Elementen (Abschnitt 4.11) müssen unterstützt werden.

Es muss die Möglichkeit geben, Bildvorlagen zu erstellen, um Standardaufgaben zu unterstützen und ein konsistentes Visualisierungsdesign zu erleichtern. Usability-Funktionen, wie z.B. die Ausrichtung von Elementen, Positionierung und Feinanpassung anhand von Gitterlinien sollten auch angeboten werden.

In einem Prozessbild muss der aktuelle Onlinestatus jeder individuellen Variable angezeigt werden. Darum muss es möglich sein, dass der Bediener auf einen Blick visuell identifizieren kann, ob eine Variable von ihrer Datenquelle getrennt ist. Außerdem muss es möglich sein, die Datenquelle jeder Variable direkt auf dem HMI-Bildschirm anzuzeigen.

Das System muss eine Symbolbibliothek anbieten, die über speziell definierte Symbole erweitert werden kann. Es muss möglich sein, Symbole einzufügen und zu vererben. Wenn ein vererbtes Symbol in der Symbolbibliothek verändert wird, müssen alle Verknüpfungen ebenfalls automatisch aktualisiert werden. Alle Symbole müssen global verfügbar sein.

Es muss möglich sein, Symbole zu erstellen, die aus mehreren grafischen Elementen bestehen und die mit spezifischen Variablen verknüpft sind, um grafische Attribute anzupassen (Sichtbarkeit, Farbe, Orientierung etc.). Wenn Symbole eingefügt oder vererbt werden, muss eine Substitution von Variablen unterstützt werden. Wenn also ein grafisches Symbol wiederverwendet wird, muss ein einfacher Mechanismus zur Variablenersetzung verfügbar sein.

Das System muss den Import von Vektorgrafiken im DXF- und WMF-Format unterstützen. Zusätzlich müssen sie verarbeitet oder als unabhängige Vektorgrafiken behandelt werden. Gängige Rastergrafikformate wie BMP, JPG, PNG, TIF und GIF müssen unterstützt werden.

Das System muss die Möglichkeit bieten, festzustellen, welches Bild der Visualisierung aktiv gezeigt wird.

Die Benutzerverwaltung (siehe Abschnitt 1.9) muss auf alle Kontrollelemente eines Bildes anwendbar sein. Somit müssen Elemente abhängig von dem eingeloggten Benutzer oder einem frei anpassbaren Systemstatus gesperrt werden. Um solche Sperrbedingungen flexibel einstellen zu können, müssen Binärwerte beliebig kombinierbar sein.

4.2. Visualisierung von umfangreichen Schemata

Das System muss dazu fähig sein, auch umfangreiche Schemata darzustellen, wie z.B. Windfarmen, Stockwerkspläne von großflächigen Anlageninstallationen, Einlinienschaltbilder, geografische Karten oder Layouts von Produktionsanlagen. Es muss möglich sein, während der Runtime dynamisch in dieses Übersichtsbild hineinzuzoomen. Außerdem muss das System die Möglichkeit bieten, Elemente auf diesem Schema anzuordnen und die Detailstufe für den Bediener je nach Zoomstufe anzupassen (sogenanntes „Decluttering“). Somit muss es möglich sein, Visualisierungselemente für eine detaillierte Informationsauflösung sukzessive durch konzentriertere Repräsentationen zu ersetzen, wenn der Bediener heraus zoomt (in Richtung der Gesamtübersicht). Zusätzlich muss es möglich sein, auf dem umfangreichen Schema zu navigieren, sobald man sich in einem Nahbereich befindet. Außerdem muss es Steuermöglichkeiten für die Zoomebenen geben.

4.3. 3D-Integration

Das System sollte in der Lage sein unkomplizierte 3D-Dateien (GLB) aus einem CAD-Programm mit Projektierungen aus dem System zu verknüpfen. Die Struktur eines 3D-Modells wird übernommen und in einer Vorschau visualisiert.

In einem Konfigurator können 3D-GLB-Dateien mit Projektierungen in einer grafischen Benutzeroberfläche verknüpft werden.

Der ausgewählten Baugruppe oder einem einzelnen Objekt können:

- Eine oder mehrere Variable zugeordnet werden. Ist eine Variable verknüpft, werden die Sichtbarkeits-, Blink- und Farbeinstellungen von der Variable übernommen.

- Eine oder mehrere Funktion zugeordnet werden. Ist eine Funktion verknüpft, wird die Funktion durch klicken auf das Objekt in der Runtime ausgelöst.

- Einer Variable zusätzlich eine Kameraposition zugeordnet werden. Ist eine Kameraposition verknüpft, wird die Position aufgerufen, wenn der Grenzwert der verknüpften Variable verletzt wird.

Darstellung der 3D-Projektierung zur Runtime in einem Prozess Bild.

- Freie Navigation im 3D-Modell: Die Darstellung kann verschoben, gedreht, vergrößert oder verkleinert werden.

- Ausführung von Funktionen im 3D-Modell: Durch Klick auf ein Objekt oder eine Baugruppe kann eine projektierte Funktion ausgeführt werden.

Aufruf des 3D-Modells in einem definierten Blickwinkel: Durch Setzen eines Wertes einer "Kameravariable" kann das 3D-Modell mit Ansichten einer projektierten Position visualisiert werden.

Visualisierung einer Grenzwertverletzung: Bei Verletzung eines Grenzwertes kann ein Objekt oder eine Baugruppe im 3D-Modell eingefärbt oder blinkend dargestellt werden.

- Objekte oder Baugruppen können sichtbar oder unsichtbar geschalten werden.

Es besteht die Möglichkeit die 3D-Integartion in einer Web-basierten Lösung zu nutzen (Smart Client)

4.4. Automatic Line Coloring (automatische Linieneinfärbung)

Eine automatische Linien-Einfärbung von Leitungen oder Prozess Elementen dient zur einfachen automatischen Dynamisierung von Leitungen in der Verfahrenstechnik (für Medien) sowie in der elektrischen Energieverteilung (für Strom). Damit lassen sich die prozessgesteuerten Einfärbungen von Netzen jeglicher Art einfach realisieren.

Aufgrund der Projektierung im Prozess Bild wird die Leitungsstruktur einschließlich ihrer angeschlossenen verfahrenstechnischen Elemente (z. B. Tanks und Ventile, oder Generatoren, Schalter und Verbraucher) intern als Modell nachgebildet und der Medienfluss während der Runtime dargestellt.

Um bildübergreifende Modelle zu ermöglichen, ist die gesamte Projektierung sowie Konfiguration immer projektweit. Es ergibt sich somit pro Projekt ein gesamtes Modell, das zur Berechnung der Leitungszustände und schließlich zur Färbung der einzelnen Leitungen herangezogen wird.

4.5. Trendkurven

Das Visualisierungssystem muss eine Komponente zur Trendanzeige für Online-Werte sowie Archivwerte anbieten. Die Anzahl von Kurven darf nicht beschränkt sein. Es muss möglich sein, die Farbanzeige der Kurven, die Aktualisierungsintervalle sowie die Werte und Positionen der Y-Achsen (obere und untere Anzeigegrenzen) frei anzupassen. Das Zeitwertmuster (Datenformat) und die gewünschte Zeitperiode der Anzeige muss für den Entwicklungstechniker sowie den eingeloggten Benutzer frei definierbar sein. Es muss möglich sein, die aktuellen Anzeigeeinstellungen eines Benutzers individuell abzuspeichern.

Es sollte eine Option zur Skalierung von Werten für die Anzeigeausrichtung geben. Die Trendfunktion muss aus einer Zoom-Funktion bestehen. Ein Cursor muss verfügbar sein, um Kurvenwerte im Detail untersuchen zu können. Die Sichtbarkeit einzelner Kurven muss zur Laufzeit mittels Variablenwert gesteuert werden können. Es muss die Option geben, selektierte Kurven zur Laufzeit grafisch hervorzuheben.

Werte müssen grundsätzlich in einem X/Y-Diagramm (Zeit/Wert-Diagramm) angezeigt werden. Außerdem muss es möglich sein, zwei Trendkurven aus verschiedenen Zeitspannen übereinander zu legen, sodass charakteristische Trends verglichen werden können (z.B. tägliche, monatliche oder saisonale Unterschiede).

Die Anzeige der Werte muss sowohl als Trendkurve, als auch als Gantt Anzeige unterstützt werden. Beide Darstellungsformen müssen im gleichen Zeitbereich darstellbar sein, damit Zusammenhänge zwischen Zuständen (Gantt Anzeige) und Werten (Trendkurve) für den Anwender ersichtlich sind. Es muss möglich sein, die Diagramme direkt über das Visualisierungsbild auszudrucken oder auch die aktuell angezeigten Daten in einem Maschinen lesbaren Format zu exportieren (z.B. *.txt)

Für Bildelemente, die aktuelle Prozesswerte dynamisch anzeigen, muss es die Möglichkeit der direkten Umschaltung auf eine Trenddarstellung des entsprechenden Variablenwertes geben. Des

Weiteren soll das System für anlassfallbezogene Untersuchungen das Einfügen von Variablenwerten in eine Trenddarstellung anhand der verbundenen Anzeigeelemente erlauben. Die Interaktion erfolgt idealerweise durch Einzel- oder Mehrfachselektion von Anzeigeelementen (z.B. durch Lasso-Funktion), und direktes „Drag & Drop" der selektierten Elemente (sprich Variablendaten) in die Trenddarstellung. Somit kann der Bediener auf intuitive Art und Weise bestimmte Anzeigewerte mittels Anzeigeelement in die Trenddarstellung „schieben".

Die Sichtbarkeit von Achsen bzw. Kurven muss über Variablen gesteuert werden können. Die grafische Ausprägung von Trenddiagrammen muss über zentral verwaltete Darstellungsstile vorgegeben werden können. Sowohl die Diagrammfläche als auch die Darstellungsachsen und die einzelnen Trendkurven müssen individuell über den jeweils zugeordneten Darstellungsstil definiert werden können. So kann ohne Eingriff in die funktionale Einstellung/Anordnung des Trendbildes die grafische Erscheinung adaptiert werden. Die grafische Ausprägung der Achsen muss auch zur Laufzeit durch Auswahl von Darstellungsstilen eingestellt werden können.

In der Visulisierung (Trendkurve) ist auf Anhieb ersichtlich, wenn eine Grenzwertverletzung geschehen ist. Dazu werden die Grenzwertverletzungen, durch farbliche Hervorhebung, deutlich ersichtlich gemacht. Dabei ist es möglich den jeweiligen Grenzwerten Farben zuzuordnen.

Definierte Grenzwerte können über einen Dialog ein- und ausgeblendet werden, um zu visualisieren, wie sich die Prozesswerte im Verglich zu den definierten Grenzwerten verhalten haben.

Es wird die Möglichkeit geboten zwei Datenpunkte (Positionen auf der Trendkurve) zu vergleichen, weiters sind beide Werte in der Visualisierung ersichtlich.

4.6. Integration von Videos

Das System muss eine Integration von Videos für die Online-Betriebsüberwachung oder für Offline-Service- und Wartungsarbeiten ermöglichen.

