YRKESHÖGSKOLAN NOVIA

Industriell kommunikation

Implementering av OPC UA-kommunikationsprotokoll

Raoul Kyrönlahti

Examensarbete för ingenjörs (YH)-examen

Utbildningsprogrammet för el- och automationsteknik

Vasa 2020

EXAMENSARBETE

Författare:	Raoul Kyrönlahti
Utbildning och ort:	El- och automationsteknik
Inriktningsalternativ:	Automationsteknik
Handledare:	Roger Mäntylä / Jan Berglund

Titel: Industriell kommunikation

Datum 29.9.2020 Sidantal 42 Bilagor 1

Abstrakt

Detta examensarbete är utfört på uppdrag av Yrkeshögskolan Novia och behandlar en del av ett större projekt som är utfört för skolan gällande OPC UA kommunikation för industriellt bruk.

Syftet med detta examensarbete var att skapa ett ramverk för läromedel inom kommunikationsteknik som grundar sig på industri 4.0. Industrin håller på att genomgå sin fjärde industriella revolution (industri 4.0) som baserats på ”Internet of Things” genom att integrera industriella processer till internet som cyberfysiska system eller med automatiserade maskiner och datacenters. Detta ger en möjlighet för Yrkeshögskolan Novia till att bredda och förnya sin kunskap inom kommunikationsteknik. Målet är att skolan kan ligga i framkant med industrin och dess genomgående förändringar för ny standardisering.

Metoderna som använts är att forska och samla all den teoretiska informationen om OPC UA gällande dess arkitekturella uppbyggnad och funktionalitet. Teoridelen har sedan praktiskt implementerats till projektets simulerade industriella process för varierande test av den funktionalitet OPC UA erbjuder. En funktionalitets beskrivning av en PLC och dess syfte är även medtaget i teoridelen.

Examensarbetet resulterade till skapandet av tre dokument och en laboration. Dessa kan användas som en grund till att skapa olika typer av industriella processer med PLC-programmering och OPC UA-kommunikation inom automationsteknik, anpassade både för lokal och global nätverksstruktur. Arbetet uppnådde sitt mål med att bli ett ramverk som kan användas inom industriell kommunikation för Yrkeshögskolan Novia och lämna utvecklingsmöjligheter för skolan att bygga vidare på.

Språk: svenska
Nyckelord: OPC UA, PLC, automation, industri 4.0, kommunikationsteknik

---

BACHELOR'S THESIS

Author:	Raoul Kyrönlahti
Degree Programme:	Electrical Engineering and Automation
Specialization:	Automation
Supervisor(s):	Roger Mäntylä / Jan Berglund

Title: Industrial Communication

Date 20.9.2020 Number of pages 42 Appendices 1

Abstract

This thesis was commissioned by Novia University of Applied Sciences and deals with a part of a larger project carried out for the school regarding OPC UA communication for industrial use.

The purpose of this thesis is to create a courseware framework for communication technology based on industry 4.0. The Industry is undergoing its fourth industrial revolution (Industry 4.0) which is based on the Internet of Things, by integrating industrial processes with the internet as cyber-physical systems or with automated machines and data centers. This opens an opportunity for Novia University of Applied Sciences to broaden and renew its knowledge within communication technology. The school's goal is to be a front runner alongside with the industry and its ongoing changes for a new standardization.

The methods used were to research and gather all the theoretical information about OPC UA regarding its architectural structure and functionality. The theoretical part was then practically implemented into the project's simulated industrial process for various tests of the functionality that OPC UA offers. In addition to this, a basic understanding of a PLC and its purpose is included in the theory.

This thesis work resulted in the creation of three documents and a lab assignment. These can be used as a base for creating different types of industrial processes combining PLC programming and OPC UA communication within local and global networks. The project achieved its goal of becoming a framework that can be used for industrial communication by Novia University of Applied Sciences, and leaves head room for further development.

Language: Swedish
Key words: OPC UA, PLC, automation, industry 4.0, communication technology

---

Innehållsförteckning

Förkortningar och akronymer

API	Application Programming Interface
CPS	Cyber-Physical Systems
ERP	Enterprise Resource Planning
HTTP(S)	Hypertext Transfer Protocol (Secure)
I/O	In och utgångar
ICT	Information and Communication Technology
IEC	International Electrotechnical Commission
IoE	Internet of Everything
IoP	Internet of People
IoT	Internet of Things
IT	Information Technology
MES	Manufactoring Execution System
OPC UA	Open Platform Communications Unified Architecture
OT	Operational Technology
PLC	Programmable Logic Controller
SDK	Software Development Kit
SOAP[1]	Simple Object Accsess Protocol
TLS	Transport Layer Security
W3C	World Wide Web Contsortium
WSS	Web Service Security
XML	Extensible Markup Languge

[1] SOAP var en akronym för Simple Object Access Protocol i ett tidigare skede, men sedan version 1.2 är namnet inte längre en akronym. SOAP är en av de tre grundstenarna som utgör webservices, de övriga två är UDDI och WSDL. SOAP är standardiserat av W3C

1 Inledning

Detta examensarbete beskriver den teori och metodik som används till att konfigurera en industriell kommunikationsprocess med hjälp av OPC UA.

