HAW HAMBURG

BACHELOR THESIS
Lukas Raudszus

Entwicklung eines digitalen Zwillings einer Hub-Drehbrücke zur Simulation und Testung der Steuerungssoftware

FAKULTÄT TECHNIK UND INFORMATIK
Department Informations- und Elektrotechnik

Faculty of Engineering and Computer Science
Department of Information and Electrical Engineering

HOCHSCHULE FÜR ANGEWANDTE WISSENSCHAFTEN HAMBURG
Hamburg University of Applied Sciences

Lukas Raudszus

Entwicklung eines digitalen Zwillings einer Hub-Drehbrücke zur Simulation und Testung der Steuerungssoftware

Bachelorarbeit eingereicht im Rahmen der Bachelorprüfung
im Studiengang *Bachelor of Science Elektro- und Informationstechnik*
am Department Informations- und Elektrotechnik
der Fakultät Technik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer: Prof. Dr. Frerk Haase
Zweitgutachter: Dipl.Ing. Björn Foitschik-Weber

Eingereicht am: 29. November 2024

Lukas Raudszus

Thema der Arbeit

Entwicklung eines digitalen Zwillings einer Hub-Drehbrücke zur Simulation und Testung der Steuerungssoftware

Stichworte

Digitaler Zwilling, Hub-Drehbrücke, SIMIT, Softwarearchitektur

Kurzzusammenfassung

In der vorliegenden Bachelorarbeit wird die Konzeption, Entwicklung und Implementierung eines digitalen Zwillings für eine Hub-Drehbrücke behandelt. Ziel des digitalen Zwillings ist es, dass die Steuerungssoftware der Brücke an ihn getestet und optimiert werden kann, bevor das reale System errichtet wird. Für einen strukturierten Aufbau der Simulation wird eine Softwarearchitektur entworfen, die es bei der Erstellung zu beachten gilt.

Lukas Raudszus

Title of Thesis

Development of a digital twin of a lift swing bridge bridge for simulation and testing of the control software

Keywords

Digital twin, lift and swing bridge, SIMIT, software architecture

Abstract

This bachelor's thesis covers the design, development and implementation of a digital twin for a lift and swing bridge. The aim of the digital twin is to test and optimize the control software of the bridge before the real system is built. A software architecture is designed for a structured structure of the simulation, which must be taken into account during creation.

Inhaltsverzeichnis

Abbildungsverzeichnis

Abkürzungen

HMI Human Machine Interface - Mensch-Maschine-Schnittstelle.

I/O Ein-/Ausgabe.

IP Internet Protocol - Internet Protokoll.

IRT Isochronous Real-Time.

LWL Lichtwellenleiter.

PNO Profinet-Nutzerorganisation.

RT Real Time - Echtzeit.

SPS Speicherprogrammierbare Steuerung.

TCP Transmission Control Protocol - Übertragungssteuerungsprotokoll.

1 Einleitung

1.1 Einführung

In der heutigen industriellen Automatisierung ist die Verwendung einer speicherprogrammierbaren Steuerung, oder auch kurz als „SPS“ bezeichnet, nahezu unverzichtbar. Eine SPS wird zur Überwachung und Steuerung verschiedenster Maschinen und Anlagen verwendet. Sie findet in vielen unterschiedlichen Anwendungsbereichen Verwendung, von der Fertigungsindustrie bis hin zu Infrastrukturprojekten wie Brücken und Flughäfen. Eine SPS wird in fast jeder modernen automatisierten Anlage verwendet. Um eine SPS für solche Anwendungen einsetzen zu können, muss diese jedoch programmiert werden. Dabei muss das Programm auf die individuellen Anforderungen und Systemabläufe der Anlage, sowie die Wünsche des Kunden, zugeschnitten werden.

Die Entwicklung des Programms ist ein zeitintensiver Prozess, der eine sorgfältige Planung und Implementierung bedarf. Häufig stehen die Programmierer dabei jedoch vor dem Problem, dass eine endgültige Prüfung und Optimierung des Programms erst nach Abschluss der Anlageninstallation erfolgen kann. Dies liegt daran, dass manche Fehler oder notwendige Anpassungen erst im laufenden Betrieb der Anlage auffallen können. Als Folge dessen kann es sein, dass das Programm potenziell über einen längeren Zeitraum hinweg nachjustiert werden muss, obwohl die Anlage bereits installiert und betriebsbereit ist. Bei großen und komplexen Projekten, bei denen die Fertigstellung der Anlage mehrere Jahre in Anspruch nehmen kann, ist es oftmals nur möglich, Teilprüfungen an bereits errichteten Teilen der Anlage durchzuführen. Zum Testen des Anlagenteils muss jedoch häufig das Programm angepasst werden, sodass nur der Teil der Anlage angesteuert wird. Dies kann jedoch ebenfalls zu Fehlern führen, da das Gesamtsystem immer noch Schwachstellen aufweisen könnte.

Um diese Herausforderung zu meistern, stellt das Konzept eines sogenannten „digitalen Zwillings“ eine vielversprechende Lösung dar. Bei einem digitalen Zwilling handelt

es sich um eine virtuelle Nachbildung eines realen Systems. Die virtuelle Nachbildung der Anlage simuliert die einzelnen Anlagenkomponenten und deren Verhalten. Ähnlich wie die reelle Anlage liest die virtuelle Anlage die SPS-Signale aus, verarbeitet diese und sendet anschließend eine entsprechende Rückmeldung an die SPS zurück. Am Beispiel eines Ventils bedeutet dies, dass bei einem Ansteuerungssignal der SPS „Ventil öffnen“ das Ventil in der virtuellen Anlage geöffnet wird. Sobald das virtuelle Ventil geöffnet ist, würde die virtuelle Anlage die Rückmeldung „Ventil geöffnet“ an die SPS zurücksenden. Die Simulation stellt also die bidirektionale Kommunikation zwischen der SPS und der Anlage nach, wodurch sich potenzielle Fehler frühzeitig finden und beheben lassen.

1.2 Motivation

Die Motivation für die vorliegende Bachelorarbeit entstand durch die erfolgreiche Zusammenarbeit mit der „Actemium Cegelec Mitte GmbH“ während meines Praxissemesters. Schon während dieser Zusammenarbeit konnte ich bereits meine ersten praktischen Erfahrungen im Projekt der Friesenbrücke sammeln.

Beim Projekt der Friesenbrücke handelt es sich um ein millionenschweres Projekt, welches sich die Actemium Cegelec Mitte GmbH sichern konnte. Ziel dieses Projekts ist der komplette Wiederaufbau der Friesenbrücke, welche sich zwischen Leer und dem niederländischen Groninge befindet und durch einen Unfall im Jahr 2015 zerstört wurde.

Seit dem Unfall wird die Zugverbindung zwischen den beiden Städten nur noch über den Schienenersatzverkehr bedient. Um den Zugverkehr wiederherzustellen, soll die Friesenbrücke daher neu errichtet werden.

Die neue Brücke, die im Zuge des Projekts entsteht, ist dabei als Hub-Drehbrücke konzipiert, welche in Europa als größte ihrer Art gelten wird. Die Konstruktion der Friesenbrücke als Hub-Drehbrücke erlaubt es, den mittleren Brückenteil anzuheben und zu drehen, wodurch eine große Durchfahrt entsteht, die selbst große Frachtschiffe passieren lässt. Diese besondere Konstruktionsweise sorgt nicht nur für eine hohe technische Anforderung an die Brückenkonstruktion, sondern auch für das komplexe Steuerungssystem.

Damit die Fertigstellung der Brücke wie geplant erfolgen kann, ist jedoch noch viel Arbeit erforderlich. Ein wesentlicher Teil davon ist die Programmierung des Steuerungssystems.

Da es noch länger dauern wird, bis der Bau der Brücke abgeschlossen ist, gestaltet es sich als Herausforderung, ein Steuerungssystem für eine Anlage zu entwerfen, die sich noch in der Bauphase befindet.

Um dieses Problem zu lösen, wird für die Brücke ein digitaler Zwilling erstellt, welcher die Anlage hardwareseitig simuliert. Die virtuelle Brücke, die dabei entsteht, soll das Verhalten der Anlage und der einzelnen Komponenten simulieren, wodurch es den Programmierern möglich ist, das Steuerungsprogramm frühzeitig zu entwickeln und zu testen. Zusätzlich soll der digitale Zwilling nach Fertigstellung der Brücke zur Schulung des Bedienpersonals dienen, da diese durch den Zwilling den Umgang mit der Steuerungssoftware lernen können, ohne das reelle System zu gefährden.

Zusammenfassend ist die Motivation für die Erstellung dieser Bachelorarbeit in der Notwendigkeit zur Entwicklung eines Steuerungssystems, das bereits während der Planungs- und Bauphase der Anlage entwickelt und getestet werden soll, begründet. Die Entwicklung des digitalen Zwillings stellt somit einen wichtigen Beitrag für das Projekt der Friesenbrücke dar und könnte wegweisend für zukünftige Projekte mit ähnlichen Anforderungen sein.

1.3 Aufgabenstellung

Ziel dieser Bachelorarbeit ist es, einen digitalen Zwilling der oben genannten Friesenbrücke zu erstellen. Dabei liegt der Fokus des Zwillings auf der Nachstellung der Betriebsfunktionen und korrekten Nachbildung der I/O-Kommunikation zwischen dem Steuerungsprogramm der SPS und der Brückenanlage. Der digitale Zwilling soll dabei nicht nur zur Testung der Steuerungssoftware dienen, sondern im späteren Verlauf auch für die Schulung des zukünftigen Wartungspersonals verwendet werden.

Für die Erstellung des digitalen Zwillings wird die Simulationssoftware SIMIT der Firma Siemens genutzt. Sie ermöglicht uns, die reellen Komponenten und Anlagenteile der Brücke virtuell nachzubauen und diese zu simulieren. Die virtuelle Anlage sollte dabei möglichst modular aufgebaut sein, um Anlagenteile oder Komponenten bei Bedarf einzeln testen zu können.

Zur Nachbildung der einzelnen Komponenten werden in SIMIT Makros verwendet. Jedes dieser Makros dient als Bausteinvorlagen der einzelnen Komponenten wie z.B. der Ventile, Zylinder oder Motoren. Durch die logische Verknüpfung der einzelnen Makros untereinander wird anschließend die Funktion der reellen Brücke nachgebildet. Nach erfolgreicher Nachbildung der einzelnen Komponenten und Brückenfunktionen kann das Steuerungssystem am digitalen Zwilling getestet werden.

1.4 Aufbau der Arbeit

Die Struktur dieser Bachelorarbeit ergibt sich aus den in der Abbildung 1.1 dargestellten Themen. Die Arbeit teilt sich dabei in einen Grundlagenteil, den Hauptteil und den Schlussteil.

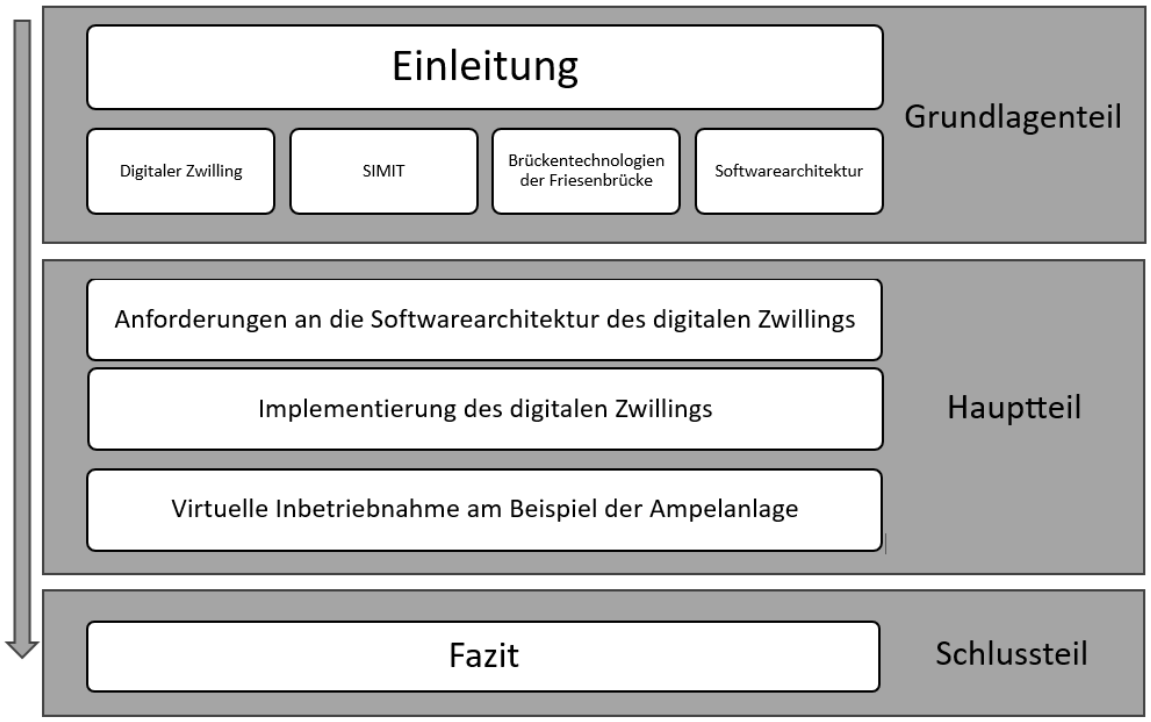

Abbildung 1.1: Aufbau der Bachelorarbeit

Nach der Einleitung werden im Grundlagenteil die theoretischen Konzepte eines digitalen Zwillings erklärt. Dabei wird zunächst auf die Definition und die funktionalen Vorteile

des digitalen Zwillings eingegangen. Anschließend wird die Softwareumgebung SIMIT vorgestellt, die für die virtuelle Inbetriebnahme der Friesenbrücke verwendet wird, bevor detailliert auf die verwendete Brückentechnologie und Netzwerktechnik eingegangen wird. Abschließend wird im Grundlagenteil noch einmal das Thema Softwarearchitektur behandelt, wobei der Nutzen und die allgemeine Struktur dieses Konzepts erläutert werden. Der Hauptteil beschreibt die technische Umsetzung des Konzepts der Softwarearchitektur für unseren digitalen Zwilling. Dabei werden die spezifischen Anforderungen an die Softwarearchitektur der Simulation erläutert. Anschließend werden die funktionalen und nicht-funktionalen Anforderungen, die unsere Simulation erfüllen soll, behandelt. Es wird beschrieben, in welcher Struktur die Simulation aufgebaut wurde und über welche Schnittstellen die Simulation mit der Steuerung kommuniziert.

Danach wird die Implementierung des digitalen Zwillings anhand einiger Module und Makros schematisch dargestellt und deren Funktionsweise erklärt. Die Umsetzung der Module und Makros erfolgt dabei auf Basis der erarbeiteten Architektur des digitalen Zwillings. Nach der Implementierung wird noch einmal das Vorgehen für die virtuelle Inbetriebnahme des Steuerungsprogramms anhand des Beispiels der Ampelanlage beschrieben.

Schlussendlich werden im Schlussteil die behandelten Themen und Ergebnisse der Bachelorarbeit in Form eines Fazits reflektiert.

2 Digitaler Zwilling

Eine allgemeingültige Definition gibt es für den digitalen Zwilling noch nicht, jedoch lässt sich aus den vielen theoretischen Ansätzen eine Kernaussage formulieren. Ein digitaler Zwilling ist die virtuelle Darstellung eines realen Objekts oder Systems. Die ursprünglich in der Fertigungs- und Raumfahrtindustrie entwickelte Technologie wird heute in zahlreichen Bereichen, wie der Medizin, Infrastrukturprojekten und im Bauwesen, eingesetzt. Ein digitaler Zwilling erlaubt die Simulation und Analyse eines physischen Systems sowie seine Testung unter realistischen Bedingungen, ohne auf das physische Objekt selbst angewiesen zu sein.