4.7. GIS Integration

Alle Arten von Leitungen, sei es nun für Strom, Gas, Wasser oder auch Informations- und Kommunikations-Technologie (IKT), müssen in einem System bezüglich ihrer Lage dokumentiert werden. Da diese Geoinformationen digital abgespeichert sind, sollten auch diese Daten mit den SCADA-Systemen in den Leitständen kombiniert werden. Im Grunde geht es darum, einer Leitung in ihrer geografisch richtigen Lage eine Zustandsinformation mitzugeben. In den meisten Fällen erfolgt dies durch die Einfärbung der Leitung, wobei jeder Farbe bestimmte Informationen zugeordnet sind. Auf diese Weise kann der Status von Anlagen überwacht werden, einschließlich der Informationen über die geografische Lage der Anlage. Das System muss grundlegend die dynamische Einblendung von Objekten auf Basis von Variablenwerten unterstützen.

Da insbesondere die Signalisierung von Meldungen oder Alarmen in einem maßstäblichen Lagebild enorme Vorteile bringt, muss das System die Darstellung eines Fehlerortes, dynamisch im Bild bei Auftreten eines entsprechenden Ereignisses, unterstützen. Die Anzeige erfolgt vorzugsweise durch ein Symbol, das in der Darstellung direkt am Fehlerort eingeblendet wird. Zusätzlich muss zur besseren visuellen Darstellung eine Fehlerortumgebung grafisch hervorgehoben werden können. Da der Anwender bei Betrachtung des Übersichtsbildes (Landkarte, Grundriss) die Ansicht in der Regel zoomen muss, ist eine zoomstufenabhängige Einblendung von Objekten, insbesondere der Fehlerortumgebung, notwendig.

- Das System unterstützt die Verwaltung von bis zu 800.000 GIS Objekten (GIS Marker), sowie die geeignete Darstellung einer entsprechenden Menge von GIS Markern für einen Anwendungsfall

- In Darstellungen von Versorgungsnetzen können Störungen – etwa Leitungsunterbrechungen oder Kurzschlüsse – durch einen räumlich exakt positionierten Fehlermarker dargestellt werden. Durch Interaktion mit dem grafischen Symbol zur Darstellung des Fehlerorts muss eine Meldung quittiert werden können. Information zur vorliegenden Störung kann direkt am Fehlersymbol angezeigt werden. Diese Anzeige umfasst insbesondere Zeitstempel der Störungsmeldung, Distanz auf der Versorgungsleitung, Kennung der Station, welche die Meldung generiert hat, sowie Meldungstext aus dem Alarmsystem.

- Ein statischer Beschreibungstext für ein GIS Objekt (GIS Marker, GIS Linie) kann mittels Tooltip in der Laufzeit Anwendung angezeigt werden.

- Ein permanenter Text kann für ein GIS Objekt (GIS-Marker) konfiguriert und in der Laufzeit Anwendung angezeigt werden.

- Es können kundenspezifische Graphiken für GIS Objekte (GIS-Marker) verwendet werden

- Es können Zustandsänderungen durch Anpassung der Hintergrundfarbe von GIS Objekten (GIS Marker) visualisiert werden.

- Auf GIS-Markierungselementen und Flächenelementen können bestimmte Funktionen verknüpft werden. Dazu gehören beispielsweise die Quittierung von Alarmen oder die Anzeige von Detailinformationen. Diese Funktionen können durch einen Mausklick auf das Element ausgelöst werden.

- Die GIS Integration des SCADA Systems muss die Auswahl verschiedener Online-Kartenanbieter sowohl im Engineering als auch zur Laufzeit ermöglichen. So kann zwischen Karten verschiedener Anbieter (z.b. Open Streetmap, Google, Bing, BAIDU), aber auch zwischen verschiedenen Darstellungsarten (Landkarte, Sattelitenansicht etc.) gewählt werden.

- Die GIS Integration des SCADA Systems unterstützt sowohl den Online-Abruf von Kartendaten als auch das lokale Caching in Fällen, in denen keine Internetverbindung zur Verfügung steht.

- Für den Massenimport von Markern, insbesondere in Landkartendarstellungen, muss das System den Import von „Keyhole Markup Language" codierten Dateien (KML, KMZ) unterstützen. Das System unterstützt hierbei auch den inkrementellen Import von geänderten XML-Dateien.

4.8. Repräsentation von HMI-Bildern und Prozesswerten im Intranet / Internet

Das System muss die Möglichkeit bieten, Prozessinformationen über Internet oder Intranet zu veröffentlichen, und dafür eine Verschlüsselung anbieten (siehe Abschnitt 9). Eine volle Prozessübersicht von jedem Arbeitsplatz mit einem Standard-Webbrowser muss möglich sein. Die Bildanzeige von webbasierten Visualisierungen muss identisch mit der Anzeige auf einer nativen Station sein. Außerdem muss die Funktionalität der webbasierten Visualisierung identisch mit jener der nativen Version sein. Es muss möglich sein, dass das Visualisierungsdesign nur einmal erstellt wird und dann auf die webbasierte Variante ohne Änderungen angewendet werden kann.

Zwei Versionen der Webdarstellung müssen angeboten werden:

- Reine Repräsentation: Der Benutzer kann die Anlage überwachen, aber nicht bedienen.

- Bedienung und Überwachung über das Web: Der Benutzer kann die Anlage bedienen und überwachen.

Eine detailliertere Erklärung der benötigten Fernzugriffsvarianten wird in Abschnitt 12 gegeben.

4.9. HTML5-basierte Web-Visualisierung

Eine Visualisierung mit HTML5 bietet eine unkomplizierte Darstellung wesentlicher Informationen in einer gewohnten Web-Umgebung. Wichtige Kennzahlen oder Prozessinformationen können einfach z.B. am Smartphone oder Tablet abgerufen werden. Die HTML5-Visualisierung muss in die Software integriert sein. Sie ist eine Ergänzung zur umfassenden Anwendung, mit Funktionalitäten, die Dashboards, KPIs etc. einfach zugänglich macht. HTML5-Screens erweitern das Portfolio der mobilen Zugriffsmöglichkeiten auf die wichtigsten Produktionsinformationen.

Die HTML5-Lösung ist browserunabhängig. Alle gängigen Browser, die HTML5 unterstützen, können verwendet werden. Für die Bereitstellung von webbasierten Visualisierungsinhalten ist weder zusätzlicher technischer Aufwand noch spezielles Web-Programmier-Know-how erforderlich. Die Visualisierungsinhalte für die Web-Visualisierung werden aus der gleichen technischen Datenbank generiert wie die regulären OT-Visualisierungsinhalte.

Minimale Anforderung an die darzustellenden Daten für Dashboard Applikationen:

- Variablenwerte
- Daten der Chronologischen Ereignisliste (CEL, SOE oder Audit Trail)
- Meldungen der Alarmmeldeliste (AML)
- Darstellung von Werte-Trends auf Basis von Archiven

Typische Funktionen, die in einer Dashboard-Anwendung verfügbar sein sollen und die in der Web-Visualisierung unterstützt werden sollen::

- Sitzungsbasierte Auslieferung von HTML5 Visualisierungsinhalten an HTML Web Clients.
- Darstellung grundlegender Visualisierungsinhalte, die im Engineering erstellt wurden.
- Weiterleitung von Prozessinformation, wie Variablenwerte, Alarm- oder Ereignis-Meldungen einer Runtime an einen oder mehrere HTML Web Clients.
- Unterstützung aktiver Bedienhandlungen, wie Sollwert setzen.
- Der Abruf von Trenddaten (aus Archiven) erfolgt in optimierter Form. Das System wählt je nach Darstellungssituation geeignete Wege zur Datenreduktion, ohne die grafische Auflösung am Web-Client zu beeinträchtigen.
- Mobiles, ortsunabhängiges Bedienen und Beobachten.
- Betrieb des HTML Web Servers auf einem separaten Rechner, wie zum Beispiel in einer DMZ möglich.
- Sichere Netzwerkkommunikation über HTTPS, basierend auf SSL-Zertifikaten.
- Schutz von sensiblen Visualisierungsbereichen oder Prozessen mittels Benutzerauthentifizierung und Unterstützung von Benutzerebenen.
- Die Dashboard Applikation unterstützt eine Authentifizierung eines Web-Clients mit erhöhter Sicherheit gegen die Benutzer-Authentifizierung und gegen Active Directory. Die Anmeldung erfolgt durch Eingabe des Benutzernamens und Passwort.

- Die Dashboard Applikation unterstützt die Wiedergabe einer Audio-Datei auf Basis eines frei definierbaren Ereignisses im Prozess.

- Erweiterte Funktionen, die in einer Dashboard-Anwendung verfügbar sein sollten - die Web-Visualisierung:Möglichkeit Kommentare und Alarmursachen zu setzen

- Alarmklassen, -gruppen und -bereiche werden unterstützt. und können zur Filterung der Meldungen der Alarmmeldeliste verwendet werden.

- Grundlegende Unterstützung der Befehlsverarbeitung für komplexe Energieprotokolle, wie z. B. Geräteauswahl, Ausgabe von einstufigen Befehlen, Anzeige und Entriegelung von anstehenden Verriegelungen.

- Anzeige von großen Prozessbildern in einem „World View", Zooming, Scrolling und decluttering ist möglich.

- Unterstützung der Befehlsgabe für komplexe Energieprotokolle, Verfügbare Befehlsarten und -methoden sollen sein:

    - Sichere Geräteauswahl,(Betriebsberechtigung)

    - Einstufige und zweistufige Befehle

    - Anzeige und Entriegelung von anstehenden Verriegelungen.

    - Sollwerteingabe

    - Manuelle Wertkorrektur

    - Forced Commands (für Notabschaltungen)

    - Aktivieren und Deaktivieren der Online-Kommunikation zur angeschlossenen Hardware ("Sperr"/"Freigabe"-Befehle)

    - "Revision"-Kennzeichnung für Geräte

    - Aktivierung des Substitutionswertes (Alternativ- oder Ersatzwert)

    - Impulsbefehl (konfigurierbarer, temporärer Impuls)

- Anzeige des Zustands in elektrischen Netzen (Umspannwerke, Netzleitstände), auf Basis der topologischen Berechnung
    - Dynamische Linieneinfärbung nach Zustand entsprechender Netzabschnitte

    - Anzeige der Versorgungssituation: Mehrfachversorgung, gesicherte Versorgung

    - Anzeige von Kurzschluss- und Erdschlussanzeigern (Fehlerortung)

4.10. Lokales HTML5-Web-Visualisierungs-Frontend

Die Web-Visualisierung soll ohne zusätzlichen technischen Aufwand möglich sein. Ein browserbasiertes Frontend soll Daten aus der Anwendung darstellen durch

- Dynamische grafische Elemente
- Unterstützung einer Modellierungsstruktur (Gerätemodellierung)
- Trend-Bildschirm
- Alarm screens
- Alarm-Bildschirme
- Ereignis-Bildschirme
- Einfache Prozessbildschirme
- Befehlsverarbeitung
- Steuerung von Rezepten zur chargenorientierten Produktion

  Mehrere Sitzungen müssen am Backend auch bei redundanten Servern unterstützt werden.

4.11. .NET Controls

Das System muss die Integration von .NET Elementen in der Visualisierung unterstützen. Es muss möglich sein, die jeweiligen Elemente in das Entwicklungsprojekt zu importieren und sie dort an die zugehörigen Datenschnittstellen anzubinden (Variablen, Funktionen).

4.12. WPF

Das System muss die Integration von WPF-Elementen (Windows® Presentation Foundation) in der Visualisierung unterstützen. Es muss möglich sein, die jeweiligen Elemente in das Entwicklungsprojekt zu importieren und sie dort an die zugehörigen Datenschnittstellen anzubinden (Variablen, Funktionen). Gemäß der entsprechenden Richtlinie basiert die Integration von WPF-Elementen auf XAML (Extensible Application Markup Language).