1.1 Bakgrund

Den globala samhällsstrukturen håller på att digitaliseras allt mer och industrin genomgår sin fjärde industriella revolution (Industri 4.0) för tillfället. Syftet med detta examensarbete är att ta fram ett ramverk inom kommunikationsteknik som grundar sig på industri 4.0. Examensarbetet har en inriktning på automationsteknologi på Yrkesskolan Novia. Skolan har planer att ligga i framkant med industrin, gällande framtida kommunikationsmetoder och automationsteknik.

1.2 Uppdragsgivare

Uppdragsgivare för detta examensarbete är Yrkeshögskolan Novia, Vasa enheten. Skolan är godkänd av det Nationella centret för utbildningsutvärdering (NCU). De erbjuder en högklassig yrkesutbildning med anknytning till arbetslivet.

Yrkeshögskolan Novia är en internationell yrkeshögskola med fem institutioner och innehar internationella samarbetsavtal. Novia är även den största svenskspråkiga yrkeshögskolan i Finland med ett starkt nätverk i Svenskfinland. Novias utbildningsverksamhet sträcker sig längs med den finländska kusten i städerna Jakobstad, Vasa, Åbo och Raseborg.

1.3 Projekt

Projektet består av två delar, den första delen ska vara ett teoretiskt underlag för läromedel kring OPC UA, som kommer att användas till framtida utbildningssyfte kring industriellkommunikation på Yrkeshögskolan Novia. Den teoretiska delen består av funktions principer och den arkitekturella uppbyggnad som utgör OPC UA och dess metodik.

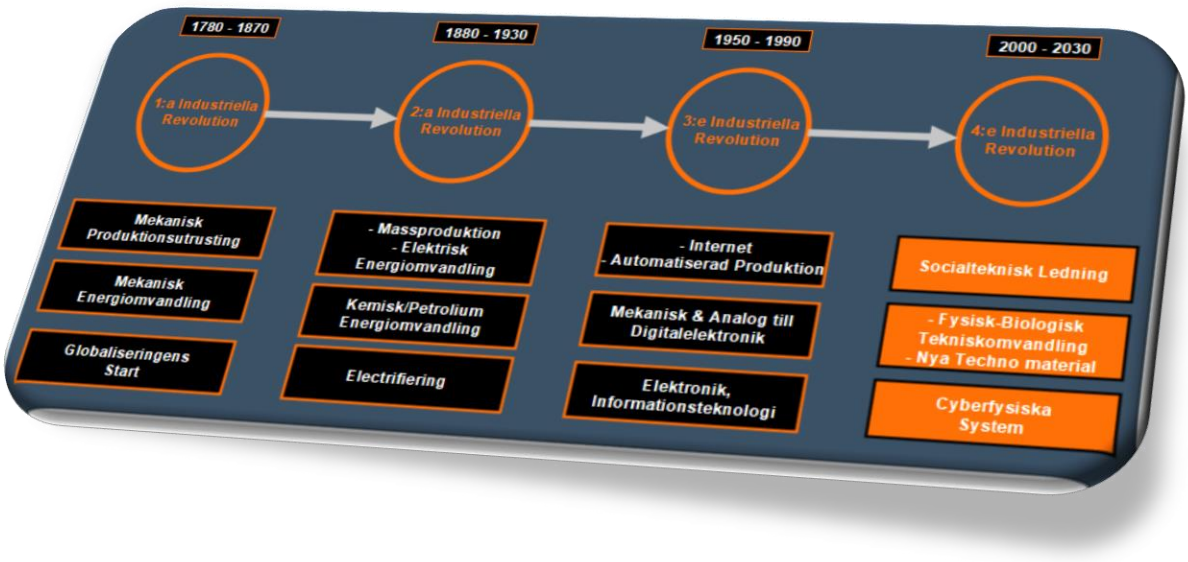
Del två består av olika typer av laborationsprocesser som använder OPC UA-implementering till kommunikationsmetod. Denna praktiska del kommer att användas till att komplimentera den teoretiska delen av kursen i form av laborationsanvisningar eller till eventuella laborationsexempel.