2.1 Definition und Konzept

Ein digitaler Zwilling ist eine virtuelle Nachbildung, die dazu dient, alle relevanten Prozesse, Eigenschaften und Funktionen eines Systems möglichst realitätsnah abzubilden. Dies wird durch eine komplexe Simulation des Objekts erreicht, welche oft auch in Zusammenspielen mit Echtzeitdaten und KI-basierten Prognosemodellen erfolgen kann. Der digitale Zwilling übernimmt dabei die funktionalen und physischen Eigenschaften des Systems. Dazu zählen unter anderem:

- Funktionale Eigenschaften:
  Bewegungsabläufe, Prozessabläufe, Steuerlogik und die Reaktionen des Systems
- Physische Eigenschaften:
  Materialeigenschaften, Wettereinwirkungen, Druck, Temperatur und mechanische Belastung

Durch die umfassende Modellierung dieser Eigenschaften ist es dem Entwickler des digitalen Zwillings möglich, eine realistische Umgebung zu schaffen, in der das System analysiert und optimiert werden kann, ohne auf das reelle System angewiesen zu sein.

2.2 Funktion und Vorteil eines digitalen Zwillings

Ein digitaler Zwilling bietet besonders während der Entwicklungsphase eines Systems eine Vielzahl von Vorteilen. Diese sind in der Automatisierungsbranche und hier speziell in der Steuerungstechnik von großem Interesse. Mithilfe eines digitalen Zwillings lassen sich Funktionen wie die Simulation, Überwachung, Prognose und Fehleranalyse eines Systems effizient realisieren.

Simulation

Ein digitaler Zwilling ermöglicht die Simulation von physikalischen, mechanischen und softwarebezogenen Prozessen. Damit bietet der digitale Zwilling die Möglichkeit, alle möglichen Steuerungs- und Betriebsfunktionen eines Systems zu testen, bevor dieses reell existiert.

Überwachungsfunktion

Durch einen digitalen Zwilling lässt sich eine Überwachungsfunktion realisieren, welche alle Zustände und Daten der Anlage in Echtzeit überwacht. Während der Überwachung werden ungewöhnliche Betriebszustände und Abweichungen erkannt. Der digitale Zwilling kann in diesem Anwendungsfall mit einer Vielzahl von Betriebsdaten wie Druck, Temperatur, Geschwindigkeiten und Positionen der Komponenten versorgt werden, um das Verhalten der reellen Komponente analysieren zu können. Durch die Echtzeitüberwachung wird sichergestellt, dass Abweichungen, die auf potenzielle Störungen hindeuten könnten, frühzeitig erkannt und gemeldet werden.

Prognosefunktion

Durch einen digitalen Zwilling können zukünftige Zustände einer realen Anlage simuliert werden, um so einen Einblick auf das zukünftige Verhalten der Anlage zu erhalten. Dabei können durch den Einsatz von KI-Algorithmen und Betriebsdaten der Anlage Muster erkannt werden und somit Aussagen über das zukünftige Verhalten von Komponenten und Systembereichen getroffen werden. Eine solche Prognosefunktion ist besonders im Wartungsbereich von Anlagen nützlich, da sie es ermöglicht, den Ausfall bestimmter Komponenten vorherzusagen.

Am Beispiel der Friesenbrücke könnten Prognosealgorithmen dazu verwendet werden, um die Antriebe oder die Hydraulik der Brücke zu simulieren und diese auf Verschleißindikatoren wie Überhitzung, Vibration oder zunehmenden Energieverbrauch zu überwachen. Anhand dieser Daten könnten Wartungsintervalle geplant werden und Stillstandszeiten durch eine präventive Wartung minimiert werden.

Fehleranalyse

Fehlerfälle können im digitalen Zwilling nachgestellt werden, um Lösungen zu testen, bevor sie im reellen System angewendet werden.

3 SIMIT

3.1 Allgemein

SIMIT ist eine von Simens entwickelte Simulationssoftware, die für die Entwicklung digitaler Zwillinge, virtuelle Inbetriebnahmen und Schulungen in Automatisierungsprojekten eingesetzt wird. Durch sie können Simulationen für industrielle Systeme erstellt und getestet werden, bevor die physische Anlage fertiggestellt ist. In der Software können alle Ebenen eines Automatisierungssystems, von der Steuerung bis zur Feldebene, realistisch nachgebildet werden.

Laut der Siemens AG zeichnet sich die Plattform durch ihre hohe Benutzerfreundlichkeit, vereinfachte Simulationsmodellierung, hohen Leistungsfähigkeit sowie Offenheit und Flexibilität aus. Eine virtuelle Inbetriebnahme mit SIMIT trägt zur Steigerung der Effizienz in der Softwareentwicklung und Erhöhung der Planungssicherheit bei.

3.2 Programmaufbau

Der Grundaufbau des Programms, welcher in Abbildung 3.1 zu sehen ist, umfasst mehrere Schlüsselelemente, die zur effizienten Steuerung und Bearbeitung von Projekten dienen.

Die Menüleiste (1) bietet dabei Zugang zu den verschiedenen Funktionen und Einstellungen von SIMIT, wobei spezielle häufig verwendete Funktionen über extra Schaltflächen bereitgestellt werden. Diese beinhalten Funktionen wie z.B. Speichen, Simulation starten und Simulation stoppen.

In den Task-Cards (2) werden Objekte wie Bibliothekskomponenten, Controls und Grafikobjekte sowie Makros und die angebundenen Signale aufgelistet, die im aktuellen Projekt

verwendet werden können.

Der Abschnitt (3) stellt den Hauptarbeitsbereich dar. Hier werden die Editoren zur Bearbeitung des Projekts geöffnet. Der Arbeitsbereich lässt sich teilen, wodurch zwei Editoren neben- oder untereinander im Arbeitsbereich geöffnet werden können

Die Projektnavigation (4) stellt das aktuelle Projekt in einer Baumansicht dar.

Im Eigenschaftsfenster (5) werden detaillierte Informationen zum ausgewählten Objekt angezeigt und deren Bearbeitung ermöglicht.

Die Editor- und Statuszeile (6) dient zum Wechseln zwischen bereits geöffneten Editoren und dem Zugang zur Portalansicht sowie der Anzeige des aktuellen Anwendungsstatus.

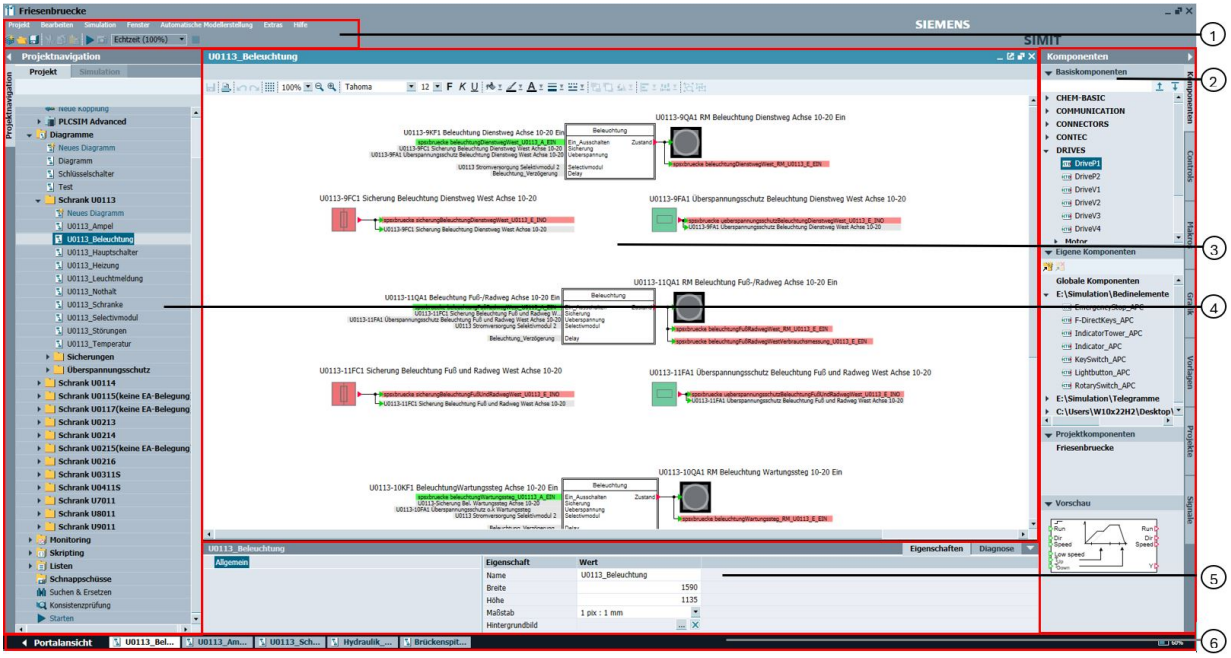

Abbildung 3.1: Bedienoberfläche von SIMIT

SIMIT ermöglicht die Erstellung einer beliebigen Anzahl an Simulationsfenstern, die sich für die Simulation aktivieren und deaktivieren lassen. Über die Starttaste können alle aktivierten Simulationsfenster auf einmal gestartet werden. Innerhalb der einzelnen Simulationsfenster werden die unterschiedlichen Anlagenteile der Brücke angelegt, die

simuliert werden sollen. Die Anlagenteile werden dabei in unterschiedlichen Fenstern erstellt, damit eine gewisse Struktur und Übersichtlichkeit im Programm bewahrt werden können. Die Funktion der einzelnen Anlagenteile wird durch Verschaltung der einzelnen Anlagenkomponenten gebildet. Hierzu können Funktionsbausteine angefertigt werden, die das Verhalten der benötigten Komponente widerspiegeln. Die genaue Struktur, in der das Programm und die Funktionsbausteine, aufgebaut werden sollen, wird im Kapitel 6 „Anforderungen an die Softwarearchitektur der Simulation„ behandelt.

4 Brückentechnologien der Friesenbrücke

Im Dezember 2015 wurde die alte Friesenbrücke, die sich bei Weener über die Ems erstreckte, durch die Kollision mit einem Frachtschiff beschädigt und zerstört. Seit diesem Vorfall ist die Zugverbindung zwischen Leer und dem niederländischen Groningen unterbrochen und wird über den Schienenersatzverkehr bedient. Um den regulären Zugverkehr wieder aufzunehmen, soll nun eine neue Friesenbrücke errichtet werden. Die neue Friesenbrücke, in Abbildung 4.1 schematisch dargestellt, ist dabei als eine 335 m lange Hub-Drehbrücke konzipiert.

Abbildung 4.1: Seitenansicht der geplanten Friesenbrücke

Der Aufbau der Hub-Drehbrücke besteht dabei im Wesentlichen aus zwei Vorlandbrücken und einem beweglichen mittleren Brückenteil. Der rund 145 m lange mittlere Brückenteil ist dabei auf einen Drehpfeiler montiert, welcher das Anheben und Drehen des mittleren Brückenteils ermöglicht und somit das Herzstück der Anlage darstellt.

Die Hydraulikanlage, die das Öffnen und Schließen der Friesenbrücke ermöglicht, besteht aus einem im Hydraulikraum des Drehpfeilers untergebrachten Hydraulikaggregat. Dieses Hydraulikaggregat steuert sechs Hydraulikzylinder an, die für das Anheben des beweglichen Brückenteils verantwortlich sind, sowie acht hydraulische Getriebemotoren, die der Brücke die Drehbewegung ermöglichen.

Das Hydraulikaggregat des beweglichen Brückenteils besteht dabei hauptsächlich aus einem Hydrauliköltank, sechs Motor-Pumpen-Einheiten, einer Nebenstromfiltereinheit, Ventilblöcken sowie Druck- und Rücklauffiltern.

Jede Motor-Pumpen-Einheit besteht aus einem 75kW-Elektromotor, einer Axialkolbenpumpe und zwei Zahnradpumpen, die direkt am Durchtrieb der Hauptpumpen angebracht sind. Dabei dient eine der Zahnradpumpen zur Schmierung der Axialkolbenpumpe, während die andere den Steuerölkreis der Hydraulikanlage versorgt. Unterschiedliche Geschwindigkeiten der Brückenbewegungen werden durch analoge Volumenstromverstellung der Hauptpumpe realisiert.

Die Ventilblöcke werden von den sechs Motor-Pumpen-Einheiten des Hauptantriebs gespeist. Durch die Ventilblöcke werden verschiedene Funktionen wie die Druckabsicherung und der drucklose Umlauf der Hydraulikanlage realisiert. Zudem sind hier Wegventile verbaut, die zur Steuerung der Zylinderbewegung sowie der Druckabsicherung der Hubzylinder dienen. Mithilfe von Sensorik werden dabei die Druckverläufe überwacht, um einen sicheren Betrieb zu gewährleisten.

Hinter jeder Hydraulikpumpe befindet sich ein Druckfilter, der dazu dient, das System vor Partikeln zu schützen, die von der Pumpe angesaugt werden können.

Weiterhin ist das Hydrauliksystem mit zwei Rücklauffiltern ausgestattet. Diese werden über einen Sensor überwacht, der bei einer Verschmutzung von 75 und 100 Prozent ein entsprechendes Warnsignal an die Steuerung gibt.

Am Tank des Hydrauliksystems ist der Nebenatomfilter montiert, welcher die Hydraulikflüssigkeit permanent im Nebenstrom über einen Filter reinigt. Zusätzlich ermöglicht das Aggregat das Befüllen und Entleeren des Tanks.

An den Enden der Brückenspitzen des beweglichen Brückenteils sind jeweils zwei weitere Hydraulikaggregat installiert, welche Zylinder ansteuern, die für die Spitzenverriegelung, den Schienenübergang und die Verriegelung des Schienenübergangs zuständig sind.

Die Spitzenverriegelung dient dazu, die Brücke in Position zu halten und ein Anheben zu verhindern. Der Schienenübergang ermöglicht ein Trennen der Schienen des beweglichen Brückenteils von den Schienen der festen Vorlandbrücken, wenn die Brücke bewegt werden soll. Um den festen Sitz der Schienen im Verkehrsfall zu gewährleisten, gibt es eine Verriegelung des Schienenübergangs, um ein ungewolltes Öffnen der Schienen zu verhindern.

4.1 Netzwerktechnik der Friesenbrücke

4.1.1 Was ist Profinet IO

Profinet IO ist eine besondere Variante der Profinet-Technologie, die zur Kommunikation zwischen industriellen Steuerungen und dezentralen Feldgeräten geeignet ist. Sie wurde von den verschiedenen Mitgliedsfirmen der Profinet-Nutzerorganisation(PNO) entwickelt und benutzt Ethernet als Basis für die Datenübertragung. Profinet IO ermöglicht eine schnelle und effiziente Übertragung von Daten in Echtzeit, was besonders in der Fertigungs- und Antriebstechnik benötigt wird. Die Technologie kombiniert die Fähigkeit zur Echtzeitübertragung mit der Möglichkeit, auch Diagnoseinformationen, Parameter und Variablen über das gleiche Netzwerk zu übertragen.

Die Profinet IO-Technologie basiert, im Gegensatz zum herkömmlichen Master-Slave-Modells von Profinet, auf einem Provider-Consumer-Modell. Im Provider-Consumer-Modell senden die Provider ihre Daten kontinuierlich, ohne auf eine explizite Anforderung durch den Kommunikationspartner zu warten, wodurch die Latenzzeiten gesenkt und die Datenübertragung beschleunigt wird.