4.13. SVG

Das System unterstützt die Einbindung von SVG-Grafiken (Scalable Vector Graphics). Die Ausführung von eingebettetem Java Script Code muss unterstützt werden, um beispielsweise die Anzeige zu animieren.

4.14. Touch und Multi-Touch

Für ein intuitives und benutzerfreundliches HMI muss das System eine Bedienung über Touchscreen unterstützen. Außerdem müssen auf Multi-Touch basierende Interaktionen nativ unterstützt werden. Dazu gehören: Tap (Auswahl), Double Tap (Doppelklick), Press, Drag, Swipe, Flick (Quick Swipe), Two Finger Drag, Spread/Pinch (Vergrößern mit zwei Fingern), Zoom (Vergrößern) und Panning (Schwenken über eine Ansicht). Ein hohes Maß an Benutzbarkeit und Stabilität muss durch die Nutzung der nativen Touch- und Multi-Touch-Funktionen des Systems garantiert werden.

5. Downstream-Kommunikation

Das System muss eine Verbindung zu mehreren verschiedenen IEDs, SPS-Systemen, RTUs, Feldleitgeräten, Feldüberwachungsgeräten, Schutzrelais und Bussystemen unterstützen. Mehrfache Verbindungen müssen simultan laufen können und flexible Updates zu einem späteren Zeitpunkt

erlauben. Eine Verbindung muss über direkte Protokolltreiber erleichtert werden (proprietäre Kommunikationsbusse) oder über den besten verfügbaren Kommunikationsstandard.

Es muss möglich sein, Verbindungen während der Runtime jederzeit selektiv zu unterbrechen und wiederherzustellen sowie die zugehörigen Variablen in den Simulationsmodus umzuschalten.

Die Kommunikation zwischen Treiber und Steuerung muss – je nach Protokoll – entweder spontan oder im Polling-Modus erfolgen. Im spontanen Betrieb überträgt die Steuerung nur aktiv Daten an das Prozessleitsystem, wenn sich ein Wert ändert. Dadurch wird die Busauslastung reduziert. Im Polling-Betrieb muss es möglich sein, jedem Wert einen Hysterese Bereich zuzuweisen, um einen permanenten Transfer von flatternden numerischen Werten an das Prozessleitsystem zu verhindern. Der Zeitstempel muss auf der Steuerung als Echtzeitstempel gehandhabt werden. Alternativ kann er von dem Protokolltreiber auf dem Runtime-System definiert werden. Das System muss dazu fähig sein, den Echtzeitstempel konsistent an jeden Standort im Netzwerksystem zu verteilen. In der Runtime sollte das System in der Lage sein 2 Zeitstempel separat anzuzeigen und auszuwerten (interner und externer Zeitstempel).

Die Runtime-Informationen eines jeden (Kommunikations-)Treibers müssen in den Datenfluss des Systems voll integriert sein. Deswegen muss es möglich sein, die Kommunikationsleistung zu überwachen, den Kommunikationsstatus zu evaluieren und eine Evaluierung der Verbindungsqualität durchzuführen (Statistiken).

Wenn Datenpunkte redundant vorliegen (über unterschiedliche Treiber oder unterschiedliche Verbindungen), soll es möglich sein, diese als einen Datenpunkt im Leitsystem zu verwenden. Dabei soll automatisch weitergeschaltet werden können, wenn die Primärquelle schlechte Datenqualität liefert.

Das System muss über die folgenden Kommunikationsstandards kommunizieren können:

5.1. DNP3

- Das System muss als DNP3 Master agieren
- Das System kann als DNP3 Dual Endpoint Master agieren
- Das System unterstützt Serielle Kommunikation, TCP/IP-Kommunikation oder UDP Kommunikation
- Das System unterstützt Redundanz bei Serieller Kommunikation
- Das System kann Event Class Polls in konfigurierbaren Zyklus senden
- Das System unterstützt Echtzeitstempelung
- Das System unterstützt Unsolicited Responses
- Das System ist konform mit Subset Level 1, 2, 3 und 4 für Requests und Responses
- Das System unterstützt DNP3 File Transfer
- Das System unterstützt DNP3 Secure Authentication V2 und V5 nach IEEE Std™ 1815-2010 und IEEE Std™ 1815-2012 und IEC 62351-5
- Das System unterstützt TLS nach IEEE Std™ 1815-2012 und IEC 62351-3
- Das System unterstützt den online Import von Variablen nach einem Class 0 poll

- Das System unterstützt den offline Import von Variablen aus einem DNP3 XML Device Profile
- Das System unterstützt für Counters die Befehle, Freeze, Freeze no ack, Freeze and clear, Freeze and clear no ack
- Das System unterstützt für Binary Outputs und Analog Outputs die Befehle, Direct Operate No Ack, Direct Operate, Select und Operate (auto SBO)
- Das System unterstützt die konfigurierbare Reaktion auf internal Indication bits IIN 1.1, 1.2 und 1.3, IIN 1.4 und IIN 2.3
- Das System unterstützt die Konfiguration mehrerer Varianten für die Belegung von Double Point Werten
- Das System generiert erweiterte Kommunikationsstatistiken für den physical layer, data link layer, transport layer und application layer
- Das System bietet die Möglichkeit bestimmte Events doppelt zu übertragen, mit und ohne Zeitstempel.
- Das System bietet zur Laufzeit ein Monitoring für DNP3-Verbindungen

5.2. Modbus TCP und Modbus RTU

- Das System muss als Modbus Master agieren
- Das System muss die folgenden Modbus-Function Codes senden können: 1, 2, 3, 4, 5, 6, 16, 17
- Das System muss die „Sequence of Events“ von den folgenden IEDs lesen können: AREVA MiCOM P125/P126/P127, GE Multilin F650 und UR-Series, IEC NPx800, Schneider SEPAM
- Das System muss über hostnamen oder IP-Adresse kommunizieren können
- Das System muss einen konfigurierbaren TCP port unterstützen
- Das System muss mehrere Verbindungen unterstützen
- Das System muss Protokoll Konverter berücksichtigen, die für mehrere seriell angebundenen Modbus Slaves nur eine beschränkte Anzahl an TCP Verbindungen unterstützt ("Kanalbündelung")
- Das System muss das Senden von RTU-Frames über TCP/IP unterstützen.

5.3. SNMP

Simple Network Management Protocol

- Das System muss als Network Management System (NMS) agieren
- Das System muss die Formate SNMPv1, SNMPv2 und SNMPv3 unterstützen
- Das System muss Get, GetNext, GetBulk, Set und Response unterstützen
- Das System muss SNMPv1, SNMPv2 und SNMPv3 Traps empfangen können sowie INFORM Requests unterstützen.

- Abfrageintervalle pro Verbindung konfigurierbar
- Konfigurierbares Intervall bis zu 24 Stunden
- Optionaler Offset, um die Last der Netzwerkkommunikation zu verteilen, wenn mehrere Verbindungen gestartet werden und die Bandbreite des Kommunikationsnetzwerks begrenzt ist.

Dieses Protokoll wird für die Fernverwaltung, Diagnose und den Schutz von Netzwerken und Hosts verwendet. SNMP lässt sich für das Management von Geräten verwenden, die einen sogenannten SNMP-Agent ausführen.

Das System muss die Abbildung von SNMP-Agent-Daten als Tabelle unterstützen. Hierzu kann ein Mapping in eine geeignete Variablenrepräsentation erforderlich sein, beispielsweise die Umwandlung in eine Zeichenkette (String).

5.4. IEC62056-21

Das System muss die Möglichkeit bieten, Messgeräte oder Zähler über IEC 62056-21 anzubinden.

Das System muss den Mode C unterstützen.

5.5. OPCUA

Der OPCUA-Treiber dient zur Kommunikation mit OPC UA Servern. OPC UA ist die Abkürzung für OPC Unified Architecture. Hauptmerkmale des Treibers:

- Die Kommunikation erfolgt spontan über Subscriptions, d.h. geänderte Variablen werden automatisch vom Server gemeldet.
- Der Treiber unterstützt mehrere Server
- Die Variablen können direkt aus dem Server gelesen und importiert werden.
- Die Adressierung der Variablen geschieht wahlweise über den Browse Name und Browse Path (TranslateBrowsePathToNodeId) oder persistenten NodeIds
- Der Treiber unterstützt mehrere Subscriptions pro Verbindung
- Der Treiber unterstützt Einstellungen für Subscriptions
- Der Treiber unterstützt den Zugriff auf Arrays auf Basis von einzelnen Elementen oder kompletten Arrays
- Der Treiber unterstützt den Zugriff auf kompletten Strukturen auf Basis von Extension Objects
- Der Treiber unterstützt AbsoluteDeadbands für MonitoredItems
- Der Treiber unterstützt einen konfigurierbaren DataChange Trigger für MonitoredItems
- Der Treiber berücksichtigt OPC UA Server OperationLimits
- Die Verwendung von Zertifikaten muss unterstützt werden in Kombination mit Security Modes Sign und Sign&Encrypt nach den Policies Basic128RSA15, Basic256, Basic256SHA256
- Die Benutzer Authentifizierung beim Verbindungsaufbau muss unterstützt werden.

- Der Treiber unterstützt Event Notifications für Alarms and Conditions und stellt Informationen über Alarme von einem OPC UA Server zur Verfügung.

- Der Treiber unterstützt die Abfrage historischer Daten von OPC UA Servern („history read")

- Der Treiber bietet die Möglichkeit nur eine Teilmenge des OPC-UA-Server-Datenmodells zu berücksichtigen. Es ist möglich einen der mehrere Knoten als Startknoten für das Auslesen des Datenmodells anzugeben.

5.6. BACnet

Das System verfügt über einen BACnet Client Treiber gemäß ANSI/ASHRAE (American Society of Heating, Refrigerating and Air-Conditioning Engineers) und ist damit konform zum Standard 135-2012.

Dieser Treiber ermöglicht die Kommunikation zwischen einem oder mehreren BACnet-fähigen Geräten (BACnet-Devices) und der Runtime über BACnet/IP. Angeschlossenen BACnet-Geräte werden in der Regel als Server betrieben. BACnet Secure Connect muss untertstützt werden.

Das BACnet Protokoll definiert Objekte und Objekt-Eigenschaften. Der Treiber ermöglicht das Lesen und Schreiben einer oder mehrerer beliebiger Eigenschaften eines Objekts. Grundsätzlich wird sowohl pollendes Lesen als auch spontane Kommunikation über COV (Change-of-value) Subscriptions unterstützt.

5.7. Siemens S7 TIA

Der S7-Treiber für S7-1500/1200-Treiber nutzt die erweiterte TIA-Kommunikation über das Transportprotokoll TCP/IP zur S7-1200 und S7-1500. Der Zugriff erfolgt dabei variablenweise über den symbolischen TIA-Zugriffspfad. Optimierte Bausteine sollten unterstützt werden.

5.8. IEC 61850

- Das System sollte als Kommunikationsclient agieren, IEC 61850 Edition 2 unterstützen und muss von einem unabhängigen, von UCAIug akkreditierten Testlabor zertifiziert sein.