2 De fyra industriella revolutionerna

Traditionellt sätt så har teknologin används till att definiera sättet på hur vetenskaplig kunskap har utvecklats i produktionen av gods och tjänster eller att uppnå mål med hjälp av diverse verktyg och tekniker. Det nuvarande begreppet teknik härstammar från det grekiska ordet ”tekhnologia” (systematisk behandling) från 1500-talet. Vi konstruerade maskiner för att kunna uppnå vissa mål genom att underlätta det analoga arbetet för människan eller ersätta den mänskliga faktorn helt och hållet från vissa typer av arbete. En utveckling ifrån personlig och lokal nivå av användning, transformerades till ett fenomen som förvandlade samhället och den industriella ekonomin [1, p. 3].

Den avgörande faktorn för framväxten av industrialiseringen gjordes med hjälp av utvecklandet av vetenskaplig kunskap och teknik. Förändringen från ett jordbrukarsamhälle med en social organisation till ett industriellsamhälle vars fokus var enbart kring industrin, vi känner till denna förändring mer som ”Den industriella revolutionen”. Begreppet Industriell revolution kom först på slutet av 1700-talet att användas i ”The Lexicon of Thought” [1, p. 4].

Figur 1 visar de största förändringarna som skett från den första- fram till den nuvarande fjärde industriella revolutionen.



Figur 1 Översikt på den industriella revolutionen.

Trenderna inom den industriella revolutionen kan sammanfattas och delas in i tre delar, där den första delen sammanknyter samhällen och expanderar globaliseringen med hjälp av mekanik, elektrifiering, petrokemisk[2] förbränning och internetdigitalisering. Del två omfattar nyttjandet av energiomvandling för arbete och i den tredje delen handlar det om att optimera massproduktion och skapa mekanismer som kan överstiga människans begränsningar med hjälp av maskinautomatisering [1, p. 7].

2.1 Industri 1.0, 2.0 och 3.0

Industriella revolutionen **1.0** ansvarade för vår övergång ifrån ett manuellt arbete till **maskinarbete** och är kopplat till uppfinningen av ångmotorn på slutet av 1600-talet. Detta påverkade inte enbart utvecklingen av vetenskap och teknik, utan ändrade hela samhällsstrukturen [2, p. 13] .

Industriella revolutionen **2.0** var kopplat till **elektricitet** och produktionen av transportband under 1900-talet och resulterade i en ökad arbetseffektivitet, vilket ledde till ett behov av strukturella ändringar i företagsledningen. Den största modifiering i samhällsstrukturen under industriella revolutionen 2.0 var ändringarna inom energisektorn, där ångan ersattes av elektricitet [2, p. 15]. 1867 var det inledande året till förändringarna, det var året då Werner von Siemens upptäckte den dynamoelektriska principen, denna upptäckt väckte idén om att använda elektricitet som en kraftkälla [3]. Det skedde även stora ändringar i det sociala samhället, födelsen och etableringen av konsumentsamhället – där samhället var inriktat på materiella värden och konsumtion, vilket i sin tur ledde till att social status i form av utbildning, yrke och nationalitet blev sekundärt då det kom till ekonomiska faktorer och klass status [2, p. 16].

Industriella revolutionen **3.0** började med mikro-elektronik och halvledar-utveckling i början på 1950-talet och har definierats som den **digitala revolutionen** [1, p. 8]. Under perioden 3.0 skedde det också en stor del komplexa omvandlingar bland olika strukturer, system, institut och tekniker. Detta ledde till ändringar inom kommunikationen, produktionsorganisationen, utbildningen, konsumtionen och på fritiden. Företagsverksamheten under 3.0 underlättade det för människor att inte behöva flytta från sitt eget land för att skaffa anställning på ett internationellt företag. Skattebelastningen optimerades genom skapandet av diverse finansiella flöden [2, p. 17]. Människor och

[2] **Petrokemi** är en avdelning/verksamhet inom den industriella kemin som raffinerar petroleum, naturgas samt de sidoprodukter som uppstår från processen.

Industrier kopplades samman till en skala som aldrig skådats tidigare. Övergången från mekanisk- och analogteknik till digitalteknik påskyndades av den integrerade kretsen. Informationen digitaliserades och datoranvändningen ökade dramatiskt. I detta skede kliver vi in i en tid av industriell informationsteknologi med komponenter från Microsoft, IBM, HP med flera som drev en expansion mot automatiserade tjänster. Stora satsningar inom telekommunikationsinfrastrukturen ledde till internets födelse under 1990-talet [1, p. 8].