Die Kommunikation in Profinet IO erfolgt in einem zyklischen Modus für Echtzeitdaten und einem azyklischen Modus für Konfigurations- und Diagnosedaten. Die zyklischen Daten werden dabei in einem festen Raster, dem sogenannten Buszyklus, zwischen den IO-Teilnehmern in RT-Datenformat übertragen. Übertragungen von IO-Daten werden

gegenüber herkömmlichen TCP/IP-Daten priorisiert, was eine Buszykluszeit im Millisekundenbereich ermöglicht, die für Echtzeitanwendungen unerlässlich ist. Zusätzlich zur Echtzeitkommunikation werden ebenfalls Statusinformationen übertragen, die zur Überprüfung der Gültigkeit der IO-Daten dienen.

Profinet IO unterstützt zusätzlich auch Isochronous Real-Time (IRT) Kommunikation. Eine RT-Kommunikation wird typischerweise für Steuerungsanwendungen benutzt und sorgt dafür, dass die Echtzeitdaten priorisiert übertragen werden. Die IRT erweitert diese Funktion noch um eine noch präzisere Taktung und Synchronisation, was für Anwendungen benötigt wird, die eine hochpräzise zeitliche Koordination benötigen, wie in etwa CNC-Maschinen und Roboter.

All diese Eigenschaften haben dafür gesorgt, dass sich Profinet IO schnell zu einem der führenden Standards für Industrie-Ethernet-Anwendungen entwickelt hat. Die Möglichkeit, IO-Daten sowie die Diagnose- und Parametrisierung Informationen über dasselbe Netzwerk zu übertragen, sowie die Priorisierung der Echtzeitdaten sorgen dafür, dass eine effiziente und hohe Anlagenverfügbarkeit gewährleistet werden kann.

4.1.2 Netzwerk der Friesenbrücke

Die komplexe Netzwerktechnik der Friesenbrücke setzt sich aus drei voneinander unabhängigen und redundant aufgebauten Netzwerken zusammen. Die drei separaten Netzwerke sind dabei in Brückensteuerung/ Maschinennahen Steuerung, Videotechnik und Kommunikationstechnik unterteilt.

Brückensteuerung/ Maschinennahe Steuerung

Das Netzwerk der Brückensteuerung/ Maschinennahe Steuerung dient zur Steuerung und Kontrolle der Brücke und ist von den anderen Netzwerken isoliert aufgebaut. Die Kommunikation auf der Steuerungs- und Feldebene ist über ein Profinet-Netzwerk mit Profisafe-Profil aufgebaut, um eine sichere Kommunikation zwischen allen kritischen Komponenten zu gewährleisten. Dabei werden unter anderem Bediener-PCs, Server, SPS-Steuerungen, mobile Bedienpanels, Touch-Panels sowie Leistungsschalter, Messgeräte und weiterer dezentraler Peripherien eingebunden.

Die Kommunikation des Netzwerks erfolgt in Ring-Topologie, welche redundant über LWL-Leitungsverbindungen zwischen den drei Hauptstandorten Drehpfeiler, Brückenwärterhaus und der Netzstation aufgebaut sind. Dabei bilden die zwei Netzwerkswitches, die jeweils in den Netzwerkschränken der drei Standorte verbaut sind, unseren Netzwerkring. Die Anbindung der Brückenkomponenten an den jeweiligen Standorten erfolgt durch weitere redundant aufgebaute Ringe oder über redundante Stichleitungen. Innerhalb der einzelnen Standorte werden dabei Kupferleitungen für die anzubindenden Komponenten verwendet. Zu den Komponenten, die über die Stichleitungen angebunden werden, gehören unter anderem die Bediener-PCs, Server und Touch-Panel. Die Anbindung der Stichleitungen erfolgt dabei über die Switches der jeweiligen Standorte. In Abbildung 4.2 wird der Aufbau des Netzwerks noch einmal veranschaulicht.

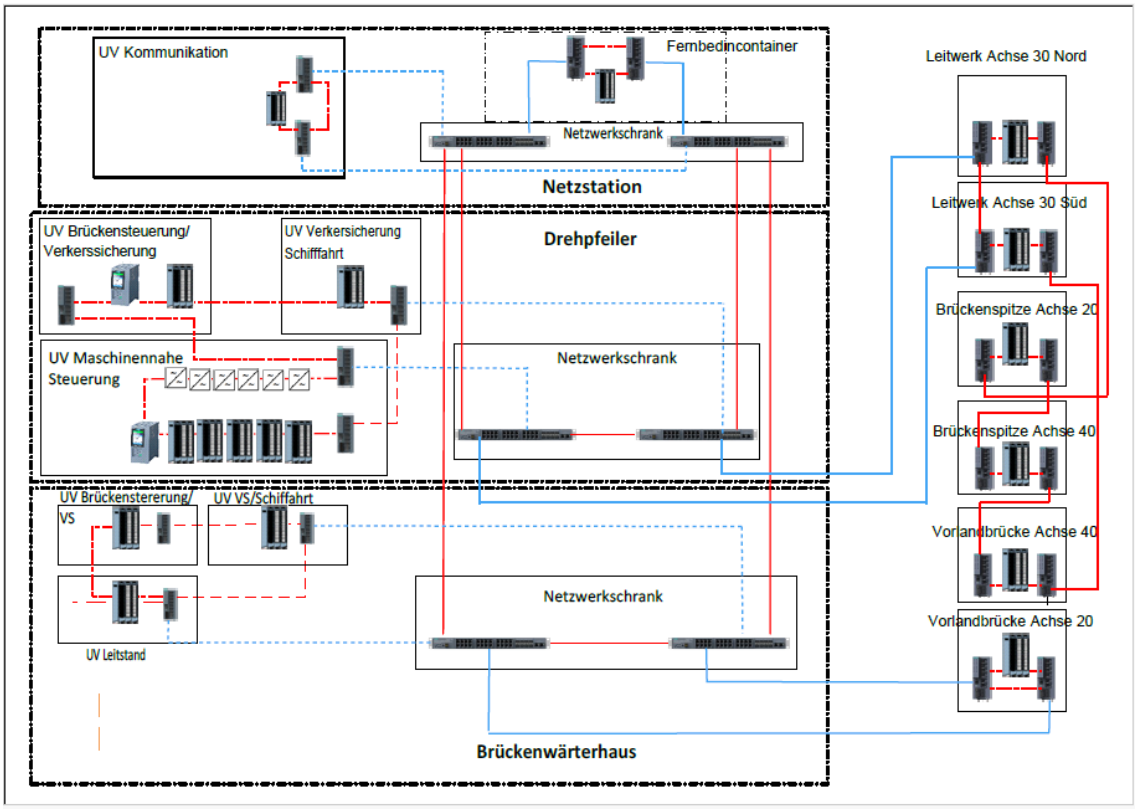

Abbildung 4.2: Netzwerk der Brückensteuerung/ Maschinennahe Steuerung

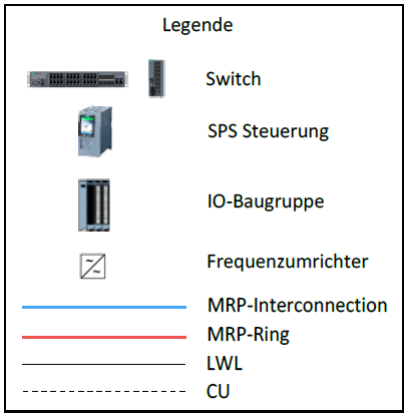

Abbildung 4.3: Legender von der Abbildung 4.2

Neben den drei Hauptstandorten gibt es noch zwei weitere Netzwerkringe. Am Netzwerkschrank des Drehpfeilers wird eins der beiden Ringsysteme über LWL-Leitungsverbindung eingebunden. Das Ringsystem ist ebenfalls redundant aufgebaut und umfasst die Peripherien des Leitwerks und der Brückenspitzen.

Am Netzwerkschrank des Brückenwärterhauses wird das zweite Ringsystem angebunden und integriert die Peripherien der Vorlandbrücke ins Netz. Auch hier ist das Ringsystem wieder redundant aufgebaut und über LWL-Leitungsverbindung angeschlossen.

Videotechnik und Kommunikationstechnik

Das Netzwerk der Videotechnik dient zur Übertragung der IP-Kammerer-Daten an den Leitstand. Mit diesen können die Gefahrenbereiche der Brücke angezeigt und überwacht werden. Das Netzwerk der Kommunikationstechnik dient der Übertragung der Sprechstellen und Lautsprecher. Sie ermöglichen die Kommunikation zwischen den Nutzern und dem Bediener der Brücke. Beide Netzwerke sind ebenfalls in einer Ringtopologie redundant aufgebaut.

Da diese beiden Netzwerke primär der Überwachung und Kommunikation der Anlage dienen und nicht direkt in der Steuerung der Anlage eingebunden sind, sind sie für die Erstellung des digitalen Zwillings nur von geringerem Interesse und werden daher nicht näher behandelt.

5 Softwarearchitektur

Eine Softwarearchitektur beschreibt die grundlegende Struktur eines Softwaresystems. Sie definiert, wie die einzelnen Komponenten eines Systems organisiert sind, sowie deren Beziehungen untereinander und mit externen Systemen. Eine durchdachte Softwarearchitektur bildet einen Entwurfsrahmen, mit dem ein Entwickler den komplexen Prozess einer Softwareentwicklung systematisch und effizient gestalten kann. Für große Projekte, wie die Erstellung eines digitalen Zwillings der Friesenbrücke, kann sie ein entscheidender Erfolgsfaktor sein, da sie sowohl die Modularität als auch die flexible Anpassung des Systems ermöglicht. Die Architektur des digitalen Zwillings legt dabei fest, wie die unterschiedlichen Module, wie die Hydraulik, die Brückenbewegung und die Sensoren, organisiert und miteinander verknüpft werden. Sie berücksichtigt dabei:

- Strukturelle Aufteilungen:
  Durch die Architektur wird das System in logische Einheiten geteilt. Diese lassen sich in Schichten betrachten und legen fest, wie die einzelnen Einheiten miteinander kommunizieren.

- Beziehungen und Interaktionen:
  Jede Komponente des Systems hat eigene, definierte Schnittstellen und Kommunikationswege, die dem Datenaustausch dienen.

- Regeln:
  In der Architektur werden Regeln definiert, die für die Konsistenz und die Qualität des Systems dienen und eine Weiterentwicklung oder Anpassung des Systems für die Zukunft erleichtern.

Für den digitalen Zwilling der Friesenbrücke dient die Softwarearchitektur als Grundlage, um die Simulation für Entwicklungs- und Testzwecke strukturiert aufzubauen. Bevor jedoch auf die speziellen Anforderungen an die Softwarearchitektur des digitalen Zwillings eingegangen wird, betrachten wir die Struktur und Zielsetzung einer Softwarearchitektur im Allgemeinen.

5.1 Zielsetzung und Anforderungen

Eine Softwarearchitektur verfolgt viele unterschiedliche Ziele, die besonders im industriellen Anwendungsbereich von Bedeutung sein können. Dabei werden im Allgemeinen häufig die folgenden Ziele verfolgt:

- Modularität und Wiederverwendbarkeit:
  Eine Softwarearchitektur sollte so entwickelt werden, dass eine klare Modularität sichergestellt ist, indem das System in logische Einheiten unterteilt wird. Diese Einheiten oder Module sollen so gestaltet werden, dass sie gezielt Anpassungen erlauben und für zukünftige Projekte wiederverwendet werden können. Die modulare Entwicklungsweise kann somit bei zukünftigen Projekten Entwicklungszeiten und Kosten sparen.

- Erweiterbarkeit:
  Eine gute Softwarearchitektur ermöglicht dem Entwickler die Bearbeitung bestehender Funktionen und das Hinzufügen neuer Funktionen, ohne das gesamte System neu entwickeln zu müssen. Dies ist insbesondere bei langfristigen Projekten entscheidend, da sich die Anforderungen an die Anlage während einer langen Entwicklungsphase häufig ändern können.

- Zuverlässigkeit:
  Das System muss in der Lage sein, zuverlässig auf Eingaben reagieren zu können und die korrekten Ausgangssignale generieren. Dies ist besonders für sicherheitskritische Funktionen wichtig. Im industriellen Kontext kann dies auch bedeuten, dass auf Eingaben in Echtzeit reagiert werden muss, um ein zuverlässiges Verhalten des Systems zu gewährleisten.

- Wartbarkeit:
  Die Softwarearchitektur sollte die zukünftige Erweiterung der Simulation auch nach Fertigstellung der Anlage erleichtern. Für einen digitalen Zwilling könnte das bedeuten, dass alle Module und Schnittstellen klar definiert sein müssen, um den Aufwand für zukünftige Änderungen zu reduzieren.

- Transparenz:
  Eine Softwarearchitektur sollte dabei helfen, ein System verständlicher und weniger komplex darzustellen, um den Entwicklern die Fehlerbehebung zu erleichtern.

5.2 Grundlegende Elemente einer Softwarearchitektur

Eine Softwarearchitektur besteht in den meisten Fällen aus einem mehrschichtigen Modell, das die einzelnen Aufgabenbereiche klar voneinander trennt. Die einzelnen Schichten umfassen dabei die Präsentationsschicht, die Geschäftslogik, die Integrationsschicht, sowie die Datenbankschicht. Jede der einzelnen Schichten spielt dabei eine spezifische Rolle in der Struktur des gesamten Systems.

Präsentationsschicht

Die Präsentationsschicht stellt die Schnittstelle zwischen dem Benutzer und dem System dar. Sie dient der Visualisierung der Daten und Ergebnisse und ermöglicht dem Bediener, den Zugriff und die Interaktion mit dem System. Die Präsentationsschicht umfasst meistens grafische Oberflächen und Bedienelemente. Im Fall der Friesenbrücke wird diese durch das HMI und einzelner Schaltflächen in der Simulation dargestellt. Die Präsentationsschicht umfasst dabei auch andere Funktionen, wie die Darstellung von Betriebszuständen, Warnmeldungen, sowie der Brückenbewegungen.

Geschäftslogik

Die Geschäftslogik kann als Kern des gesamten Systems betrachtet werden. Hier werden die funktionalen Elemente des gesamten Systems definiert. Diese umfassen alle Abläufe und Regeln, die während des Betriebs des Systems erforderlich sind. Für den digitalen Zwilling der Friesenbrücke wird in der Geschäftslogik festgelegt, wie das System auf die Steuerungssignale der SPS reagiert und welche Rückmeldungen an die SPS oder die Benutzeroberfläche zurückgesendet werden sollen. In der Geschäftslogik wird also die Logik des Systems definiert, sie stellt somit sicher, dass die Simulation auf Steuerungsbefehle und Störungen angemessen reagiert.

Integrationsschicht

In der Integrationsschicht wird die Kommunikation zwischen den einzelnen Modulen des Systems und den externen Systemen definiert. Im Fall unseres digitalen Zwillings wird hier dafür gesorgt, dass der Datenaustausch zwischen der Simulation, der SPS und der HMI reibungslos funktioniert.

Datenbankschicht

Die Datenbankschicht dient der Speicherung aller relevanten Daten. Die Schicht kann dabei als zentrale Datenquelle betrachtet werden, auf die sowohl von der Geschäftslogik als auch der Präsentationsschicht zugegriffen wird.

5.3 Sichten der Architektur

Für die Dokumentation der Softwarearchitektur eines Systems wird dieses häufig aus verschiedenen Sichten betrachtet, um seine Komplexität zu reduzieren. Da die Anforderungen an die Softwarearchitektur aber vom jeweiligen System abhängig sind, muss nicht jedes System aus allen Perspektiven betrachtet werden. Es wird unter dabei den folgenden vier Sichtweisen unterschieden.