- „Direct operate" und „Select before operate", mit und ohne erhöhter Sicherheit, müssen in der Befehlsgabe unterstützt und integriert sein

- RTU-Zeitstempelung und RTU-Qualität müssen im gesamten System verwendet werden

- Online-Browsing von Variablen in der Projektierungsumgebung

- Offline-Browsing (SCL-Dateien)

- Unterstützung von dynamischen Datasets

- Unterstützung von Dateitransfer

    - File Transfer Anforderung für den IEC 61850 Client: Grundlegende File-Transfer Funktionen zum Hochladen von Dateien „GETALL", „GETNEW" und „GETDIFF" müssen vom Treiber direkt angeboten werden. Ein Filtermechanismus ermöglicht dabei die Auswahl von Dateien anhand von Dateiname oder -suffix, sowie die Spezifikation eines Quellordners.

- Originator Category (orCat) definierbar

- Individuelle Trigger-Optionen für Reports (für Geräte, die das vom Client benötigen)
- Das System muss IEC 61850 GOOSE Publisher- und Subscriber-Funktionen anbieten.
- Das System muss Steuerlogik mit IEC 61850 Client- und Serverkommunikation kombinieren können.
- Das System muss für einen IEC 61850 Server die Verwaltung von Einstellwerten über „Setting Groups" unterstützen
- Das System muss Simulation Mode gemäß IEC 61850-7-1 ed2.0 – Part 7.8 unterstützen (GOOSE Simulation, sowohl als GOOSE Publisher als auch als GOOSE Subscriber)
- Das System muss Top-Down-Projektierung nach IEC 61850 unterstützen:
    - automatisierter Import von SSD-Dateien für Einliniendiagramme
    - automatisierter Import von kompletten SCD-Dateien inklusive Reportkonfiguration
- Das System muss die Überwachung von MMS Client/Server und GOOSE Publisher/Subscriber unterstützen.
- Das System muss Internet Protocol Version 6 (IPv6) unterstützen
- Das System muss Service Tracking (LTRK) unterstützen
- IEC 62351 Security - IEC 61850 Client
    - ACSE-Authentifizierung (IEC 62351-4)
    - Unterstützung von TLS Verschlüsselung nach IEC TS 62351-4:2007 und compatibility mode nach IEC 62351-4:2018
    - Unterstützung von MMS Secure Association nach IEC TS 62351-4:2007 und compatibility mode nach IEC 62351-4:2018
- IEC 62351 Security - IEC 61850 Server
    - Unterstützung von TLS Verschlüsselung nach IEC 62351-3 und IEC TS 62351-4:2007 und IEC 62351-4:2018 (compatibility mode)

5.9. IEC 60870

- Das System muss über IEC 60870-5-101, IEC 60870-5-103 und IEC 60870-5-104 kommunizieren können
- Das System muss sowohl als Kommunikations-Master als auch als Slave (siehe weiter unten) agieren können. IEC 60870 Master muss die TLS Verschlüsselung gemäß 62351-3 Standard sowie IEC 60870-5-7 unterstützen
- RTU-Zeitstempelung muss unterstützt werden
- "Direct execute" und "Select and execute"
- COT (Cause of Transmission) Verarbeitung soll möglich sein. Im Besonderen bei der Befehlsgabe

- Online-Browsing für Variablen soll möglich sein

- Das System muss in der Lage sein, den Absender jeder Kontrollanforderung zu ermitteln. Dies ermöglicht es, den Absender für empfangene kontrollbezogene Befehle zu ermitteln. Zur Protokollierung jeder Kontrollanforderung muss das System in der Lage sein, SoE-Einträge (Sequence of Events) zu erzeugen, die die jeweiligen Master identifizieren, welche Kontrollanforderung gesendet haben.

5.10. MQTT (Message Queuing Telemetry Transport)

Das System muss in der Lage sein, als MQTT-Client zu fungieren.

- Unterstützung des MQTT-Protokolls Version 3.1 und 3.1.1

- Unterstützung der sicheren TLS-Kommunikation

- Es muss möglich sein, benutzerdefinierte Payloads zu interpretieren (empfangen) und zu erstellen (senden)

5.11. OCPP (Open Charge Point Protocol)

Das System bietet einen OCPP Treiber und kann so die Rolle des zentralen Managementsystems (CSMS) für die Kommunikation mit Ladestationen für Elektrofahrzeuge erfüllen.

- Das System unterstützt OCCP Version 1.6 auf Basis der WebSocket Kommunikation mit JSON Nachrichtenformat (OCPP1.6-J).

- Das System unterstützt optional auch die gesicherte Kommunikation gemäß OCPP 1.6 Security Whitepaper edition 2.

- Auch die Kommunikation mit Drittsystemen zur Verifizierung der Kartenidentität für einen Ladevorgang kann realisiert werden.

5.12. SPA-Bus Protocol

Das System unterstützt das SPA-Bus Protokoll zur Kommunikation mit ABB RE_54x Schutzgeräten sowie mit SPACOM Schutzgeräten.

6. Upstream-Kommunikation

Das System wird für die Weiterleitung von Daten von untergeordneten Systemen zu übergeordneten System und umgekehrt eingesetzt werden (Gateway-Funktionalität). Darum muss das System dazu fähig sein, Kommunikationsverbindungen zu übergeordneten Systemen über folgende Kommunikationsstandards herzustellen:

- IEC 60870 (-101 und 104)
    - Gateway funktioniert als IEC 60870-5-104 Slave
    - Alternativ als IEC 60870-5-101 Slave in den Modi Balanced und Unbalanced
    - Möglichkeit des Select-Routings zur Weiterleitung des Select-Telegrams an einen anderen Protokoll-Master
    - Möglichkeit zur Massenbearbeitung durch CSV-Import/Export

- Unterstützung von Dateitransfer, sowohl in Meldungsrichtung (Master holt Dateien vom lokalen Slave) als auch in Befehlsrichtung (Master sendet Dateien an den lokalen Slave)

- IEC 60870-101 Slave muss Balanced Mode unterstützen

- Steuerung des Gateway Betriebsmodus für „Gateway Aktiv" oder „Gateway Inaktiv": Mit Mitteln der Applikation kann das Gateway aktiviert bzw. deaktiviert werden. Im inaktiven Zustand besteht keine Verbindung zum IEC 60870 Master. In diesem Zustand werden keine Wertänderungen am IEC 60870 Slave Gateway gepuffert.

- Die Konfiguration eines 60870-101 Slave soll über ein offenes Austauschformat (z.B.CSV) erfolgen können. Sowohl die Konfiguration des Treibers als auch der Export vorliegender Einstellungen müssen unterstützt werden

- IEC 60870 Slave muss die TLS Verschlüsselung gemäß 62351-3 Standard unterstützen

- Der Slave muss Secure Authentication gemäß IEC 60870-5-7 (IEC 62351) unterstützen

- Das Gateway muss die wahlweise die Verwendung von Lokalzeit (Host Zeiteinstellung) oder UTC Zeit (Coordinated Universal Time) unterstützen

- DNP 3
  Das System bietet ein Gateway mit DNP3 Outstation (Slave) Kommunikation. Folgende Funktionen werden hierbei unterstützt:

  - Seriell, TCP, UDP und TCP mit UDP broadcast

  - Self AddressKonformität mit Subset Level 1, 2 und 3 und weiteren Object Group / Variations über Subset Level 3 hinaus

  - Unterstützung von Dateitransfer

  - Unterstützung von Device Attributes

  - Unterstützung Unsolicited Responses

  - Unterstützung von direct operate no ack, direct operate, und Select before Operate

  - Unterstützung von Select / Command Routing zu Downstream Verbindungen

  - Steuerung des Gateway Betriebsmodus für „Fernbedienung" oder „Vor-Ort Bedienung": Ein Wechsel zwischen Fernbedienung und Vor-Ort Bedienung muss mittels Variable möglich sein. Bei Vor-Ort Bedienung werden Kommandos des überlagerten DNP3 Master nicht an die Anlage weitergeleitet

  - Unterstützung von DNP3 Secure Authentication V2 und V5 nach IEEE Std™ 1815-2010 und IEEE Std™ 1815-2012 und IEC 62351-5

  - Unterstützung von TLS nach IEEE Std™ 1815-2012 und IEC 62351-3

  - Unterstützung von einem Single Master

  - Unterstützung von zwei IP-Adressen für einen Master bei der Verwendung von TCP/IP. Das SCADA System kann zur Laufzeit überprüfen, welche IP-Adresse aktuell verwendet wird

- Per Sonderoption soll der Außenstation eine beliebige Master-IP-Adresse für die TCP-Kommunikation zugewiesen werden

- Das System unterstützt einen redundanten Betrieb der DNP3 Outstation Kommunikation, gemäß des Einsatzes eines redundanten Server-Paares (Prozessführender Server und Standby Server). Im Falle eines Serverausfalls und Übernahme der Prozessführung durch einen Standby Server wird die Kommunikation zum überlagerten DNP3 Master ohne Verlust von Prozesswertänderungen fortgesetzt. Das System verhindert dabei das mehrfache Senden einzelner Events (Vermeidung von Duplikaten).

- Das SCADA System kann zur Laufzeit überprüfen zu welchem Rechner der überlagerte DNP3 Master aktuell verbunden ist, ob eine Unterbrechung der Verbindung vorliegt, oder ein Kommunikations-Timeout vorliegt

- Invertierung von Binary Inputs sowie lineare Skalierung von Analog Inputs kann optional konfiguriert werden

- Das Protokoll-Gateway der Außenstation muss eine konfigurierbare Hysterese für Analogwerte unterstützen. Dies soll dazu beitragen, die Anzahl der Ereignisse durch permanente Analogwertänderungen zu reduzieren.

- Die Kommunikation der DNP3 Outstation kann statistisch auf physischer Ebene, Data-Link Ebene und Transport Ebene überwacht werden (vgl. OSI Model Ebenen 1,2 und 4). Bei zunehmenden Störungen können über Schwellwerte spezifische Alarmmeldungen generiert werden

- Der Status des Applikationsprozesses für das Gateway kann überwacht werden. Es können dabei Zustände wie „Hochlauf", „In Betrieb", „Neustart" oder „Beenden" im Rahmen der HMI/SCADA Applikation durch Variablenwerte erfasst werden

- OPC UA
  Das System bietet ein Gateway für OPC UA Server Kommunikation. Folgende Funktionen werden hierbei unterstützt:

  - Das Gateway bietet OPC UA Data Access zu konfigurierbaren Variablen von Downstream Verbindungen oder internen Variablen in Leserichtung oder auch Schreibrichtung. Flexibles Mapping von Prozessvariablen auf Knoten vom Typ Variable

  - o Kundenspezifisch erstellte OPC UA Datenmodelle können mit dem Gateway genutzt werden

  - Das Gateway unterstützt OPC UA Historical Access zu Variablen aus dem Historian

  - Das Gateway unterstützt OPC UA Alarms als Events für Alarme

  - Das Gateway unterstützt OPC UA Historical Access für Alarme

  - Das Gateway unterstützt OPC UA Conditions als Events

  - Das Gateway unterstützt OPC UA Historical Acces für Events

- Das Gateway unterstützt OPC UA Authentifizierung von Benutzer
- Das Gateway unterstützt die OPC UA Modes Sign und Sign&Encrypt
- Das Gateway muss nach der OPC-Spezifikation zertifiziert sein. Version 1.04 der OPC Foundation

- MODBUS
  Das System bietet ein Gateway für Modbus (Slave) Kommunikation. Folgende Funktionen werden hierbei unterstützt:
  - Das Gateway unterstützt die Modbus Function Codes 1, 3, 5, 6, 15 und 16
  - Das Gateway unterstützt serielle Kommunikation als Modbus RTU
  - Das Gateway unterstützt TCP/IP basierte Kommunikation als Modbus TCP
  - Das Gateway bietet Modbus Zugriff zu konfigurierten Variablen von Downstream Verbindungen oder internen Variablen in Leserichtung und Schreibrichtung

- SNMP Gateway works as SNMP agent and thus monitors the SCADA application. Support of SNMPv3 including encryption and authentication. Support of traps for events and alarms and support of MIBs. Das integrierte Prozess-Gateway muss SNMPv3 unterstützen, um eine verschlüsselte Kommunikation nutzen zu können.