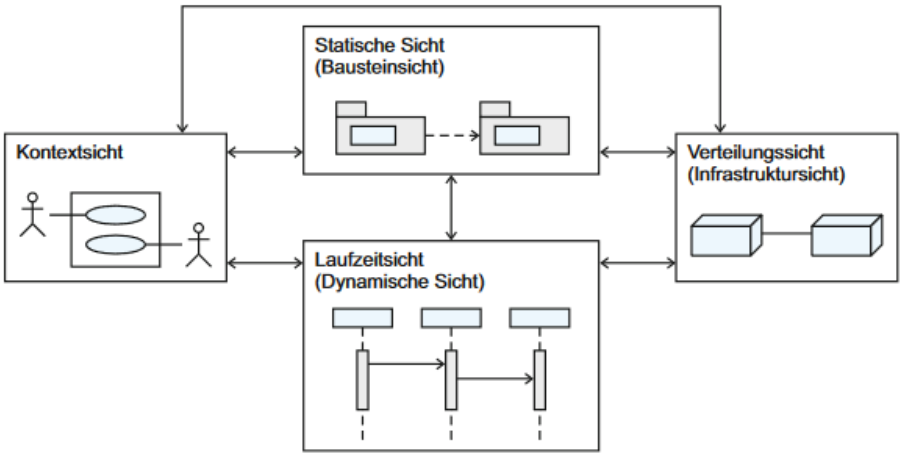

Abbildung 5.1: Sichten auf eine Softwarearchitektur

Kontextabgrenzung

In der Kontextabgrenzung werden der Umfang und die Schnittstellen des Systems definiert. Hierzu wird das System sowie alle externen Verbindungen und Interaktionen mit Benutzern dargestellt. Durch die Kontextsicht wird das System als Ganzes betrachtet, wobei die interne Funktion des Systems nicht berücksichtigt wird. Es werden lediglich alle Ein- und Ausgangsschnittstellen dokumentiert und betrachtet. Diese Sicht dient also

ausschließlich zur Betrachtung der Schnittstellen und Beziehungen zwischen dem System und der Außenwelt.

Bausteinsicht

Die Bausteinsicht dient zur Darstellung der internen Struktur des Systems. Sie beschreibt den Aufbau des Gesamtsystems und wie dieses in einzelne Module und Komponenten aufgeteilt ist. Jede Komponente und jedes Modul besitzt eine klar definierte Schnittstelle, die es anderen Komponenten oder Modulen ermöglicht, auf ihre Funktionen zuzugreifen.

In der Bausteinsicht wird verdeutlicht, wie Daten zwischen den einzelnen Modulen ausgetauscht werden. Die Sicht ist für das System somit von großer Bedeutung, da sie für die Modularität und Flexibilität zuständig ist. Die modulare Strukturierung erlaubt es dem System, dass die einzelnen Module und Komponenten unabhängig voneinander entwickelt und getestet werden können. Weiterhin erleichtert die Strukturierung das Warten und Anpassen von bereits bestehenden Teilen des Systems sowie dessen Erweiterung.

Laufzeitsicht

Die Laufzeitsicht betrachtet das Verhalten der Module während der Simulation und zeigt, wie das System auf die Steuerungsbefehle reagiert. Die Steuerungsbefehle werden an die entsprechenden Module und Komponenten weitergeleitet, die dann die gewünschten Abläufe und Funktionen innerhalb des Systems ausführen und eine entsprechende Rückmeldung erzeugen.

Verteilungssicht

In der Verteilungssicht wird die physische Verteilung der Softwarekomponenten über die verschiedene Systeme oder Rechner hinweg dargestellt.

5.4 Wie entsteht eine Softwarearchitektur

Die Entstehung einer Softwarearchitektur ist ein vielschichtiger und iterativer Prozess, der idealerweise in den frühen Phasen eines Projekts beginnt. Jede Phase des Prozesses baut auf der vorherigen auf und muss den Anforderungen des Systems entsprechen.

1. Anforderungen analysieren

Den Grundstein jeder Softwarearchitektur bilden die Anforderungen an das System. Diese lassen sich wie folgend unterteilen:

- Funktionale Anforderungen:
  Die funktionalen Anforderungen beschreiben, was das System leisten soll.
- Nichtfunktionale Anforderungen:
  Nichtfunktionale Anforderungen beschreibendie Qualitätsmerkmale, wie Wartbarkeit, Zuverlässigkeit und Erweiterbarkeit.
- Randbedingungen:
  In den Randbedingungen werden Faktoren wie Budget, der Entwicklungszeiten und Vorschriften festgehalten.

2. Technologische auswahl und Architekturstiel

Auf Basis der Anforderung trifft der Architekt Entscheidungen, die das gesamte System prägen. Diese umfassen:

- Technologiewahl:
  Es wird entschieden, welche Programmiersprache, welches Framework oder was für eine Infrastruktur verwendet wird.
- Architekturstil:
  Ein geeignetes Architekturmodell muss gewählt werden, wie eine Schichtenarchitektur oder ein Client-Server-Modell.
- Designprinzip:
  Das Designprinzip des Systems, wie z.B. ein modularer Aufbau des Systems, muss festgelegt werden.

3. Systementwurf

Nachdem die grundlegenden Eigenschaften festgelegt wurden, wird die eigentliche Architektur entworfen. Dabei liegt der Fokus auf dem Design der Systembausteine.

- Aufteilung des Systems
  Das System wird in Module oder Komponenten zerlegt, die klar definierte Aufgaben übernehmen
- Definition der Schnittstellen
  Jede Komponente erhält eine klar definierte Schnittstelle, über die sie mit anderen Komponenten kommuniziert.
- Datenfluss
  Es wird entschieden wie, der Datenfluss zwischen den Komponenten erfolgt.

4. Dokumentation der Architektur

Eine klare Dokumentation ist entscheidend, um Architektur verständlich darzustellen. Sie umfasst unter anderem:

- Komponentendiagramme
  Darstellung der Architektur in den Verschiedenen Sichten wie der Kontextsicht, Bausteinsicht, Verteilungssicht und Laufzeitsicht.
- Beschreibung der Bausteine
  Die einzelnen Schnittstellen und Aufgaben der Makros müssen beschrieben werden.
- Qualitätsziele:
  Die Qualitätsziele der Architektur müssen dokumentiert werden. Beispiele sind kurze Reaktionszeiten oder die Wartbarkeit des Systems.

5. Validierung der Architektur

Eine Architektur wird nicht von Anfang an perfekt sein, daher wird sie getestet, um Schwächen der Architektur frühzeitig zu erkennen.

- Prototyp Erstellung
  Es werden erste Prototypen der Komponenten erstellt und getestet.
- Testen der Anforderungen
  Die Architektur wird daraufhin überprüft, ob die definierten funktionalen und nicht funktionalen Anforderungen erfüllt sind.

6 Umsetzung und Weiterentwicklung der Architektur

Bei der Entwicklung einer Softwarearchitektur handelt es sich um einen iterativen Prozess, der auch während der Implementierung weiter fortgesetzt wird.

- Anpassung und Änderungen
  Wenn es Änderungen der Anforderungen oder Rahmenbedingungen gibt, muss die Softwarearchitektur entsprechen angepasst werden.

6 Anforderungen an die Softwarearchitektur der Simulation

Auf Basis der in Kapitel 5 erarbeiteten allgemeinen Struktur einer Softwarearchitektur bilden wir nun die Rahmenbedingungen für die Entwicklung der Softwarearchitektur unseres digitalen Zwillings. Die Softwarearchitektur muss dabei sowohl funktionale als auch nicht funktionale Anforderungen erfüllen, um eine präzise und zuverlässige Simulation zu gewährleisten.

Funktionale Anforderungen

Die Softwarearchitektur muss alle Grundfunktionen der Hub-Drehbrücke simulieren, dazu gehören Funktionen wie das Anheben und Absenken sowie das Auf- und Zudrehen des mittleren Brückenteils. Dabei müssen die Zustände und Signale der I/O-Peripherie, sowie die Zustände von Endlagensensoren und Wegmessern in SIMIT nachgebildet und an die SPS-Logik weitergeleitet werden.

Nichtfunktionale Anforderungen

Eine wichtige Anforderung an die Architektur ist die Flexibilität und Erweiterbarkeit der Simulation. Dies ist besonders wichtig, da sich die Brücke zum Zeitpunkt der Simulationserstellung noch in der Planungsphase befindet und einige Anpassungen oder Ergänzungen von neuen Komponenten und Anlagenteilen nicht auszuschließen sind. Es ist daher wichtig, eine modulare Struktur der Softwarearchitektur zu gewährleisten, da diese die Wartung und zukünftige Erweiterungen der Simulation erleichtert.

6.1 Überblick der Architekturschichten

Wie auch in der zuvor erarbeiteten allgemeinen Softwarearchitektur wird auch die Architektur des digitalen Zwillings aus verschiedenen Sichten betrachtet. Dabei wird sich für die Entwicklung der Softwarearchitektur des digitalen Zwillings auf die folgenden Sichten beschränkt:

- Kontextabgrenzung
  Hier werden die Schnittstellen und der Datenfluss zwischen der Simulation und der Steuerungsumgebung beschrieben
- Bausteinsicht
  In dieser Sicht wird beschrieben, wie die modulare Struktur des digitalen Zwillings in der Simulationssoftware SIMIT realisiert wird.

Auf die Verteilungssicht wird hier verzichtet, da die Simulation mit der Steuerung zusammen auf einem Computer läuft, ohne dass ein physisches Netzwerk benötigt wird. Durch die Verwendung der Simulationssoftware SIMIT ist bereits eine gewisse Struktur vorgegeben, die bei der Entwicklung der verschiedenen Sichten beachtet werden muss.

6.2 Kontextabgrenzung

Die Kontextabgrenzung des digitalen Zwillings beschreibt die Kommunikationskanäle zwischen der Simulation und den realen sowie virtuellen Steuerungselementen.

Virtuelle SPS und HMI

Der Brückenzwilling und die Steuerungssoftware werden parallel auf demselben Rechner ausgeführt. Für die SPSen und die HMI wird während der Simulation des digitalen Zwillings keine reale Hardware verwendet. Die SPSen, auf denen die Steuerungssoftware der Brücke läuft, werden mithilfe der Software PLCSIM-Advanced emuliert. Auch die Bedienoberfläche der Brücke (HMI) wird virtuell abgebildet. Dies geschieht direkt über das TIA-Portal. Die virtuelle HMI kommuniziert direkt mit den emulierten SPSen und tauscht dabei Zustandsmeldungen und Steuerbefehle aus. Zusätzlich sind die SPSen noch mit der Simulationsumgebung verbunden und tauscht mit dieser ebenfalls Steuerungsbefehle und Zustandsmeldungen aus. Eine direkte Verbindung zwischen der HMI und der

Simulationsumgebung besteht nicht. Für den Austausch von Daten wird hier immer eine SPS als Mittelsmann verwendet.

Kommunikationsschnittstelle

Die Kommunikation zwischen der Simulationsumgebung, der HMI und der emulierten SPSen erfolgt über eine TCP/IP-Verbindung auf der internen Netzwerkkarte des PCs. Diese Schnittstelle stellt sicher, dass die Übertragung der Steuerungsbefehle und Zustandsmeldungen zwischen den drei Teilnehmern gewährleistet wird. In der Abbildung 6.1 wird der Aufbau noch einmal grafisch veranschaulicht erkennen.

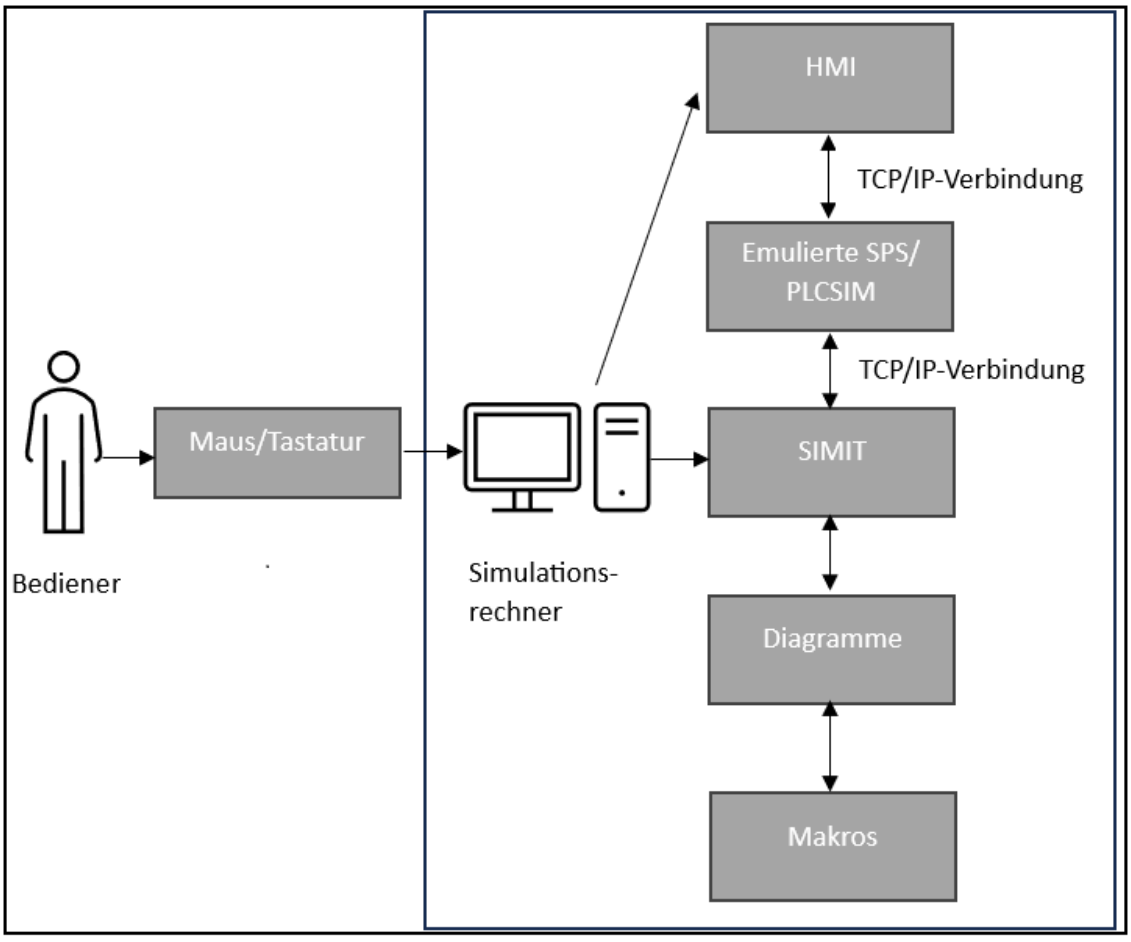

Abbildung 6.1: Kontextabgrenzung des digitalen Zwillings

Datenfluss

In der Simulation der Brückenanlage wird eine Vielzahl an Daten verarbeitet. Der Datenfluss der Softwareumgebung besteht dabei hauptsächlich aus Zustandsmeldungen und Steuerungsbefehlen zwischen den beiden SPSen und den simulierten Komponenten. Am Beispiel des Drehmoduls der Brückensimulation würde der Austausch wie folgend aussehen. Der Bediener fordert über eine Taste am HMI an, dass die Brücke auf- oder zugedreht werden soll. Dieser Befehl wird über TCP/IP an die entsprechende emulierte SPS weitergeleitet. Das Steuerungsprogramm der SPS verarbeitet diesen Befehl und erzeugt ein Ansteuerungssignal, welches die zuständigen simulierten Komponenten des digitalen Zwillings ansteuert. Die Simulation verarbeitet diesen Befehl und teilt der SPS über eine Rückmeldung wie „wird geöffnet“ oder „Endlage erreicht“ den aktuellen Zustand vom Drehmodul mit.

6.3 Modulare Strukturierung und Bausteinsicht

Die Bausteinsicht des digitalen Zwillings beschreibt, wie die modulare Strukturierung des Systems im Rahmen der Simulationssoftware SIMIT ermöglicht werden kann.

Makrobasierte Funktionsbausteine

Die Simulationssoftware SIMIT erlaubt die Erstellung sogenannter „Makros“, welche als eine Art virtueller Funktionsbausteine für die einzelnen Komponenten der Anlage genutzt werden können. Jedes dieser Makros repräsentiert dabei eine spezifische Funktion oder Komponente der Anlage, wie in etwa die Schranke, der Drehantrieb oder der Hydraulikzylinder. Die Makros bilden somit unsere wichtigsten Grundlagen, um eine modulare Architektur der Simulation zu ermöglichen.

Aufbau der Makros

Jedes Makro setzt sich aus drei funktionalen Teilen zusammen, einem Eingabe-, Verarbeitung- und Ausgabeteil, die gemeinsam die Funktion der jeweiligen Komponente oder des Anlagenteils abbilden.

Im Eingabeteil des Makros werden die Eingänge des Funktionsbausteins angelegt, an dem später die Steuerbefehle der SPS, von Tastern oder die Zustände anderer Komponenten wie z.B. Sicherungen und Hauptschalter angeschlossen werden.

Der Verarbeitungsteil dient zum Erstellen der eigentlichen Logik des Makros. Hier wird das logische Verhalten der Komponente bestimmt, die das Makro später abbilden soll. Einfache Makros, wie zum Beispiel Sicherungen und Hauptschalter, erfordern dabei meist nur eine minimalistische Logik. Bei Makros von komplexeren Komponenten können hingegen auch umfangreichere Funktionen wie Zeitglieder, Rampen und Konvertierungsbausteine zum Einsatz kommen.

Im Ausgabeteil des Makros werden die Ausgänge des Funktionsbausteins erstellt, an den später die aktuellen Zustände der Komponente anliegen.

Makros verfügen in SIMIT über zwei verschiedene Ansichten. Die äußere Ansicht des Makros, wie in Abbildung 6.2 dargestellt, dient der Repräsentation der Komponente im Simulationsfenster. An ihr werden die Kopplungen der Ein- und Ausgangssignale angeschlossen, über die die SPS mit den Komponenten kommuniziert.

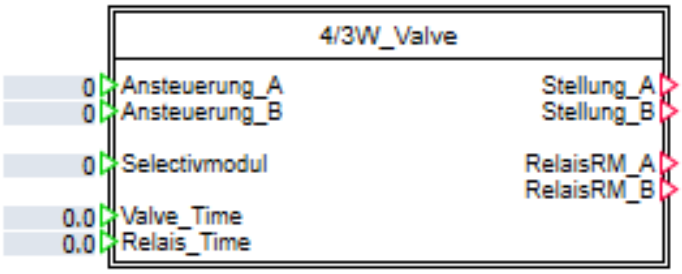

Abbildung 6.2: Außenansicht eines Makros

Die in Abbildung 6.3 dargestellte innere Ansicht des Makros dient der Bearbeitung und Beobachtung, in ihr wird das Verhalten sowie der Ein- und Ausgabeteil der Komponen-

ten definiert. Im linken Teil der Ansicht werden die Eingänge und im rechten Teil die Ausgänge angelegt. Der mittlere Teil der Ansicht ist der Verarbeitungsteil.

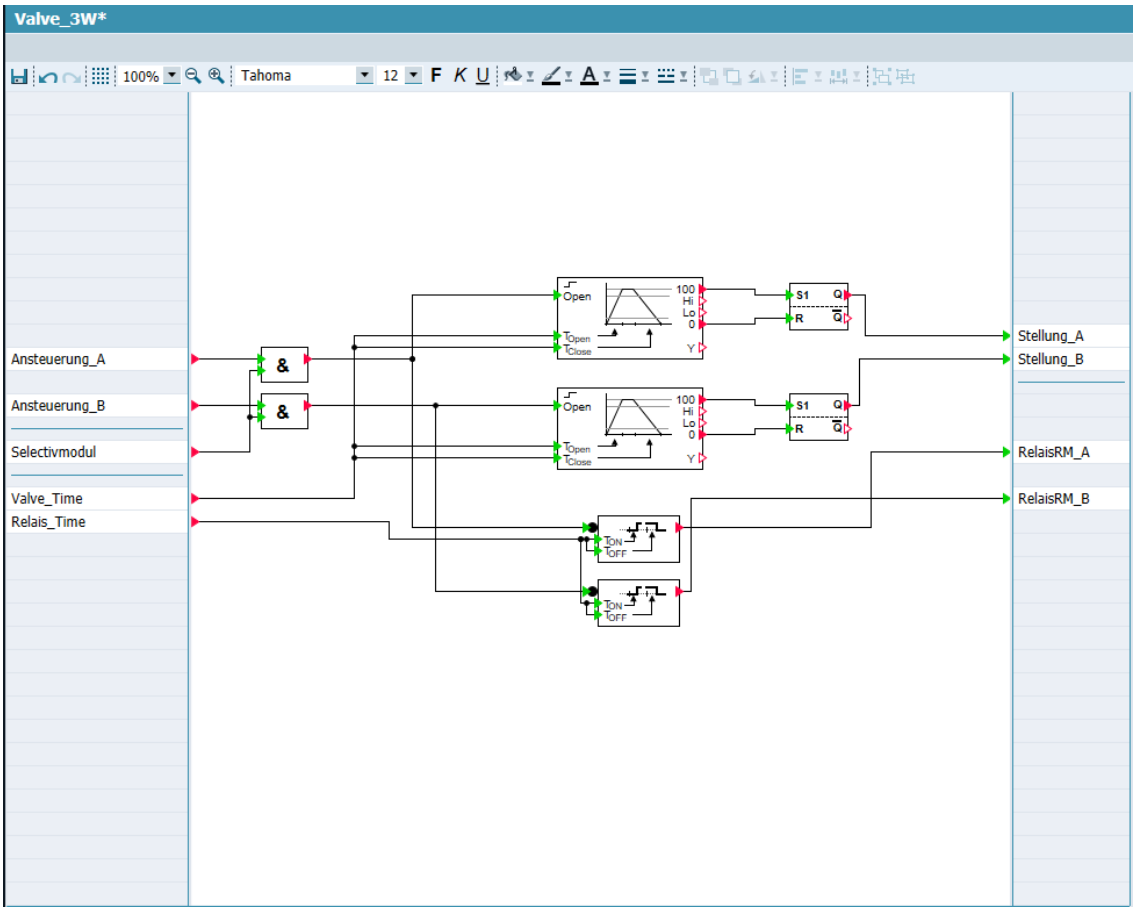

Abbildung 6.3: Innere Ansicht eines Makros

Modularität

Durch den modularen Aufbau der Simulation wird das Testen einzelner Brückenkomponenten wie des Drehmechanismus, der Schranken oder der Hubfunktion der Zylinder ermöglicht. Dies geschieht in SIMIT durch Aufteilung der Anlagenteile in unterschiedliche „Diagramme“. Jedes Diagramm stellt ein eigenes Simulationsfenster dar, in dem die erstellten Makros aufgerufen werden können. Die Makros können hier logisch miteinander verknüpft werden oder über sogenannte Peripherie-Konnektoren mit dem Steuerungsprogramm der SPS verbunden werden. In Abbildung 6.4 ist beispielhaft zu sehen, wie die

Zerlegung der Anlage in einzelne Module aussehen könnte. Hier wird für die einzelnen Anlagenteile wie z.B. der Ampel, Schranke und Beleuchtung ein eigenes Diagramm erzeugt, wodurch diese unabhängig voneinander getestet werden können.

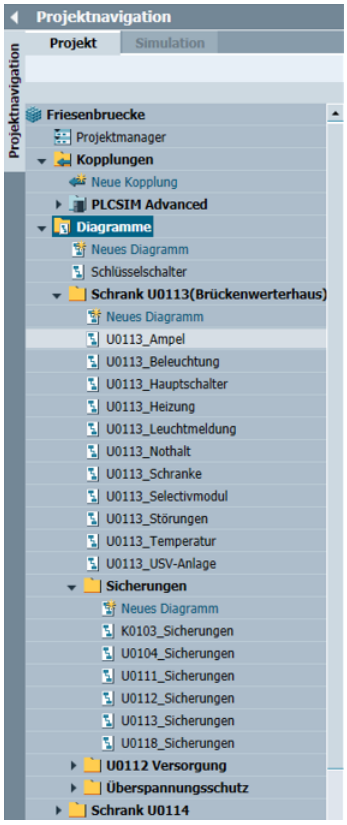

Abbildung 6.4: Aufteilung der unterschiedlichen Modul in Diagramme

Um Komponenten und Anlagenteile über Diagrammgrenzen hinweg zu verbinden, werden sogenannte „globale-Konnektoren“ verwendet. Sie können sowohl als Ausgangs- und Eingangskonrektor verwendet werden. In Abbildung 6.5 kann beobachtet werden, wie der Aufruf und die Beschaltung eines Makros in einem Diagramm aussieht. Die grauen Elemente am Makro sind die Globalen-Konnektoren, die grünen sind die Ausgänge und die roten die Eingänge der SPS.

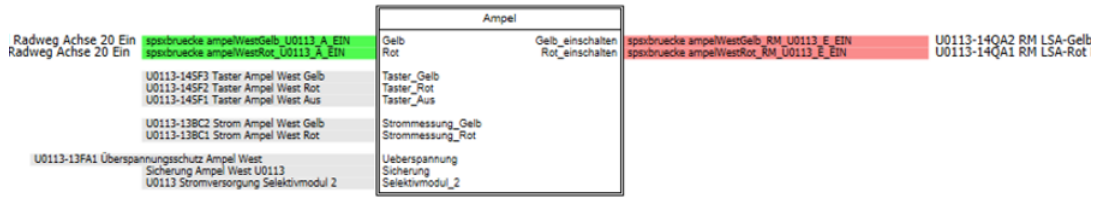

Abbildung 6.5: Makro mit angeschlossenen Verknüpfungen

Bausteinsicht

In der Abbildung 6.6 wird die Bausteinsicht dargestellt, die sich aus der Struktur von SIMIT ergibt. Die Integrationsschicht stellt die Kommunikation zwischen der Simulation und der SPS dar. Es ist zu erkennen, dass die Steuersignale der SPS mit einer TCP/IP-Verbindung über die Netzwerkkarte des Computers an die simulierten I/O-Baugruppen von SIMIT übertragen werden. Diesen wird in SIMIT eine Variable zugewiesen, die in der Präsentationsschicht der Simulation verwendet wird. Die Präsentationsschicht setzt sich aus den einzelnen Diagrammen zusammen, die für die Simulation erstellt wurden.

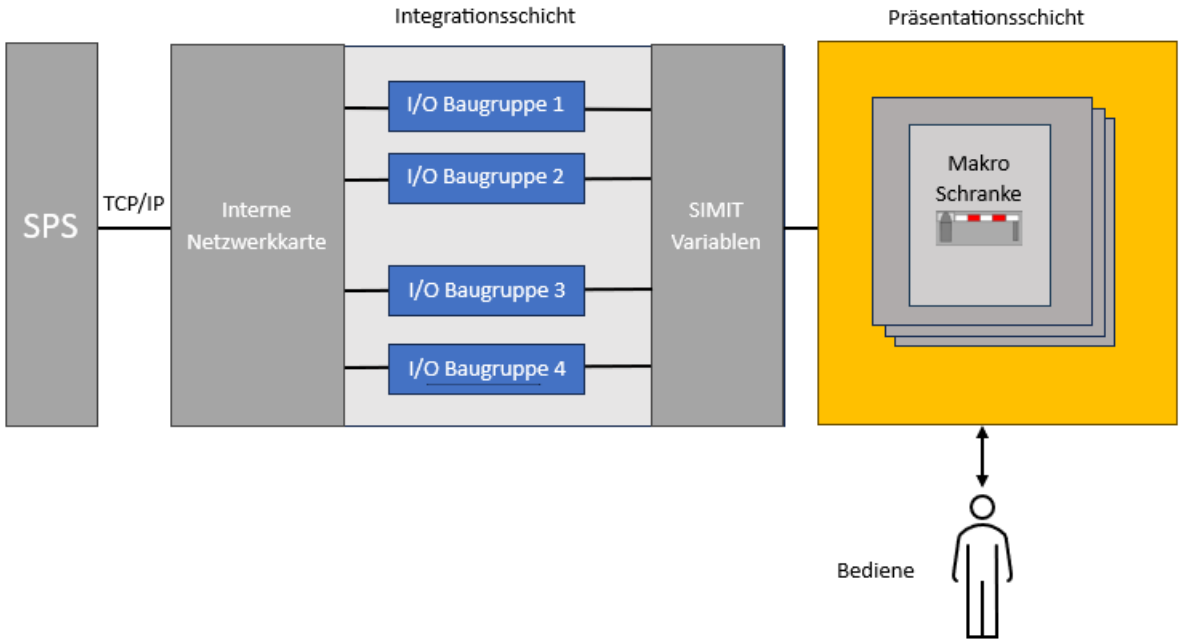

Abbildung 6.6: Bausteinsicht

Die Abbildung 6.7 zeigt den Aufbau der Präsentationsschicht durch Aufteilung in unterschiedliche Ebenen. Die erste Ebene sind die Diagramme, die aus unterschiedlichen Makros bestehen, die miteinander verbunden werden. In der zweiten Ebene wird noch einmal der Aufbau und die Verbindungen der Makros verdeutlicht.

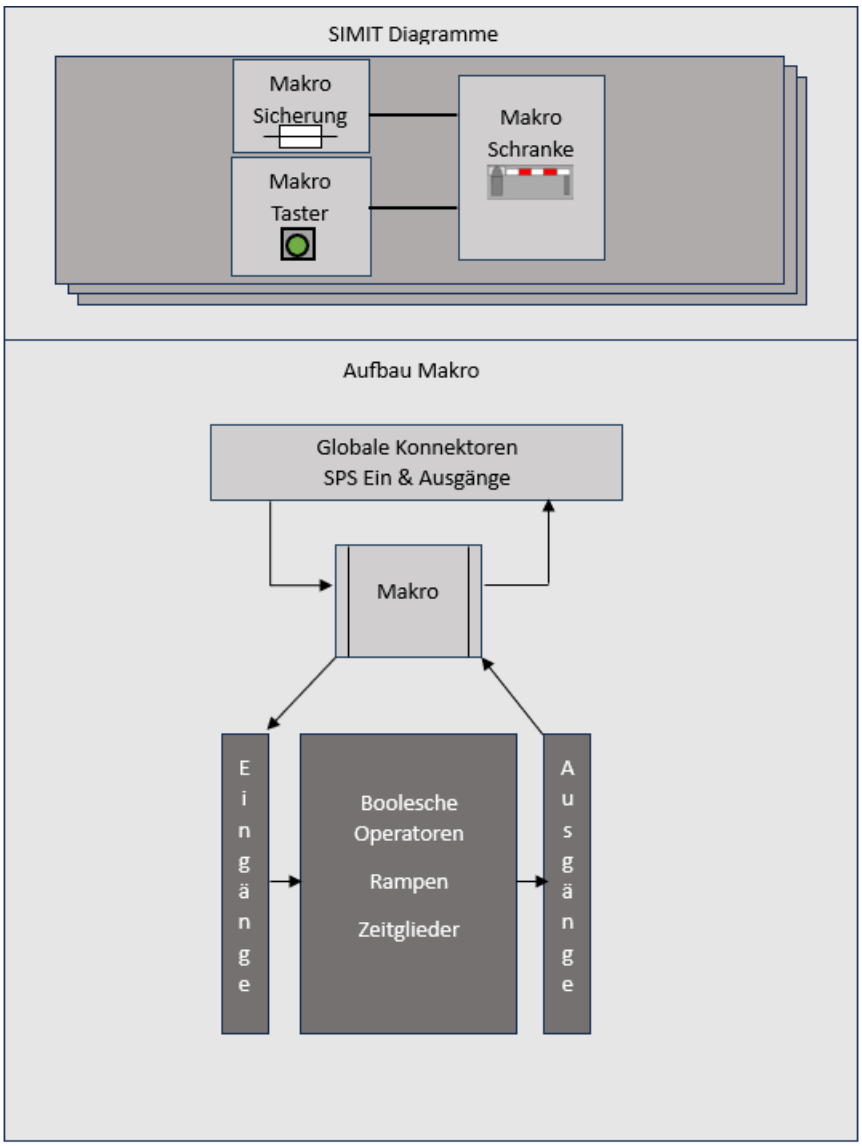

Abbildung 6.7: Aufbau der Präsentationsschicht in SIMIT

6.4 Simulation von Fehlerfällen

Der digitale Zwilling muss in der Lage sein, Fehlerzustände und Sonderfälle zu simulieren, um das Verhalten der Steuerung in sicherheitskritischen Momenten testen zu können. Diese Situationen lassen sich in SIMIT ebenfalls über die Makros lösen. An den Ein- und Ausgängen der Makros können über die äußere Ansicht Zustände erzwungen werden, um Fehlerzustände wie „Wegmessung fehlerhaft“oder „Endlage defekt“zu simulieren. Mit dieser Funktion ist es also möglich, die Steuerungssoftware auch in unvorhersehbaren Situationen zu testen.