- New (SNMP driver):

- The system must support individual polling intervals for its connections

- Syslog
  Das System kann mittels Syslog Protokoll Meldungen an einen Syslog Server übermitteln. Es wird die Kommunikation via Syslog-Format gemäß RFC 3164 via UDP-Transport unterstützt. Das System erlaubt die optionale Einstellung der automatischen Übertragung von SCADA Alarmen und/oder Ereignissen.

- ICCP/TASE.2/IEC 60870-6
  Gateway funktioniert als Server und Client und unterstützt Conformance Blocks 1, 2 und 5. Der ICCP Client unterstützt ‚Domain Specific Adressing'.
  - Unterstützung von Select / Command Routing zu Downstream Verbindungen
  - Als ICCP Client: Unterstützung von „Select and Operate" unter Berücksichtigung des Status der darunterliegenden Anlage
  - Unterstützung von TLS Verschlüsselung und MMS Security nach IEC 62351-4:2007, sowie Compatibility Mode nach IEC 623451-4:2018.
  - Unterstützung einer benutzerdefinierten Konfiguration von Data Sets bei Verwendung als ICCP Client

Die Gateway-Funktionalität soll als modulare Softwarefunktion verfügbar sein. Folgende Eigenschaften und Anwendungsszenarien werden unterstützt:

- Das Gateway kann auf einem Rechner mit laufender HMI/SCADA Runtime installiert und betrieben werden. Dabei ist es gleichgültig, ob die Runtime als prozessführender Server, Client oder als standalone HMI betrieben wird

- Das Gateway kann jederzeit nachträglich, zur Erweiterung einer Applikation installiert werden. Das Engineering Tool erlaubt die Onlinekonfiguration von Datenpunkten. Das Originalprojekt wird für diese Konfigurationsvariante nicht benötigt. Der Applikationsentwickler erhält die Liste der verfügbaren Variablen direkt durch eine Verbindung des Gateway Konfigurationstools mit der HMI/SCADA Runtime.

- Die Konfiguration des Gateways kann durch eine integrierte Modulkonfiguration des HMI/SCADA Projekts erfolgen. Somit sind jegliche Variablen bereits zur Entwurfszeit verfügbar. Änderungen in den Definitionen (Variablen, Typen etc.) des HMI/SCADA Projekts werden unmittelbar in der Gateway Konfiguration berücksichtigt.

- - Die Konfiguration des Gateways kann mit benutzerdefinierten Skripten, die auf einer speziellen API basieren, vollständig automatisiert werden.

- Das System unterstützt den Betrieb mehrerer Gateway Instanzen in Verbindung mit einer HMI/SCADA Runtime zu einem Zeitpunkt. Somit können einerseits Verbindungen zu verschiedenen Einrichtungen mittels unterschiedlicher Gateway Protokolle realisiert werden. Andererseits können auch mehrere unterschiedliche Verbindungen auf Basis desselben Protokolls realisiert werden, etwa zum Aufbau redundanter Architekturen.

- Das System muss Überwachungsmöglichkeiten bieten, um den Prozessstatus der betriebenen Gateway-Instanzen ständig zu beobachten. Es soll die Identifizierung der Zustände "Startup", "Running", "Stop" oder "Error" der eingesetzten Gateway-Instanzen ermöglichen. Darüber hinaus muss der Gateway-Status dem HMI/SCADA-Laufzeitsystem für die Live-Anzeige, die Erzeugung von Alarmen oder die Protokollierung von Prozessdaten (Archivierung) und die entsprechende Reporting zur Verfügung stehen.

-

7. Verbindung mit der Microsoft® Azure Cloud

Das System muss die Integration mit gängigen Cloud Systemen unterstützten. Neben der kommunikativen Verbindung mittels der beschriebenen Kommunikationsprotokolle und Standard-Schnittstellen sind Protokolloptionen erforderlich, die einen integrierten Betrieb mit der Microsoft Azur Cloud (im Weiteren kurz „MS Azure“) ermöglichen.

- Das System verfügt über ein Gateway zur Übertragung von ausgewählten Variablenwerten an die Kommunikationsdienste der MS Azure Cloud, Service Bus, Event Hub oder IoT Hub.

- Die Übertragung (Kodierung) erfolgt jeweils in einem geeigneten Nachrichtenformat das auch die zum Variablen-Istwert gehörenden Zusatzwerte, Zeitstempel und Statusinformation, umfasst. Dies erlaubt die Weiterverwendung der übermittelten Daten.

- Das System verfügt über Kommunikationsschnittstellen, um zuvor an MS Azure Kommunikationsdienste übertragene Variablenwerte wieder abzurufen

Somit kann das System die MS Azure Kommunikationsdienste nutzen, um Prozessdaten für cloudbasierte Services zur Verfügung zu stellen. Im Weiteren können MS Azure Kommunikationsdienste genutzt werden, um verteilte Systeminstallationen zu verbinden (Laufzeit Prozessdatenkommunikation).

8. Anpassung und Erweiterbarkeit

Das System muss Möglichkeiten zur Erweiterung der bestehenden Menge an Kommunikationsstandards bieten. Wenn eine Kommunikationsschnittstelle fehlt, so muss das System

ein Konzept für eine nachträgliche modulare Implementierung bieten. Alle Mechanismen bezüglich Datenfluss und Variablenverwaltung müssen identisch auch für die neu implementierten Kommunikationsschnittstellen unterstützt werden. Eine wohldefinierte Softwareschnittstelle, wie z.B. das Microsoft Component Object Model (COM), muss für eine modulare Integration von zusätzlichen Kommunikationstreibern verfügbar sein.

Für kleinere Datenaustausch-Szenarios oder rudimentäre Datenbankverbindungen muss eine Programmierschnittstelle für höhere Programmiersprachen, wie z.B. VBA/ C# / .NET gemeinsam mit spezifischen Funktionsbibliotheken zur Verfügung stehen.

Das System bietet eine Möglichkeit der Erweiterung, um die Kommunikation mit den unterschiedlichsten REST-basierenden Diensten zu ermöglichen. Darüber hinaus ist die Anbindung von Applikationen, die moderne Netzwerkprotokolle wie MQTT nutzen, möglich. Die Implementierung der Erweiterungen kann mittels einer höheren Programmiersprache wie C# umgesetzte werden.

9. Anwendung im Netzwerk

Das System muss im Netzwerk verwendbar sein. Eine beliebige Anzahl von Clients muss sich mit dem jeweiligen Anwendungsserver verbinden können. Für verbundene Clients müssen eine volle Bedienbarkeit und Funktionalität für alle Stationen gegeben sein. Es muss jedoch eine umfassende Zugriffskontrolle über die vorhandene Benutzerverwaltung angewendet werden (siehe Abschnitt 1.9).

Die Ladezeit für die Anzeige von Bildern darf nicht von der Anzahl der Clients abhängig sein. Es muss möglich sein, zusätzliche Clients, ohne jeglichen Entwicklungsaufwand anzubinden.

Für einen Client / Server-Verbund muss das System sicherstellen können, dass alle verbundenen Stationen auf demselben Projektdefinitionsstand sind, um für Konsistenz zu sorgen (automatische Synchronisation).

Im Falle von Projektänderungen müssen alle Clients diese Änderungen übernehmen. Das System muss also die Möglichkeit bieten, selektive und konsistente Software-Updates durchzuführen (Runtime-System und Anwendungsprojekt). Dies muss auch während der Runtime ohne Anhalten der Clients möglich sein.

Das Anwendungsprojekt wird nämlich mehrere entfernte Steuerpunkte mit den zugehörigen Kommunikationsverbindungen abdecken (Upstream- und Downstream-Verbindungen). Teilprojekte werden auf mehreren Servern bereitgestellt. Darum muss das System die Kopplung von mehreren Servern und Clients im Kontext eines globalen und interaktiven Anwendungsnetzwerks unterstützen.

Das System muss die Möglichkeit bieten, globale Projekte sowie Teilprojekte auf unterschiedlichen Stationen bereitzustellen.

Um den benötigten Netzwerkverkehr zu minimieren, müssen die Daten von dem Server auf alle Clients spontan übertragen werden.

Um Bedienungsfehler zu vermeiden, muss es eine Möglichkeit geben, in einen Prozess nur von einem Computer aus einzugreifen. Alle anderen Stationen im Netzwerk müssen als Beobachter online bleiben. Falls notwendig, müssen Sie in der Lage sein, operativen Zugriff zu beantragen.

Außerdem muss das System die folgenden netzwerkbezogenen Sicherheitsfunktionen unterstützen:

- Schutz der Netzwerkkommunikation durch Verschlüsselung (mindestens 192 Bit) sämtlicher netzwerkbasierter Kommunikation zwischen allen beteiligten Komponenten (Server, Clients, Datenbankarchive, Web-Clients).

- HTTP-Tunneling für Webserver. WebClient Plug-Ins für zum Beispiel den Internet Explorer.

- Unterstützung von IPv6-Netzwerken durch alle Systemkomponenten.

- Möglichkeit der Fehlerdiagnose für Netzwerk und Steuerkommunikation

10. Prozesssteuerung

Das System muss die Möglichkeit bieten, Steuerungsprogramme zu entwerfen und zu betreiben. Diese Option wird für die Erweiterung und Überarbeitung des Runtime-Systems benötigt, um es hinsichtlich spezifischer Automatisierungsaufgaben anzupassen. Darum wird eine Programmierumgebung für die effiziente Entwicklung von Steuerungsprogrammen benötigt.

Der jeweilige Entwicklungskontext (Programmierumgebung) muss in das System integriert sein, um einen direkten Zugriff auf die Variablen des Grundsystems zu ermöglichen. Eine redundante Datenablage muss strengstens vermieden werden. Das Prozessleitsystem muss eine Möglichkeit bieten, den Code online während des Systembetriebs zu debuggen, um ein effizientes Testen und Fehlersuchen zu erleichtern.

Etablierte, wohldefinierte Programmierstandards, wie z.B. VBA, C# oder .NET, müssen einsetzbar sein. Es muss für den Entwickler möglich sein, frei zwischen den Programmiervarianten zu wählen und diese zu kombinieren, je nach vorliegender Aufgabenstellung, um verschiedenste Funktionen auch gemeinsam nutzen zu können. Es muss möglich sein, Task-Prioritäten zu konfigurieren.

In Zusammenhang mit SIMATIC S7 GRAPH-Systemen muss es möglich sein, die Schrittketteninformationen direkt in den Prozessbildern abzubilden. Eine effiziente Fehlersuche muss durch grafische Kennzeichnung unterstützt werden.