6.5 Physikalische Eigenschaften

Da die Simulation der Komponenten und Anlagenteile rein virtuell und ohne Anbindung an die reelle Peripherie erfolgt, müssen die physikalischen Eigenschaften bei der Implementierung der Makros bedacht werden. Da die Komponenten digital sind, werden Steuerungsbefehle nahezu sofort umgesetzt und verarbeitet. Für Bauteile wie Sicherungen, Schaltern und Taster stellt dies kein Problem dar, da diese genauso sprunghaft wie in der Simulation reagieren. Für Komponenten wie den Drehantrieb und die Schranken sieht das jedoch anders aus. Hier werden für die Umsetzung des Steuerungsbefehls mehrere Sekunden bis Minuten benötigt. Diese Reaktions- und Umsetzungszeit muss bei der Entwicklung der Makros natürlich bedacht werden. Es ist also wichtig, dass für die verschiedenen Makros die entsprechenden Anlaufkennlinien und Verzögerungszeit implementiert werden. Hierzu bietet SIMIT bereits einige Möglichkeiten. Für Anlaufkennlinien können sogenannte „Rampenbausteine“verwendet werden und für Verzögerungen gibt es „delay Bausteine“Bausteine. Weiterhin gibt es schon vorgefertigte Bausteine, die das Verhalten von Ventilen, Pumpen und Motoren simulieren.

7 Implementierung des digitalen Zwilling

Dieses Kapitel beschäftigt sich mit der Umsetzung des digitalen Zwillings anhand der zuvor erarbeiteten Softwarearchitektur. Anhand der Umsetzung einiger Brückenkomponenten wie des Drehmechanismus, des Hubantriebs und der Ampel soll dabei veranschaulicht werden, wie die Softwarearchitektur der Simulation in SIMIT realisiert wurde. Dabei wird auf die Struktur der einzelnen Makros eingegangen und die einzelnen Funktion der Module erklärt.

7.1 Benutzeroberfläche

Bevor jedoch auf die einzelnen Komponenten und Makros der Simulation eingegangen wird, wird noch einmal kurz die Benutzeroberfläche der Steuerungssoftware vorgestellt. Die HMI stellt dabei die wichtigsten Interaktionsschnittstellen für den Bediener der Anlage dar. Sie ermöglicht sowohl die Bedienung als auch die Überwachung der Anlage. In ihr werden alle relevanten Zustände der Anlage dargestellt sowie deren Steuerung ermöglicht.

- Visuelle Steuerungselemente:
  In der HMI sind grafische Bedienelemente implementiert, die es dem Bedienpersonal ermöglichen, die Brücke zu steuern.
- Buttons und Schalter:
  Für die Funktionen wie „Brücke öffnen“ oder „Schranken schließen“ sind eigene Schaltflächen implementiert.
- Zustandsanzeigen:
  Statusanzeigen von Parametern wie Brückenhöhe, Volumenstrom oder Kraftmessungen sind in den einzelnen Bildern der HMI realisiert.

- Fehlermeldungen und Betriebsmeldungen
  In der Benutzeroberfläche werden aktuelle Fehler- und Betriebsmeldungen der Anlage angezeigt. Fehlermeldungen bleiben dabei so lange bestehen, bis die Fehlerquelle behoben ist und die Meldung anschließend quittiert wurde.

Abbildung 7.1: Bedienelemente in der HMI

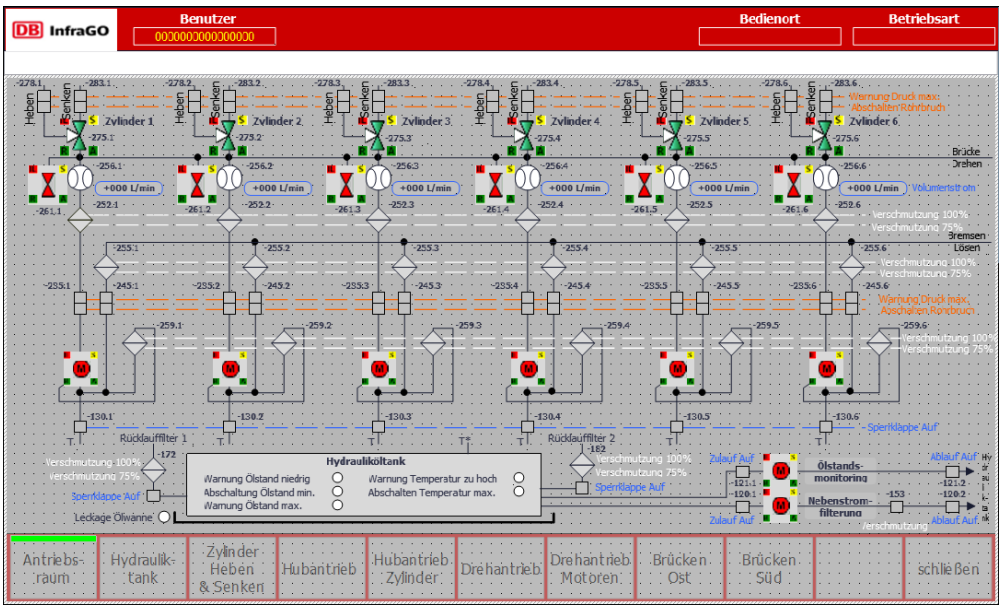

Abbildung 7.2: Darstellung der Komponentenzustände in der HMI

Neben der Benutzeroberfläche der HMI gibt es in den einzelnen Diagrammen der Simulation auch noch weitere Bedienelemente. Dabei handelt es sich meistens um Schutzorgane wie Sicherungen oder Hauptschalter. Sie dienen dazu, ein Auslösen der jeweiligen Komponente zu simulieren. Manche Anlagenteile wie z.B. der Schrankenanlage oder der Ampeln besitzen zusätzlich noch Schalter und Taster. Sie dienen dazu, ein manuelles Schalten der Komponente über einen reellen Schalter und nicht über die HMI zu simulieren.

7.2 Drehmechanismus

Der Drehmechanismus der Brücke erlaubt es, das mittlere Brückensegment zu drehen, nachdem dieses entriegelt und angehoben wurde. Er stellt für die Steuerungssoftware eine komplexe Herausforderung dar, da diese das Brückensegment exakt auf die richtigen Positionen bewegen muss, in denen die Brücke abgesetzt werden darf.

Makrostruktur des Drehmechanismus

In SIMIT wurde der Drehmechanismus in einen eigenen Diagramm dargestellt, in den das Makro für die Drehbewegung aufgerufen wird. Neben den Makro gibt es hier noch zwei Leuchtmelder, die signalisieren, wenn die Brücke geschlossen oder geöffnet ist.

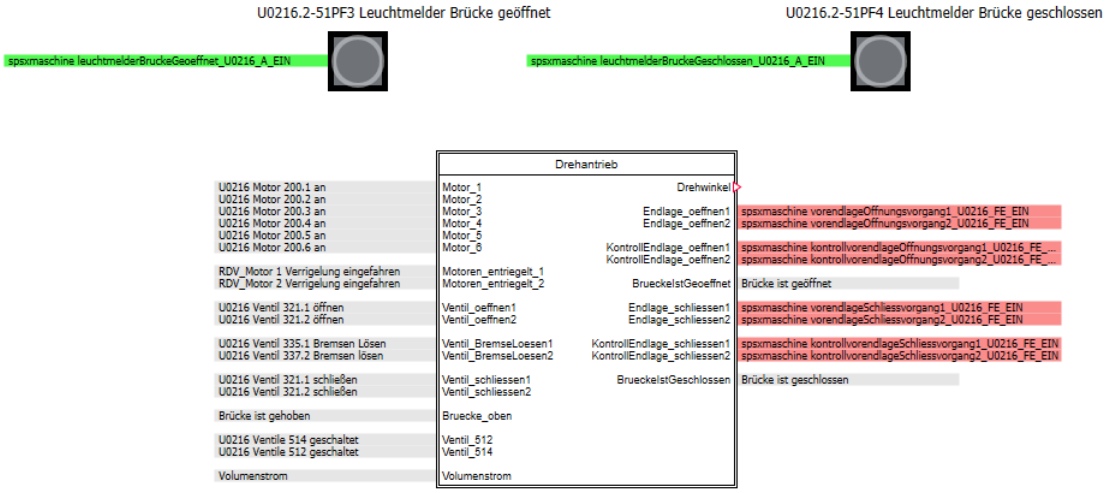

Abbildung 7.3: Diagramm des Drehantriebs

Der Aufbau des Makros wird in Abbildung 7.4 dargestellt und sieht wie folgend aus:

- Eingabeparameter:
  Die Eingabeparameter bestehen aus den Zuständen der Anlage, die notwendig sind, um die Drehbewegung des Drehpfeilers zu ermöglichen. Hierzu gehören die Motorenpumpen, die Verriegelung des Drehantriebs, die Ventile zur Drehrichtungsbestimmung sowie die Bremsen des Drehantriebs und der Volumenstrom.

- Verarbeitungsteil:
  Im Verarbeitungsteil werden zunächst die Zustände der Eingänge abgefragt. Wenn alle Zustände, die für eine Drehrichtung benötigt werden, gegeben sind, wird die Drehbewegung über einen Rampe gestartet. Der Ausgang des Rampenbausteins repräsentiert den Drehwinkel der Brücke, der unsere Endlagenschalter beim Erreichen der entsprechenden Position auslöst. Damit eine Drehbewegung gestartet wird, mussen sich die Brücke im Zustand „Brücke oben “befinden, die Bremsen und die Verriegelung gelöst und eine der Motorgruppen eingeschaltet sein. Die Geschwindigkeit der Drehbewegung der Brücke ist abhängig vom Volumenstrom der Pumpen.

- Ausgabeparameter:
  Im Ausgabeteil sind die aktuelle Position sowie die unterschiedlichen Endlagen, die während der Drehbewegung ausgelöst werden, angelegt.

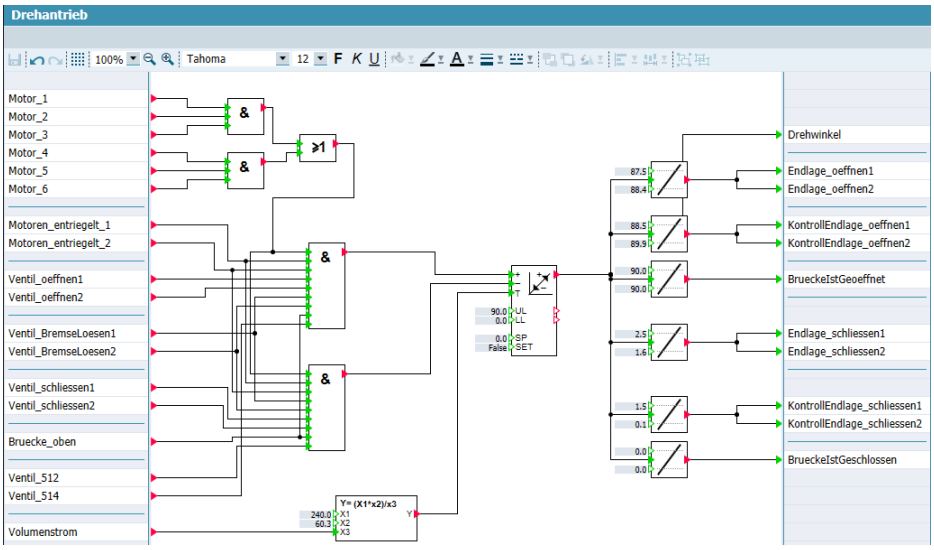

Abbildung 7.4: Makro des Drehantriebs

Rückmeldung des Drehmechanismus

Der Drehmechanismus sendet kontinuierlich Rückmeldungen über die aktuelle Position der Brückenbewegung. Diese Rückmeldung kann im Programm dazu genutzt werden, um die Geschwindigkeit der Brückenbewegung über den Volumenstrom anzupassen, damit die Endlagenposition der Brücke präzise erreicht werden kann.

7.3 Hubmechanismus

Durch den Hubmechanismus der Brücke wird das Anheben des mittleren Brückensegments ermöglicht. Für die Implementierung des Mechanismus ist es erforderlich, alle sechs Hydraulikzylinder, die für das Anheben des Brückenteils verantwortlich sind, nachzubilden. Das erstellte Diagramm des Hubmechanismus wird in Abbildung refHubantrieb dargestellt. Das Modul besteht aus sechs Zylinder Makros, einem Makro für die Geschwindigkeitsberechnung der Hubbewegung und einem Makro zur Auswertung der Brückenposition. In der Abbildung refHubantrieb wurde jedoch nur ein Zylinder mit abgebildet, da sie ansonsten zu groß wird.

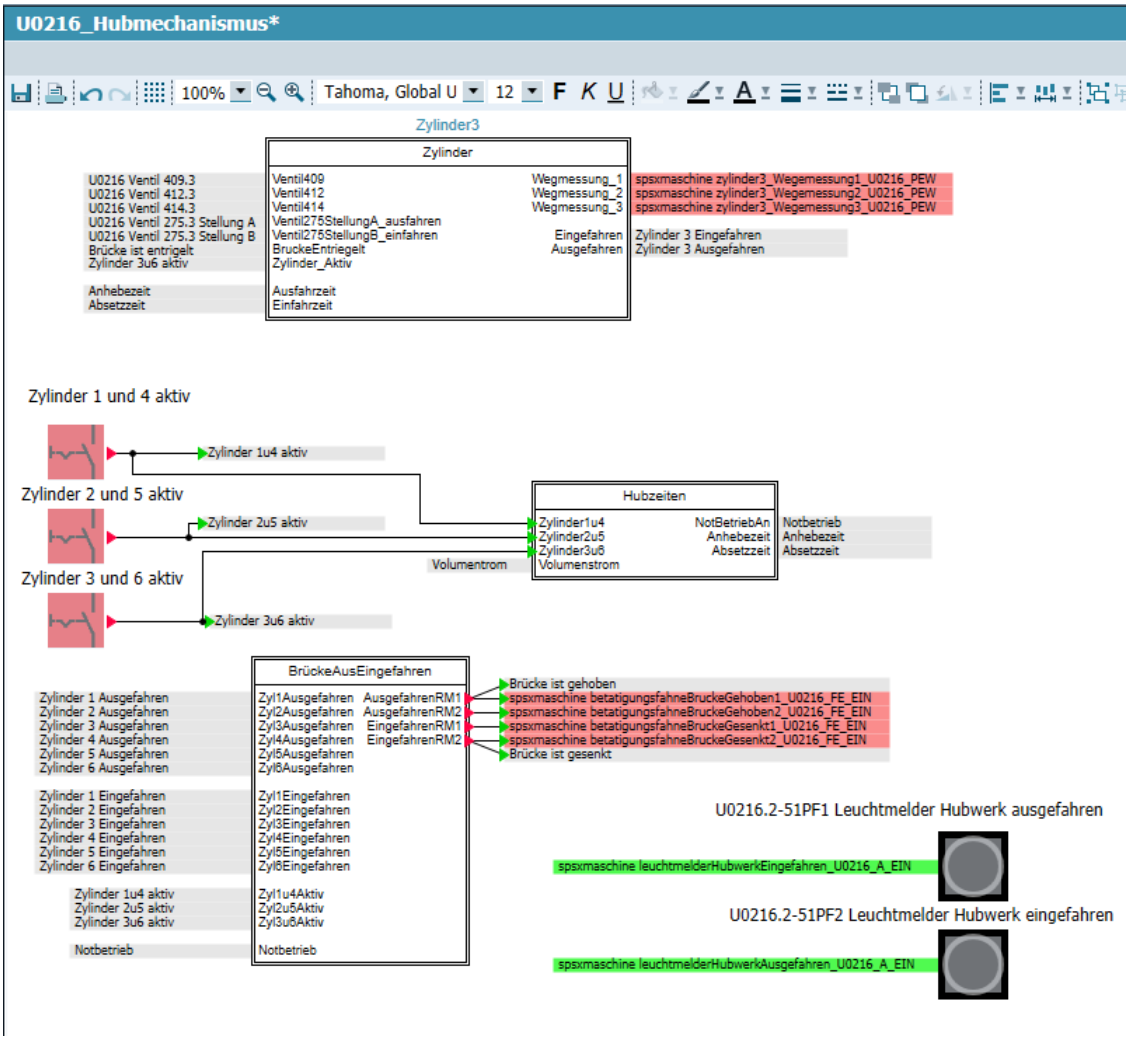

Abbildung 7.5: Diagramm des Hubmechanismus

Makrostruktur des Hubmechanismus

Die Zylindermakros aller sechs Hubzylinder sind identisch aufgebaut und sind wie folgend strukturiert:

- Eingabeparameter:
  An den Eingängen des Makros befinden sich alle Zustandsmeldungen, die für den Betrieb des Zylinders notwendig sind. Diese umfassen Rückmeldungen der Ventilzustände, die den Betrieb des Zylinders steuern, Signale für die Freischaltung des Zylinders sowie die Fahrtzeiten der Hubbewegung.

- Verarbeitungsteil:
  Im Verarbeitungsteil wird zunächst abgefragt, ob der Zylinder durch eins der Ventile drucklos geschaltet ist. Weiterhin wird überprüft, ob die Brücke entriegelt und der Zylinder aktiviert ist. Wenn diese Bedingungen erfüllt sind, wird über das Ventil 275 die Richtung der Hubbewegung ausgewählt. Die Geschwindigkeit, mit der sich der Zylinder bewegt, wird in einem separaten Makro berechnet und über eine globale Variable an den Zylinder angebunden. Die Hubbewegung des Zylinders wird über einen Rampenbaustein simuliert, dessen Ausgangswert die Ausfahrtdistanz des Zylinders darstellt.

- Ausgabeparameter:
  Am Ausgang des Makros wird die Fahrdistanz für die Distanzmessung angelegt. Weiterhin gibt es eine Endlage für den eingefahrenen und ausgefahrenen Zustand des Zylinders

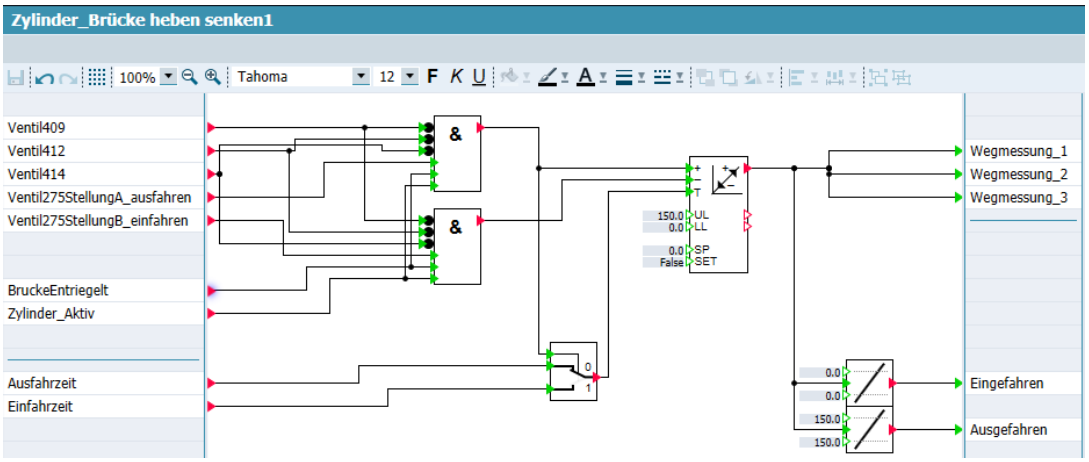

Abbildung 7.6: Makro der Zylinder

Makro Hubgeschwindigkeit

Das Makro der Hubzeiten dient für die Berechnung der Hubgeschwindigkeit sowie die Auswahl der aktiven Zylinder. Die Zylinder können dabei nur in Paaren aktiviert und deaktiviert werden, um ein gleichmäßiges Kraftverhältnis beim Anheben der Brücke zu gewährleisten. Ein Betrieb des Hubwerks ist dabei mit einer Zylinderzahl von sechs oder vier Zylindern möglich. Der Aufbau des Makros sieht dabei wie folgend aus:

- Eingabeparameter:
  Im Eingangsbereich sind die Taster für die Auswahl der aktiven Zylinder sowie der Volumenstrom des Hubantriebs eingestellt angelegt.

- Verarbeitungsteil:
  Zunächst wird die Anzahl der aktiven Zylinderpaare durch Auswertung der Schalter überprüft. Wenn alle sechs Zylinder aktiviert sind, wird der Hubantrieb im Normalbetrieb betrieben. Durch die Deaktivierung einer Zylindergruppe wird die Anlage nur noch mit vier Zylindern betrieben. Durch die Reduzierung der Zylinderzahl befindet sich die Anlage im Notbetrieb, und die benötigte Fahrdauer der Hubbewegung erhöht sich. Neben der Anzahl der Zylinder ist die Fahrgeschwindigkeit der Brücke auch vom vorhandenen Volumenstrom abhängig.

- Ausgabeparameter:
  Die Ausgangsparameter setzten sich aus den Fahrgeschwindigkeiten fürs Ein- und Ausfahren sowie einer Meldung für den Fahrbetrieb zusammen.

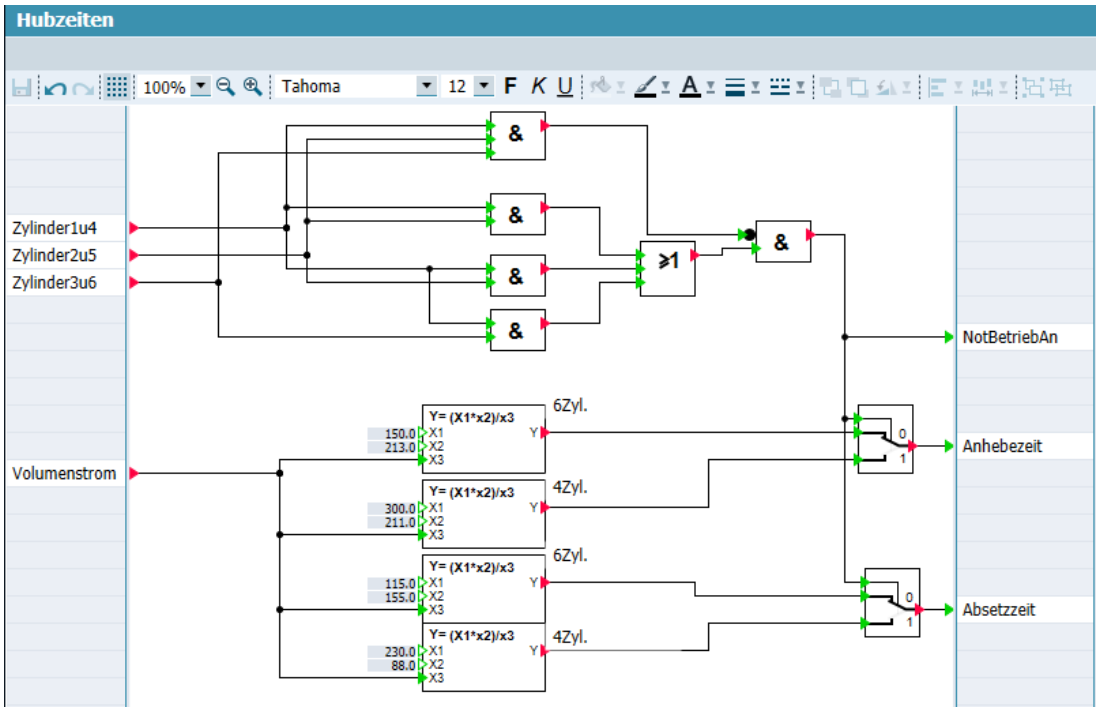

Abbildung 7.7: Makro zur Hubgeschwindigkeitsermittlung

Zusammenfassung Hubmechanismus

Das Modul des Hubmechanismus stellt ein großes und komplexes Modul dar. Die Funktion des Hubmechanismus entsteht dabei aus der Zusammenschaltung von acht einzelnen Makros, die in Kombination die Funktion abbilden können. Das Modul liefert der Steuerung dabei konstant Rückmeldungen über die genaue Position der einzelnen Zylinder. Weiterhin wird eine Rückmeldung beim Erreichen der Endlagen der Brückenbewegung erzeugt. Das Makro zur Auswertung der Brückenposition wird dabei nicht näher erklärt, hier gibt es nur eine einfache Boolesche Abfrage, ob alle aktiven Zylinder aus- oder eingefahren sind.

7.4 Implementierung einer Ampelanlage

Für die Ampelanlage gibt es in der Simulation auch ein eigenes Diagramm. Neben dem eigentlichen Makro der Ampelanlage gibt es hier noch einige andere Komponenten, die für die Steuerung der Ampel wichtig sind. So wird in diesem Diagramm neben dem Makro auch noch die Sicherung, der Überspannungsschutz, die Stromüberwachung, sowie Statusleuchtmelder und die Bedientaster der Ampel dargestellt.

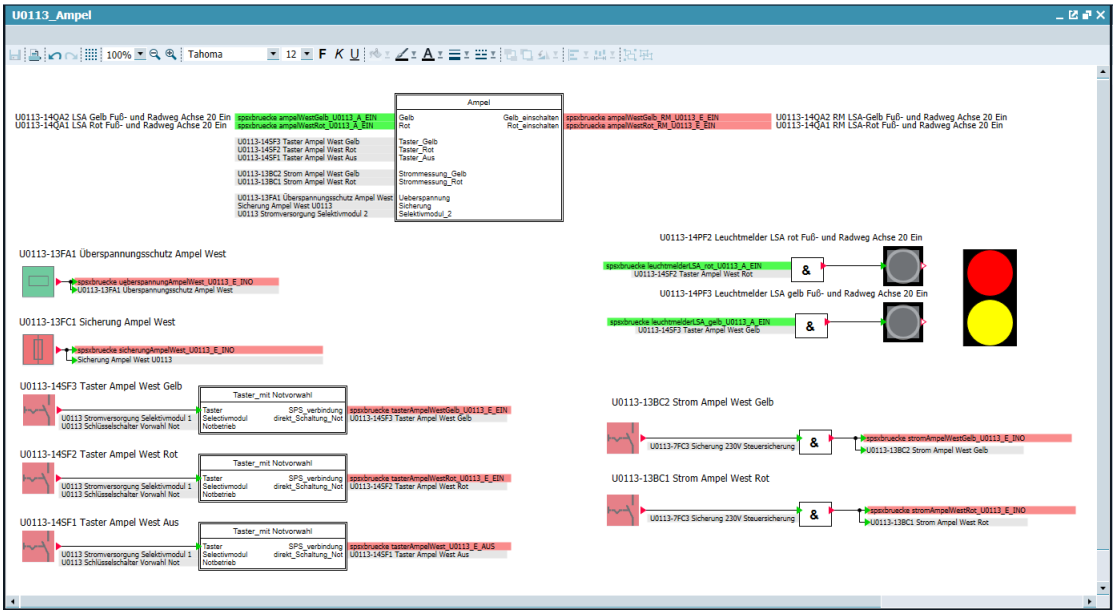

Abbildung 7.8: Diagramm der Ampelanlage

Das Makro der Ampelsteuerung setzt sich dabei wie folgend zusammen:

- Eingabeparametern:
  Die Eingabeparameter des Makros setzen sich aus den Schutzorganen wie der Sicherung und dem Überspannungsschutz, sowie der Spannungsversorgung durch das Selektivitätsmodul und der Strommessung der beiden Lampen zusammen. Weiterhin werden die Steuersignale der SPS über die Anschlüsse "Gelbünd "Rotängebunden. Zusätzlich gibt es noch Taster, die die manuelle Bedienung der Ampel ermöglichen.

- Verarbeitungsteil:
  Im Verarbeitungsteil wird zunächst überprüft, ob es ein Ansteuerungssignal von der SPS oder dem Taster gibt. Ist dies der Fall, wird die Ampel nach einer kurzen Verzögerung geschaltet, falls die korrekte Funktion der Schutzorgane und der Spannungsversorgung gegeben ist. Eine Ansteuerung des Ampelmakros über die Taster ist dabei nur möglich, wenn diese zuvor aktiviert wurden. Die Taster sind nur aktiv, wenn sich die Brücke im Wartungs- oder im Notbetrieb befindet. Die Einstellung der Brückenbetriebsarten erfolgt in einem separaten Diagramm, da dies auch für andere Anlagenteile relevant ist. Im Wartungsbetrieb erfolgt die Bedienung der Ampel mit den Tastern über die SPS. Wenn sich die Anlage im Notbetrieb befindet, kann die Ampel über die Taster direkt gesteuert werden, damit eine Bedienung der Ampel auch beim Ausfall der SPS möglich ist.

- Ausgabeparameter:
  Die Ausgabeparameter der Ampelanlage bestehen lediglich aus zwei Rückmeldungen, die der SPS signalisieren, ob die gelbe oder rote Leuchte der Ampel eingeschaltet ist.