10.1. Programmierschnittstellen

Das System muss Möglichkeiten zur Anordnung und Ausführung von Steuerungsskripten unter Einsatz folgender Programmierstandards bieten:

- VB (Visual Basic) und VBA (Visual Basic for Applications)

- Die auf dem .NET Framework basierenden Sprachen C# und Visual Basic.NET

- Unterstützung von COM (Component Object Model)

Steuerungsskripte müssen entweder durch einzelne Events (Funktionsaufruf), zyklische Events, einen vordefinierten Zeitplan (z.B. Kalenderfunktion) oder Benutzereingabe ausgelöst werden.

Das System bietet dazu eine Add-In Schnittstelle an, welche die Möglichkeit zur Verfügung stellt, die Funktionalitäten des Systems zu erweitern und Projektierungshilfen zur Verfügung zu stellen. Das System sollte unterschiedliche Typen von Add-Ins zur Verfügung stellen:

- Wizard: Wird vom Benutzer gestartet, wird durchlaufen und nach Lösung der Aufgabe wieder beendet. Andere Fenster sind während der Ausführung eventuell nicht zugänglich. Ein Wizard kann mit einer Benutzer-Oberfläche oder im Hintergrund ablaufen.

- Service: Wird mit dem Engineering oder der Runtime gestartet und läuft im Hintergrund.

Der Einsatz aller Standardmethoden und Programmiermechanismen aller Sprachen muss möglich sein. Außerdem müssen die jeweiligen Standardbibliotheken verwendbar sein, die üblicherweise mit einem Programmierstandard verknüpft sind (Framework-Funktionen, z.B. .ADO.NET für den Datenbankzugriff).

Für die Steuerung spezifischer Runtime-Funktionen (SCADA) müssen die jeweiligen APIs zur Verfügung gestellt werden. Diese müssen eine konsistente und synchronisierte Manipulation der jeweiligen Steuerungsmechanismen erlauben. Der Zweck und die Auswirkungen einer jeden Funktion müssen durch ihre Signatur, Attribute und Parameter klar ersichtlich sein.

10.2. Soft-SPS

Das System muss eine mit IEC61131-3 konforme Programmierschnittstelle bieten. Die betreffenden Aufgaben (zyklische Aufgaben) müssen in einem Soft-SPS-Kontext mit anpassbarem Timing ausführbar sein. Das System muss die notwendigen Hilfsmittel für die Programmerstellung und Versionierung sowie Diagnosetools und funktionale Bibliotheken (mathematische Funktionen, kombinatorische Funktionen, Signalfilter, Befehlsgabe etc.) bieten.

Die Soft-SPS muss dazu imstande sein, transparent auf die Variablen (Istwert und Statusinformation in Echtzeit, Lesen und Schreiben) des SCADA-Systems sowie auf die von externen Datenquellen (über Kommunikationsverbindungen) empfangenen Daten zuzugreifen. Es muss also eine Verbindung zu echten Hardware-I/Os (Input-/Output Signale) machbar sein. Es muss die Option geben, die SPS-Komponente gemeinsam (integriert) mit dem SCADA-System oder individuell als separate Komponente auf einer Fernstation zu betreiben. Die SPS-Komponente kann im lokalen als auch im entfernten Betrieb als Gegenstelle für Schalthandlungen verwendet werden. Hierzu muss das System die Abwicklung einer ‚Select and Execute' (sicheres Selektieren und Schalten) Sequenz unterstützen.

Das Kommunikationsprotokoll, das den Datenaustausch zwischen Soft-SPS und SCADA-Laufzeitsystem abwickelt, muss eine Pufferung für den Fall einer Unterbrechung des Kommunikationskanals unterstützen. Ereignisse (Wertänderungen) müssen bis zu mindestens 48 Stunden gepuffert werden. Nach Wiederherstellung der Verbindung werden die entsprechenden Ereignisse und Alarme an die SCADA-Laufzeit übertragen.

Die integrierte Soft-SPS muss direkte Verbindungen zur Schnittstellen haben, die Modbus, IEC 61850 Client, Server und GOOSE unterstützen.

Das Engineeringsystem muss IEC 61131-10 (PLCopen) zum Importieren und Exportieren von Projektelementen unterstützen.

Das Engineering System muss einen integrierten Vergleich zwischen SPS Programmen unterstützen. Anhand diese Vergleichs müssen sich Änderungen zwischen unterschiedlichen Projektständen, sowohl für textuelle als auch für grafische Programme, darstellen lassen.

Die Soft-SPS ist unter Linux lauffähig und kann über Docker-Container bereitgestellt werden.

The gesamte IEC 61131-3 Umgebung muss UTF8 (Unicode) unterstützen.

Die IEC 61131-3 Umgebung soll eine direkte Integration von Python-Skripten unterstützen. Es muss eine Schnittstelle für den bidirektionalen Austausch von Variablenwerten vorhanden sein. Die IEC 61131-3-Laufzeitumgebung muss es ermöglichen, vordefinierte Python-Skripte auszulösen/zu starten, die auf einer dedizierten Python-Engine verarbeitet werden.

10.3. Zeitsteuerung über Zeitpläne

Es muss eine integrierte Methode zur Verwaltung von Prozesssteuerungsmaßnahmen über einen Kalender oder einen Scheduler geben.

Der Bediener muss dazu imstande sein, Schaltzeiten zu definieren – vorzugsweise über das Öffnen eines Zeitfensters im Kalender mit gedrückter linker Maustaste. Dieser Kalender muss auch Funktionen wie das Versenden von E-Mails oder das Drucken von Reports auslösen können. Die Zeitsteuerung muss einerseits öffentliche Feiertage anzeigen können, andererseits auch die Erstellung spezieller Tage wie Betriebsurlaub, Wartungsfenster etc. ermöglichen.

Für eine klar strukturierte Darstellung der Anlage muss es möglich sein, mehrere Einrichtungen und Zeitpläne zu definieren. Außerdem muss der Scheduler alle Kompatibilitätsmerkmale des Basissystems hinsichtlich Netzwerk, Redundanz und Betriebssystem unterstützen.

Die gesamte IEC 61131-3 Umgebung muss den UTF8 (Unicode) Standard unterstützen.

10.4. Zeitsteuerung über Schichtmodelle mithilfe von Zeitplänen

Es muss die Möglichkeit geben, abhängig von Tag, Zeit, Schicht oder anderen Events Steuerungsinterventionen vorzunehmen. Ein logischer Gruppierungsmechanismus muss bereitstehen, um logisch verbundene Einheiten zu kombinieren. Danach können bestimmte Aktionen selektiv auf diese Gruppen angewendet werden. Dadurch muss eine Organisation der Installationen, Gebäude, Büros und anderer hierarchischer Strukturen ermöglicht werden.

Steuerungsbezogene Zeitpläne müssen Variablen und Funktionen umfassen, die abhängig von absoluten oder relativen Zeitpunkten festgelegt bzw. ausgeführt werden. Absolute Zeitpunkte sind klar definierte Punkte in der Zeit. Relative Zeitpunkte basierend auf Timing-Modellen, wie z.B. Start und Ende von Evaluierungsperioden, Schichten oder Wartungsfenstern sowie frei definierbaren Events. Es muss möglich sein, Zeitpläne zu aktivieren und zu deaktivieren.

Außerdem müssen Zeitmodelle auch Schichten und Pausen einbeziehen. Sie stellen die Grundlage für die Arbeitszeiten in einer Anlage dar. Sie müssen also als Berechnungsbasis für die relative Zeit in den Zeitplänen dienen. Autorisierte Benutzer müssen Events frei definieren können.

Die Kalenderschnittstelle muss die Einstellung alternativer Zeitpläne an ausgewählten Tagen ermöglichen. Es muss also möglich sein, je nach Wochentag oder Saison verschiedene Schichtmodelle einzustellen.

Länderspezifische Ferien werden aus Kalendertools wie der Microsoft Outlook Kalenderfunktion importiert werden. Der Kalender muss eine Monats-, Wochen- und Tagesansicht bieten.

Die Anzeige- und Bearbeitungsoptionen müssen auch in der Benutzerschnittstelle (Visualisierungs-Client) verfügbar sein und müssen online anpassbar sein. Änderungen müssen in der Chronologischen Ereignisliste protokolliert werden.

Um eine klare Übersicht während der Projektierung und Wartung zu bewahren, muss das System eine klar strukturierte Schaltpunkt-Vorschau anbieten. Die Vorschau der Ausführung muss von den Filtereinstellungen (Einrichtung, Zeitplan, Variable, Funktion, Zeit) sowie den jeweiligen Daten (aktuelle Projektierungsdaten, aktuelle Ausführungsdaten) abhängig sein.

10.5. Prozessanalyse

Zur Auswertung von Prozessverhalten und -leistung werden Funktionen für eine historische Analyse benötigt. Das System muss in der Lage sein, Prozessdatenaufzeichnungen, sowie Meldungen und Alarmlisten mittels konfigurierbarer Darstellungen zu visualisieren. Neben der Darstellung

bestimmter Zeitspannen muss die Selektion der Daten anhand von Schichtinformation aus einem integrierten Schichtverwaltungssystem unterstützt werden. Somit können Berichte über Alarme, Meldungen und Prozessdaten selektiv für einen Teilbereich der Anlage oder die gesamte Anlage gleichermaßen, und unter Berücksichtigung der Dauer und Art von produktiven Schichten erstellt werden. Die Einbeziehung aktueller Prozessdaten (z.B. aktuell anliegende Prozesswerte) muss möglich sein. Die Darstellung der Ergebnisse muss über dynamisch generierte Meldungslisten, Trenddarstellungen oder konfigurierbare Reports erfolgen.

10.6. Chargenorientierte Produktion

Das System verfügt über eine Funktion zur Steuerung chargenorientierter Fertigungsprozesse nach ISA 88 (sog. Batchbetrieb).

Es wird die Erstellung und Verwaltung von Aggregatsklassen unterstützt. Ablaufrezepte können unabhängig – sprich, für unterschiedliche Anlagen verwendbar - erstellt werden. Erst bei Ausführung eines solchen Ablaufrezepts muss die konkrete Anlage definiert werden.

Die ISA 88 konforme Batch Engine ist vollständig in das System integriert und verfügt über die Möglichkeit einer redundanten Ausführung.

11. Datenaufzeichnung und -archivierung

11.1. Archivverwaltung

Die Automatisierung benötigt Möglichkeiten zur Erhaltung bestimmter Prozesswerte für die spätere Auswertung und Analyse. Darum muss das System Mechanismen anbieten, welche die Erstellung von Datenarchiven und die Verwaltung des zugehörigen Datenvolumens ermöglichen.

Alle IEC-kompatiblen Datentypen, wie Binärwerte, numerische Werte signierte Werte und String Werte, müssen aufzeichenbar sein. Dementsprechend müssen Archive durch Auswählen der benötigten Variablen erstellt werden. Alle zusätzlich existierenden Statusinformationen zu einer Variable müssen archiviert werden, gemeinsam mit den Prozesswerten und einem präzisen Zeitstempel (Auflösung in Millisekunden). Die Übertragung der Werte in ein Archiv wird zyklisch, eventbasiert oder spontan, also wenn ein Wert sich ändert, angestoßen werden.

Außerdem muss es die Möglichkeit geben, Werte zu existierenden Archiven hinzuzufügen und die Datenvolumen konsistent zu reorganisieren (z.B. Sortieren nach zeitlicher Anordnung).

Unabhängig davon, ob es sich um ein Einzelserver- oder Mehrfachserver-System handelt, müssen die Archive zentral verwaltet werden.

Außerdem muss eine redundante Datenablage strengstens vermieden werden. Um eine ausreichende Systemperformance zu erzielen, müssen Archivdaten generell in optimierten Binärformaten abgelegt werden. Als Option müssen Daten in den Formaten XML oder CSV (Character Separated Values) gespeichert werden können.

Außerdem soll es eine Möglichkeit geben, Archive zu komprimieren, um diese Ressourcenschonend ablegen zu können. Hierfür wird ein Algorithmus wie Swinging Door Trending Compression verwendet.

Für die Langzeitarchivierung muss die Auslagerung von Daten in eine SQL-Datenbank (OLEDB / ODBC Verbindung) als Alternative zur internen Datenablage möglich sein. Diese Daten müssen zu einem späteren Zeitpunkt transparent wiederauffindbar und darstellbar sein.

Das System muss eine Archivierungsperformance bieten, die auch bei hohem Aufkommen von Werteaufzeichnungen eine begrenzte Systemlast (i.S. Rechnerleistung) sicherstellt. Im Fall hoher Archivierungslasten muss eine redundante Architektur voll funktional bleiben. Das erforderliche Datenaufkommen liegt, je nach Anforderung an die Anwendung - z.B. konstante zyklische Aufzeichnung oder spontane, ereignisbasierte Aufzeichnung – bei mehreren zehntausend Aufzeichnungen pro Sekunde.

11.2. Chargenarchive

Für spezielle Anwendungen ist es notwendig, eine chargenorientierte Archivierung zu unterstützen. Jede Charge muss als ein individueller Block archiviert werden. Der zugehörige Chargenname muss ebenfalls gespeichert werden. Die Chargennamen müssen in einer Indexdatei abgespeichert werden, um eine schnelle Suche zu ermöglichen.

11.3. Verdichtung von historischen Daten

Um Speicherplatz zu sparen, muss es möglich sein, Werte auf verschiedenen Verdichtungsstufen abzuspeichern. Idealerweise müssen Werte, die mit einer hohen zeitlichen Auflösung gespeichert worden sind (z.B. Sekunden) in Archiven mit einer niedrigeren zeitlichen Auflösung gespeichert werden können (z.B. Stunden). Für die Erstellung von Verdichtungsarchiven müssen automatische Berechnungen, wie z.B. Minimum und Maximum, Mittelwert und Summen, unterstützt werden. Ein mehrschichtiges Kaskadieren von Archiven muss die Kompression von Daten mit jeder gewünschten Rate möglich machen.

Autorisierte Benutzer müssen Archive manuell bearbeiten und in Standard-Dateienformate exportieren können (dBase, ASCII, XML). Es muss Funktionen für die Aktivierung und Unterbrechung einer Archivaufzeichnung geben. Dies wird entweder ereignisgesteuert oder automatisch genutzt werden.

11.4. Redundanz und Datenkonsistenz

Die Systemfunktionen zur Archivierung und Aufzeichnung müssen voll integriert mit den allgemeinen Redundanzfunktionen sein, wie in Abschnitt 1.5 behandelt. Ein Datenverlust während eines Serverausfalls muss also verhindert werden. Außerdem müssen Werte, die vorübergehend ungültig sind, dementsprechend signiert werden (z.B. „Wert getrennt“, Variablenstatusinformationen).

12. Fernzugriff auf das System

Automatisierungssoftware muss flexiblen Zugriff für dedizierte Personen über verschiedene Kanäle bieten. Die verschiedenen Methoden des Fernzugriffs müssen einen betrieblichen Zugriff von entfernten Standorten erleichtern, z.B. über Intranet oder Internet sowie Mobilnetzwerke (UMTS, 3G etc.).

In jedem Fall muss der Fernzugriff durch die bereits definierten Mechanismen der Benutzerverwaltung (siehe Abschnitt 1.9) gesteuert werden, um einen unautorisierten Zugriff zu vermeiden. Die Zugriffsmethoden müssen frei konfigurierbar sein und dürfen nicht das interne Anwendungsdesign des Systems beeinflussen. Es darf kein Anpassungsaufwand für den Aufbau einer Fernzugriffsverbindung entstehen, falls das System geändert oder erweitert wird. Die Visualisierungsinhalte müssen automatisch angepasst werden (z.B. Bildschirmauflösung), da verschiedene grafische Anzeigegeräte zum Einsatz kommen werden.

12.1. Terminal Server

Das System muss die Veröffentlichung mehrerer Visualisierungsclients im Kontext einer Terminal-Server/Client-Architektur unterstützen. Darum muss es möglich sein, die Benutzerschnittstellen des Systems über einen zentralen Terminal Server an Remote Terminal Clients (z.B. Thin Clients) zu veröffentlichen.

Dies wird z.B. dafür genutzt werden, einen externen Zugriff auf das System über eine VPN-Verbindung (Virtual Private Network) zu dem Terminal Server zu bieten, der sich innerhalb des dedizierten IT-Netzwerkes der Prozessinfrastruktur befindet. Ein gleichzeitiger Zugriff durch mehrere Benutzer muss unterstützt werden.

12.2. Webserver

Die Software muss die Möglichkeit bieten, grafische Benutzerschnittstellen (HMIs) unter Einsatz von webbasierter Kommunikation und Visualisierungsstandards (HTTP, HTML etc.) zu veröffentlichen. Darum muss es möglich sein, auf Prozesse über etablierte Webbrowser wie Microsoft Internet Explorer zuzugreifen, diese anzuzeigen und zu steuern. Die webbasierte Ansicht muss identisch mit dem originalen Bilddesign sein und dieselben Steuermöglichkeiten bieten.

12.3. Mobile Statusanzeige

Für Zwecke der Fernüberwachung muss es möglich sein, die wichtigsten Prozessinformationen (Variablen, Statistiken, Betriebszustände) auf Standardmobilgeräten, wie z.B. Smartphones oder Tablets, anzuzeigen. Die Auswahl der anzuzeigenden Variablen muss frei konfigurierbar sein. Die Informationen müssen auf Seiten mit Listen oder grafischen Elementen, wie z.B. Messuhren oder Fortschrittsbalken, angezeigt werden. Wenn der Mobilzugriff mithilfe von dedizierten mobilen Apps umgesetzt ist, so müssen Windows Phone, Google Android und Apple iOS unterstützt werden.

13. Sicherheit

Automatisierungsaufgaben benötigen ein hohes Maß an Prozesssicherheit, Verfügbarkeit und Datenschutz. Es werden also die folgenden Sicherheitsaspekte berücksichtigt werden, um für ein hohes Sicherheitsbewusstsein in Entwicklung, Betrieb und Wartung des betreffenden Überwachungssystems zu sorgen. Bereits die Inbetriebnahme (bzw. die Echtheit der Installation) des Systems muss bereits vorab durch z.B. Hash Codes vom Hersteller für die Setups sichergestellt sein.

13.1. Projektierungssystem

- Erkennung von manipulierten Programmdateien (Dateisignatur)
- Automatische Synchronisation von Dateien (z.B. Runtime-Definitionen) in der Anwendungstopologie (Netzwerk), Schutz gegen inkonsistente Entwicklungszustände
- Selektiver Passwortschutz im Projektierungstool
- Projekt Versionierung und -vergleich, Erstellung und Wartung einer Änderungshistorie
- Fehlervermeidung durch das Verwenden von Wizards (Automatisiertes Engineering)
- Konsistente Objektorientierung und Wiederverwendung von entwickelten Komponenten (also Einsatz bewährter Komponenten)
- Volle Integration aller Lösungskomponenten (also konsistente Anordnung von Komponenten und Schnittstellen, automatische Nutzung von Datenverarbeitungsmechanismen

13.2. Laufzeitsystem

- Betriebsjournal

- Umfassende Unterstützung von (selektiven) Authentifizierungsmechanismen (Verhinderung von nicht authentifiziertem Zugriff) inklusive Windows Domain Anbindung der Benutzer Verwaltung. Optional kann dem Benutzer des Laufzeitsystems bei dem Versuch einer Bedienung mit unzureichenden Benutzerrechten detaillierte Information über erforderliche Benutzerebenen angezeigt werden.

- Statusverarbeitung und selektive Sicherheitsmechanismen (Prozess Verriegelungen). Das Vorliegen eine aktiven Prozess Verriegelung kann dem Benutzer entweder grafisch (Sichtbarkeit, Einfärbung von Elementen), oder durch Einblendung einer textuellen Beschreibung der Verriegelungsursache kenntlich gemacht werden.

- Versionierung und Statushandhabung für Rezepte, Protokollierung von autorisierten Änderungen.

- Unterstützung einer Systemevaluierung/-zertifizierung sowie von Re-Evaluierungsszenarios durch ein transparentes Management von Entwicklungs- und Konfigurationsaktivitäten (Audit Trail), idealerweise gemäß wohldefinierter Branchenstandards, wie z.B. FDA 21 CFR Part 11

- Konsistente Fortführung des Systembetriebs nach Serverausfällen, unterbrechungsfreie Redundanz

- Benutzerverwaltung: volle Unterstützung von Active Directory, Vermeiden redundanter Benutzerdefinitionen

- Hohe Kompatibilität zwischen den einzelnen Systemversionen

- Änderungshistorie und Sicherung von Projektversionen, Rückverfolgbarkeit

- Betrieb der Runtime als Dienst (ohne Desktop-Interaktion) /Unterstützung von Terminal-Server- und Thin-Client-Lösungen

- gesicherter Betrieb, um den Zugriff auf den Windows Desktop zu verhindern

13.3. Netzwerk

- Schutz der Netzwerkkommunikation durch Verschlüsselung (mindestens 192 Bit) sämtlicher netzwerkbasierter Kommunikation zwischen allen beteiligten Komponenten (Server, Clients, Datenbankarchive, Web-Clients).

- Das System muss OpenSSL 3 für TLS (Transport Layer Security) verwenden.

- HTTP-Tunneling für Web-Server. Webclient-Plug-Ins für z.B. Internet Explorer

- Unterstützung von IPv6-Netzwerken durch alle Systemkomponenten.

- OPC-UA: Client und Server unterstützen Zertifikate und Benutzerauthentifizierung

- Möglichkeit der Fehlerdiagnose für Netzwerk und Steuerkommunikation

- Verteilte Systeme und Inselbildung müssen möglich sein (Security in Depth-Strategie)

14. PRP

PRP (Parallel Redundancy Protocol, IEC 62439-3) muss durch die Software ohne zusätzliche Kosten unterstützt werden. Es muss auf allen Windows-Betriebssystemen unterstützt werden, auf denen das SCADA-System lauffähig ist, auch in Kombination mit einem Hypervisor (virtuelle Umgebung). Das System muss hierbei Gigabit Ethernet unterstützen.

15. Systeminbetriebnahme und -wartung

15.1. Variablen-Diagnose

Es muss möglich sein, eine Liste aller bzw. eine gefilterte Menge von Variablen anzuzeigen. Dieses Bild wird hauptsächlich für die Inbetriebnahme der Variablen eingesetzt werden, aber auch für Fehlersuche, Diagnose und Wartungsaufgaben.

15.2. Debugging Tool

Das System muss ein Tool anbieten für die Auswertung von Problemen, die nicht mit den Mitteln der HMI-Anwendung gelöst werden können. Das Tool muss weitere Informationen über das Runtime-System aber auch die Kommunikationsprotokolle zur Verfügung stellen.