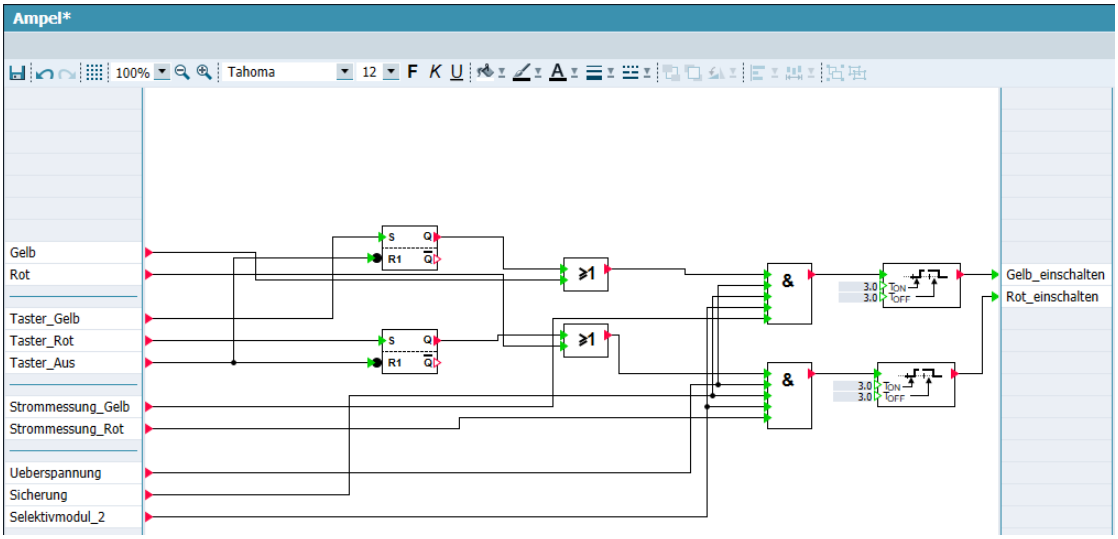

Abbildung 7.9: Makro der Ampelanlage

7.5 Zusammenfassung der Implementierung

Anhand der implementierten Diagramme und Makros wird ersichtlich, wie die erarbeitete Softwarearchitektur des digitalen Zwillings in SIMIT umgesetzt werden konnte. Die einzelnen Komponenten sind jeweils als eigene Makros realisiert worden. In den einzelnen Diagrammen wurden die Funktionen der einzelnen Anlagenteile abgebildet. Dies geschieht durch den Aufruf und die Verknüpfung der Makros untereinander oder mit den Steuersignalen der SPS. Die einzelnen Makros besitzen alle eine definierte Eingabe, Ausgabe und Verarbeitungslogik, mit der die Funktion der Komponente nachgebildet wird. Durch diese Struktur wird eine hohe Modularität, Flexibilität und Testbarkeit des digitalen Zwillings ermöglicht.

8 Virtuelle Inbetriebnahme am Beispiel der Ampelanlage

Abschließend wird noch einmal exemplarisch dargestellt, wie die virtuelle Inbetriebnahme des digitalen Zwillings ablaufen kann. Hierfür wird die Ampelanlage als konkretes Beispiel verwendet, um die grundlegenden Prinzipien und Vorgehensweisen zu erläutern.

Das Kapitel soll dabei als Vorlage für das Vorgehen dienen, wie die Steuerungssoftware einer beliebigen Anlage getestet und validiert werden könnte. Dies soll verdeutlichen, wie der digitale Zwilling für die Testung der Steuerungssoftware genutzt werden kann.

8.1 Vorbereitung und Initialisierung der Testumgebung

Zunächst muss die Simulation für die virtuelle Inbetriebnahme gestartet und mit der Steuerungsumgebung verbunden werden. Hierzu muss die Simulation in SIMIT über den „Startbutton “ betätigt werden. SIMIT öffnet beim Starten der Simulation automatisch eine PLCSIM-Advanced instance, in der alle SPSen des Programms emuliert werden.

Die emulierten SPSen müssen nun noch mit dem aktuellen Programm der Steuerungssoftware beladen und gestartet werden. Abschließend wird noch einmal die Simulation der HMI über das TIA-Portal gestartet.

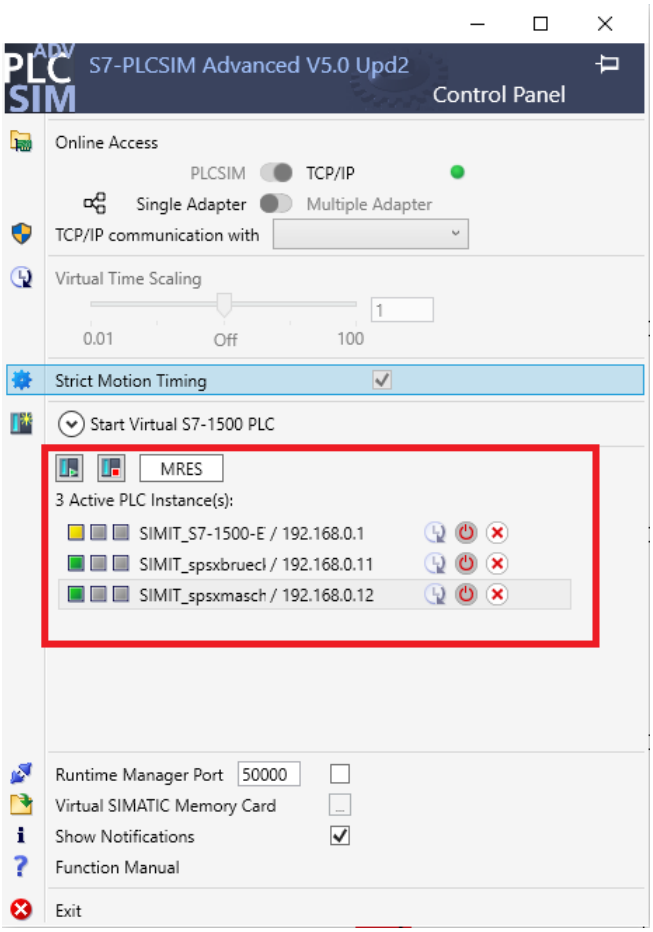

Abbildung 8.1: Darstellung der emulierten SPSen in PLCSIM Advanced

8.2 Test der Grundfunktionen im Normalbetrieb

Nachdem die virtuelle Anlage in Betrieb genommen und mit der Steuerungsumgebung verbunden wurde, kann das Steuerungsprogramm getestet werden. Dazu wird in dieser Phase zunächst überprüft, ob die Grundfunktionen der Ampel im Normalzustand der Anlage ordnungsgemäß ausgeführt werden:

- Automatikbetrieb:
  Wenn sich die Ampelanlage im Automatikbetrieb befindet, wird diese durch eine Schrittkette der Brückensteuerung automatisch geschaltet. Es wird überprüft,

ob die gelbe und rote Ampelleuchte gemäß den definierten Betriebszuständen der Schrittkette geschaltet wird.

- Handbetrieb der HMI:
  Es wird die Steuerung der Ampelanlage über die HMI überprüft. Hierzu wird getestet, ob die Ampelanlage über das HMI geschaltet werden kann, wenn sich die Ampel im Handbetrieb befindet.

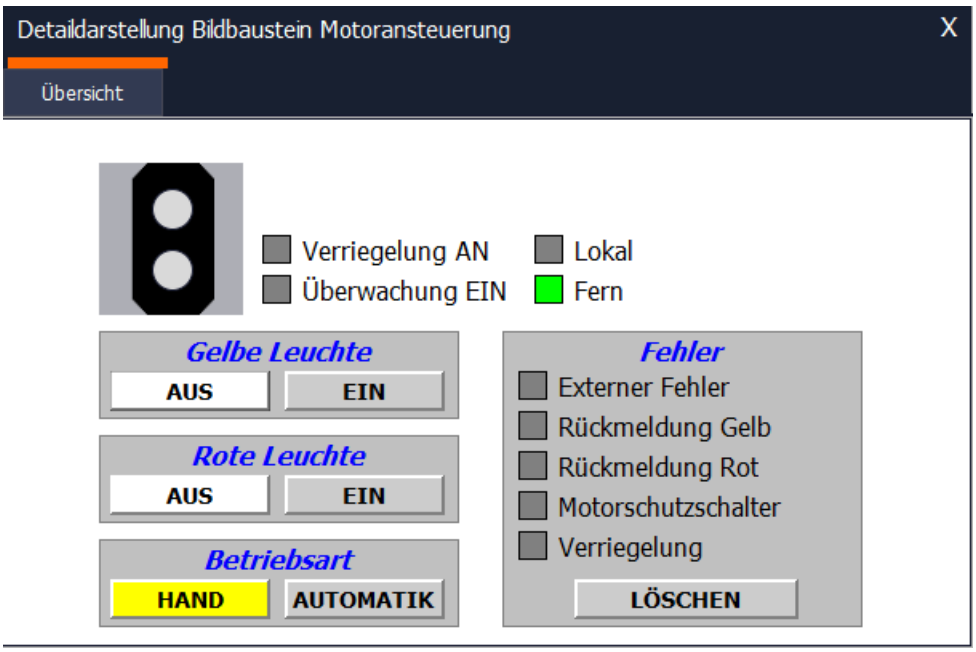

Abbildung 8.2: Bedienoberfläche der Ampelanlage in der HMI

8.3 Test der manuellen Bedienung

Nachdem die korrekte Funktionsweise der Grundfunktion bestätigt wurde, wird die manuelle Steuerung der Ampelanlage überprüft. Dabei wird insbesondere das Verhalten der Ampelsteuerung im Wartungs- oder Notbetrieb getestet.

- Wartungsbetrieb:
  Wenn der Wartungsbetrieb der Anlage aktiv ist und die Bedienhoheit beim Schaltschrank der Ampelsteuerung liegt, soll die Steuerung ausschließlich über das HMI

des Schaltschranks und die lokalen Taster möglich sein. Es muss überprüft werden, ob die Kommunikation der Taster mit der Steuerungssoftware ordnungsgemäß funktioniert.

- Notbetrieb:
  Wenn sich die Ampelanlage im Notbetrieb befindet, soll die Ampel ausschließlich über die Taster zu steuern sein. Die Taster sollen die Ampel dabei direkt steuern, ohne dass die SPS beteiligt ist. Durch diesen Test soll sichergestellt werden, dass die Ampel auch bei einem Ausfall der Steuerung bedient werden kann.

8.4 Störungssimulation

Um die Inbetriebnahme des Steuerungsprogramms der Ampelanlage abzuschließen, sollte diese abschließend noch einmal im Störungsfall untersucht werden. Dies ist besonders wichtig, damit die Robustheit des Programms auch unter kritischen Bedingungen sichergestellt ist. Dazu sollten die folgenden Kriterien erfüllt sein.

- Alle Störungen der Ampelanlage wie SSicherung ausgelöstöder "Überspannungsschutz ausgelöstßollten über eine Fehlermeldung in der HMI angezeigt werden. Die korrekte Funktion der Fehlermeldungen ist dabei zu testen.
- Am Ampelbaustein der HMI sollte erkenntlich sein, dass sie gestört ist.

9 Fazit

Im Rahmen der vorliegenden Bachelorarbeit wurde exemplarisch die Entwicklung eines digitalen Zwillings für die Simulation der Friesenbrücke behandelt. Ziel dieser Arbeit war die virtuelle Nachbildung der Brücke, damit die Steuerungssoftware schon während der Entwicklungsphase der realen Anlage getestet werden kann.

Im theoretischen Teil der Arbeit wurden dazu die Grundlagen des Konzepts eines digitalen Zwillings und einer Softwarearchitektur erläutert, auf deren Basis die Entwicklung des digitalen Zwillings erfolgte.

Der praktische Teil der Arbeit widmete sich der Implementierung des digitalen Zwillings. Hier wurde die erarbeitete Struktur verwendet, um die verschiedenen Brückenfunktionen nachzubilden. Durch den Einsatz von Makros in SIMIT konnte dabei die modulare Struktur der Architektur umgesetzt werden, durch die die flexible Anpassung und Erweiterung der Anlage ermöglicht wird.

Bei der Verwendung von SIMIT für die Entwicklung des digitalen Zwillings, ist jedoch festzuhalten, das die Software in der Basisversion zwar ein mächtiges Werkzeug für die Simulationserstellung ist, es jedoch an vielen Funktionen und Makros fehlt, um eine effiziente und schnelle Programmierung zu ermöglichen. Durch die Erstellung der Makros für den digitalen Zwilling wurde jedoch eine Basis für zukünftige Projekte der Actemium Cegelec Mitte GmbH geschaffen.

Auch wenn sich die Steuerungssoftware zum jetzigen Zeitpunkt noch in der Entwicklung befindet, konnte sich der digitale Zwilling bereits als nützlich erweisen. So konnte mit ihm das Programm für die Ampel- und Schrankenanlage bereits getestet werden.

Literaturverzeichnis

[1] www.profibus.de: Profinet. (Eingesehen am 18.11.2024). – URL https://www.profibus.de/profinet-technologie#Vorteile

[2] www.profibus.de: Profisafe. (Eingesehen am 18.11.2024). – URL https://www.profibus.de/profisafe#berblick

[3] Balzert, Helmut ; Balzert, Helmut: Was ist eine Softwarearchitektur? In: *Lehrbuch der Softwaretechnik: Entwurf, Implementierung, Installation und Betrieb* (2011), S. 23–27

[4] Chughtai, O Vogel I Arnold A. ; Mehlig, E Ihler T Kehrer U. ; Zdun, U: Software-Architektur. (2009)

[5] Dunkel, Jürgen ; Holitschke, Andreas: *Softwarearchitektur für die Praxis*. Springer-Verlag, 2013

[6] Hasselbring, Wilhelm: Software-Architektur. In: *Informatik-Spektrum* 29 (2006), S. 48–52

[7] Klostermeier, Robin ; Haag, Steffi ; Benlian, Alexander: Geschäftsmodelle digitaler zwillinge. In: *Geschäftsmodelle digitaler Zwillinge: HMD Best Paper Award 2018* (2020), S. 1–35

[8] Rückert, Frank U. ; Sauer, Michael (Hrsg.): *Die Erstellung eines digitalen Zwillings Eine Einführung in Simcenter Amesim*. Springer Vieweg, 2021 (essentials). – URL https://doi.org/10.1007/978-3-658-33407-9

Glossar

Axialkolbenpumpe Eine Hydraulische Pumpe, die Flüssigkeiten durch eine axiale Bewegung von mehreren Zylinder fördert..

Client-Server-Modell Hierbei handelt es sich um eine Architektur, in der ein Client Anfragen an den Server stellt, der diese bearbeitet und die Ergebnisse zurückliefert.

Echtzeitdaten Bei Echtzeitdaten handelt es sich Informationen, die ohne Verzögerung oder mit geringer Latenz übertragen werden, sodass diese unmittelbar analysiert und verarbeitet werden können..

Master-Slave-Modell Ein Kommunikationsprinzip, bei demalle Slaves dem Master untergeordnet sind. Durch den Master werden alle Aktionen initialisiert und die Slaves reagieren nur auf sie..

Peripherie Als Peripherie werden in der Automatisierungstechnik alle externen Komponenten bezeichnet, die an ein Steuerungssystem angeschlossen werden. Dazu gehören Geräte wie Sensoren, Aktoren..

PLCSIM-Advanced Eine Software von Siemens zur Simulation von SPS-Programmen in einer virtuellen Umgebung..

Profinet-Nutzerorganisation Eine Zusammenschluss von Unternehmen, die sich der Weiterentwicklung und Standardisierung der Profinet-Kommunikation widmet. Durch sie werden Standards für industrielle Netzwerke festgelegt.

Profisafe Ein Sicherheitsprotokoll, das Profinet und Profibus verwendet wird. Es sorgt für eine sichere Datenübertragung für sicherheitskritische Anwendungen.

Provider-Consumer-Modell Ein Prinzp in dem der Provider/Anbieter) Daten oder Dienste zur verfügung stellt die vom Consumer(Verbraucher) genutzt werden..

Ring-Topologie Eine Netzwerktopologie, bei der die Teilnehmer in einem geschlossenen Kreis verbunden sind. Daten werden von Gerät zu Gerät weitergeleitet.

Sensoren Bauteile, die physikalische, chemische oder Größen erfassen und diese in ein elektrisches Signal umwandeln..

TIA-Portal Eine von SImens entwickelte Softwareplattform, die zur Programmierung, Konfigurierung und Überwachung von Automatisierungssystemen dient. Sie ermöglicht die Entwicklung von Steuerungs-, HMI-, und Antriebssystemen..

Zahnradpumpe Zahnradpumpen fördern Flüssigkeiten durch das Ineinandergreifen von Zahnrädern..

Erklärung zur selbständigen Bearbeitung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne fremde Hilfe selbständig verfasst und nur die angegebenen Hilfsmittel benutzt habe. Wörtlich oder dem Sinn nach aus anderen Werken entnommene Stellen sind unter Angabe der Quellen kenntlich gemacht.

Ort	Datum	Unterschrift im Original