16. Spezialfunktionen für die Energiebranche

Energieanlagen und Schaltanlagenautomatisierung benötigen spezifische HMI/SCADA-Funktionen für einen optimalen Betrieb. Allgemein gesagt, müssen diese Funktionen einen sicheren und zuverlässigen Betrieb der jeweiligen Prozesse gewährleisten. Dies muss einerseits mithilfe einer Anwendung von Sicherheitskontrollmechanismen erreicht werden. Andererseits muss die Systeminteraktion mit erweiterten Funktionen erleichtert werden, um eine optimale Anordnung von Steuerinstrumenten und Informationen für den Bediener zu bieten. Generell müssen alle nachfolgenden Eigenschaften zur Benutzbarkeit und Betriebssicherheit des Systems beitragen.

16.1. Topologische Einfärbung

Der Status jeder Stromleitung im Netzwerk muss zu jeder Zeit klar erkennbar sein. Dies ist von höchster Bedeutung, wenn verschiedene Spannungsebenen verwaltet werden müssen.

Um für einen klaren Überblick über die Schaltzustände (Leitungen, Schalter) innerhalb der Anlageninstallation zu sorgen, muss das System die automatische Einfärbung von Komponenten unterstützen.

Das System muss mit einer integrierten und standardisierten Option für die Anwendung einer automatischen topologischen Einfärbung ausgestattet sein. Basierend auf einem einfachen Liniendiagramm (erstellbar in einem WYSIWYG-Editor) sollte das System dazu fähig sein, die Rastertopologie selbst zu berechnen und die jeweiligen Elemente entsprechend ihrem aktuellen Zustand in den konfigurierten Farben einzufärben. Verschiedene Zustände wie unversorgt, versorgt (einfach, mehrfach, gesichert), ungültig, unbekannt, Erdschluss oder Kurzschluss etc. müssen ohne zusätzlichen Programmier- oder Skriptaufwand eingefärbt werden.

Es soll möglich sein, durch einfaches "Kopieren und Einfügen" Detailansichten von Netzabschnitten (Gerätedarstellungen, Feeder Bay-Detailansichten) zu erstellen. Die so erstellte detaillierte Abschnittsansicht kann die Leitungszustände gleichzeitig zum ursprünglichen (Quell-)Modell anzeigen. Zusätzlich können Steuerbefehle auch über das Detailabbild ausgegeben werden.

16.2. Überprüfung der Topologie

Basierend auf den Definitionen der topologischen Einfärbung muss das System dazu fähig sein, Schutzmechanismen anzubieten (z.B. Schaltgerät-Verriegelung, Vermeidung gefährlicher Schaltzustände, Ermittlung unversorgter Bereiche, Ermittlung undefinierter Schaltzustände und Vergrößerung undefinierter Bereiche). Es muss möglich sein, Informationen über potenziell gefährliche Schaltzustände an den Bediener weiterzuleiten und dementsprechend bestimmte Schalthandlungen zu blockieren. Es muss jedoch für autorisierte Bediener möglich sein, beliebige Verriegelungsbedingungen bewusst zu quittieren.

16.3. Befehlsgabe

Das System muss Möglichkeiten zur sicheren Befehlsgabe anbieten. Es muss möglich sein, zweistufige sowie zweihändige Befehlsgaben zu konfigurieren, mit Rücksichtnahme auf protokollspezifische Funktionen wie „Select and Execute" (IEC 60870) oder "Select before Operate" (IEC 61850).

Um eine unerlaubte Schalthandlung durch den Bediener zu verhindern, wird eine Verriegelungslogik zu jeder Befehlsgabe hinzugefügt werden. Die Verriegelungslogik muss mithilfe von zulässigen Schaltzustandswerten berechnet werden oder über die Einbeziehung des topologischen Status der Leitungen.

Es muss möglich sein, verschiedene Benutzerebenen für jede Befehlsgabehandlung separat zu konfigurieren.

16.4. Schaltfolgen

Das SCADA-System muss die Möglichkeit bieten, ein Stapelverarbeitungsprogramm mit Schaltbefehlen zu definieren. Die Programmierung muss während der Runtime des Systems gemacht werden können. Ein grafischer Editor für die Programmierung wird benötigt.

Die Programmierung muss über eine Teach-in-Methode, also einen Makrorecorder, erfolgen, während der SCADA-Systemarbeitsplatz nicht mit dem aktiven Prozess verbunden ist.

16.5. Betriebssicherheit, Sichere Betriebssperre

Das System muss integrierte Mechanismen zur Vermeidung von inkonsistenten, unkontrollierten und potenziell gefährlichen Situation anbieten.

Energieinfrastruktur-Systeme repräsentieren üblicherweise stark verteilte Anwendungen, auf die von mehreren Standorten aus zugegriffen werden kann. Darum müssen Konkurrenzsituationen sowie Interferenz (mehrere Leitstände wollen gleichzeitig denselben Abschnitt steuern) strengstens vermieden werden.

Außerdem muss das System im Falle einer Wartung vor Ort eine Transition in einen „Wartung vor Ort"-Modus ermöglichen. Dementsprechend muss die gewartete Anlage nur durch den Leitstand vor Ort gesteuert werden können (dedizierte Station oder Panel). Alle Versuche, diesen Abschnitt von einem Fernzugriffspunkt zu steuern, müssen strengstens (sicher) verweigert werden.

Das Sperren und Entsperren eines Schaltgeräts darf nur durch einen in der Benutzerverwaltung des SCADA-Systems authentifizierten Benutzer und mit einem bestimmten Sperrcode möglich sein. Es muss möglich sein, dass mehrere Benutzer eine Sperre auf demselben Schalter einstellen. Wenn eine oder mehrere Sperren aktiv sind, dürfen keine Schalthandlungen mehr möglich sein. Nur nachdem alle Sperren entfernt wurden, darf das Schaltgerät wieder bedienbar sein.

Außerdem muss es die Möglichkeit geben, in einen Prozess von nur einem Computer einzugreifen. Alle anderen Stationen im Netzwerk müssen als Beobachter online bleiben. Falls notwendig, müssen Sie in der Lage sein, operativen Zugriff zu beantragen.

16.6. Höherwertige Netzleitfunktionen

16.6.1. Erkennung von Erdschluss und Kurzschluss

Abhängig von dem topologischen Modell und den Informationen von Schutzgeräten (Distanzschutz, zeitabhängiger Überstromschutz) muss das System dazu fähig sein, fehlerhafte Netzzustände zu erkennen, wie z.B. Erdschluss oder Kurzschluss. Außerdem muss das System diese Informationen zusammenstellen und eine (mögliche) Fehlerortung durchführen können. Dementsprechend muss das System den Bediener informieren und die betroffenen Netzwerkteile hervorheben (Linienabschnitt).

Im Falle von Situationen mit einer hohen Erdschlusswahrscheinlichkeit (z.B. während Gewittern) muss es möglich sein, das topologische Modell vorübergehend von dem Alarmierungsmechanismus zu entkoppeln, um es dem Bediener zu ermöglichen, die Situation zu überprüfen, z.B. durch Testschaltungen, um zu verifizieren, dass ein Erdschluss tatsächlich ansteht. Dadurch wird auch eine präzisere Fehlerortung ermöglicht. Nach dem Abschluss der Überprüfungen muss der Bediener wieder mit der Überwachung und Visualisierung modellbasierter Fehler fortfahren können.

Die oben beschriebene Funktionalität muss identischer Weise für den Umgang mit Kurzschlüssen verfügbar sein.

16.6.2. Impedanz basierte Fehlerortung

Bei der Verwendung von Schutzgeräten mit integrierter Fehlerortungsfunktionalität soll es möglich sein den Messwert (Ohm, km, %) der Fehlerortung auszulesen und den Fehlerort im Einlinienschaltbild des SCADA-Systems lagerichtig anzuzeigen.

16.6.3. Lastflussberechnung

Das System soll eine Lastflussberechnung eines vermaschten Netzes durchführen können. Dabei soll das Netz zwar komplex, aber symmetrisch berechnet werden. Die berechneten Werte sollen im Einlinienschaltbild dargestellt werden oder als Eingangswerte anderer Berechnungen dienen.

Die Lastflussberechnung soll auch dazu genutzt werden können, bei Schalthandlungen darauf hinzuweisen, ob diese eine Überlastung eines Netzelements zur Folge hätten (Vorausberechnung). Außerdem sollen alle Netzelemente in einer Liste aufgeführt und deren n-1 Verhalten dargestellt werden.

16.6.4. State Estimator

Das System soll die Möglichkeit bieten, eine Lastflussberechnung durchzuführen, auch wenn nicht genug gemessene Daten verfügbar sind. Dies geschieht typischerweise mit einem State Estimator, der die nötigen Werte für die Lastflussberechnung liefert.

COPA-DATA ist ein unabhängiger Hersteller von Software für Industrie- und Energieautomatisierung. Seine Produkte werden weltweit in der Fertigungsindustrie und in der Energiewirtschaft zur automatisierten Steuerung, Überwachung und Optimierung von Maschinen, Anlagen und Stromnetzen eingesetzt. COPA-DATA kombiniert fundierte Erfahrung in der Automatisierung mit den neuen Möglichkeiten der digitalen Transformation und unterstützt Kunden dabei, ihre Strategie einfacher, schneller und zielgerichteter umzusetzen.

17. Module Type Package (MTP) modulare Automatisierung

Der Standard VDI/VDE/NAMUR 2658 wird vollständig durch das System unterstützt. Diese Unterstützung beinhaltet:

- **Erstellung/Bearbeitung von MTP Dateien**
  - Mittels grafisch geführtem User Interface ist eine einfache Konfiguration von MTP Dateien möglich. Es können bestehende Dateien eingelesen bzw. neue Dateien basierend auf dem Standard erstellt werden.
  - Update der MTP Version: Das System erkennt die verwendete MTP Version und bietet ein Update auf die neueste Version an. Alle Änderungen werden in einem Protokoll festgehalten.
  - Import von MTP Dateien aus einem 3rd Party Marketplace wird unterstützt
- **Validierung von MTP Dateien**
  - Während der Erstellung einer neuen MTP-Datei wird live auf Konformität zum Standard VDI/VDE/NAMUR 2658 geprüft und ggf. sofort auf Fehler aufmerksam gemacht. Ein kompletter Validierungs-Report kann für jedes MTP erstellt werden.
- **Verwaltung von Instanzen**
  - Für jedes MTP Template müssen Instanzen für jedes physische Gerät erzeugt und verwaltet werden können.
  - Jedes physisch angelegte Modul muss mit einem Status versehen werden können. Vorhandene Status werden in einer zentralen Liste gepflegt.
- **Prozess Orchestrierung**
  - In einer grafischen Oberfläche werden vorhandene MTP Instanzen miteinander logisch per Drag & Drop verschaltet. Das HMI Layout ist konfigurierbar.
  - Das Kontrollsystem wird automatisch erzeugt incl. ISA 88 Grundfunktionen, Kommunikations-Einstellungen, Tags, Bilder, …
  - Import/Export eines POL Projektes ist möglich

COPA-DATA HEADQUARTERS
Ing. Punzenberger COPA-DATA GmbH
Karolingerstr. 7b
5020 Salzburg, Austria
Tel: +43 662 431002 0
Fax: +43 662 431002 33
sales@copadata.com
www.copadata.com