2.2 Industri 4.0

Ett tidigt koncept av det som kommer att bli kallat den fjärde Industriella revolutionen startades av den tyska regeringen under 2010 som ett projektinitiativ till att integrera industriella processer till internet som ”**Cyberfysiska system**[3]” eller med automatiserade maskiner och datacenters [1, p. 9]. **Industri 4.0** är baserat på konceptet ”Internet of things” (IoT) och innebär att varje fysiskt objekt, komponent eller ”sak” ska vara inbyggd med digitalteknik som möjliggör en samverkan mellan andra objekt och människor, men begränsas inte enbart till objekt utan sträcker sig till att hela fabriker är anslutna till ett större nätverk [2, p. 18]. Utöver industrisektorn kommer det ske förändringar inom de sociala- och samhällsstrukturerna, samt inom den ekonomiska strukturen under denna fjärde industriella revolution [1, p. 9].

Definiering av den fjärde Industriella revolutionen (4IR)

- 4IR handlar inte enbart om smarta och uppkopplade maskiner/system, dess omfattning berör ett bredare område som sträcker sig allt från **gensekvensering** till **nanoteknologi** och från **förnybar energi** till **kvantberäkning**[4].

- Vad som urskiljer 4IR från tidigare revolutioner är sammanslagningen av **teknik** och **interaktion** mellan **fysiska**, **digitala** och de **biologiska** områdena.

Definition: Industri 4.0 (I4.0)

- Avser sammanlöpningen mellan **industriellproduktion** och **informations-** och **kommunikationsteknik**.

[3] **Cyberfysiska system (CPS)** – Ett system vars mekanism är kontrollerat eller övervakat av datorbaserade algoritmer
[4] **Kvantberäkning (Quantum Computing)** - är en experimentell datavetenskap som använder kvantmekanik för att skapa en ny typ av dator. Kvantdatorer har potential att behandla stora mängder information med hjälp av qubits-som har ett oändligt antal värden-i stället för bitar.

- Avser en relation till sammanlöpningen mellan **IoT**, **IoP** och **IoE**.

En radikal ändring av arbetsstrukturen kommer att ske, den tredje industriella revolutionen slukade jobb från diverse specialister och den fjärde kommer att fortsätta i samma riktning. Då många arbetare inom det autonoma styret kontrollerar arbetet med hjälp av maskiner och datorer så kommer den nya revolutionen 4.0 tillåta maskiner att agera utan mänskliga störningar, detta kommer även ha en påverkan på rollfördelningen mellan olika länder. Det som denna revolution skiljer sig åt från de tidigare, är att den räknar med att utesluta den mänskliga faktorn helt från produktionssystemet och säkerställa en absolut automatisering av produktionsprocessen kombinerat med en utveckling av globala industriella nätverk, för att kunna eliminera alla de negativa sociala konsekvenserna. Detta kommer att öppna upp möjligheter för utveckling av framtidens ekonomiska system då den verkliga sektorn[5] kommer att behöva moderniseras till följd av revolutionens påverkan i samtliga områden och delområden inom den nationella ekonomin [2, pp. 19-28].

2.2.1 OPC UA & Industri 4.0

Industri 4.0 drivs av avancerad **ICT** i form av **intelligenta system**, **virtuella-** och **verkliga system** samt **digitala data** som blir allt mer vanligare inom industriell automatisering. Dessa sammanflätas till ett nätverk som blir till ett OPC UA-baserat **CPS** som bildar ”smarta objekt”, som i sin tur implementeras i ”smarta fabriker” [4] , detta upplägg skiljer sig åt från den traditionella automationspyramiden som använder sig av en central PLC. Produktionsmodulerna som skiljer sig åt i både kvantitet och kvalitet inom smarta fabriker kommer att kunna testas med hjälp av en plug-and-play inspirerad ”**Plug and Produce**[6]” modell [5].

OPC UA är en standard som uppfyller kraven för industri 4.0 gällande semantisk[7] interoperalitet och erbjuder **protokollen** och **tjänsterna** för ”Hur”, samt **informationsmodeller** för ”Vad”. Detta möjliggör ett smidigt utbyte av komplexa data mellan applikationer som utvecklats av oberoende part [6].

[5] **Verklig sektor** – Är i ekonomin, sammansättningen av alla grenar av materiell och immateriell produktion, med undantag för finansiella tjänster. Detta är den klassiska definitionen inom ekonomi idag. Men denna term har många motståndare.
[6] **Plug and Produce** – En informativintensiv process som kräver att bägge enheter är anslutna och att systemet innehåller ett strukturerat informations utbyte om det operativa sammanhangen med tillhörande begränsningar.
[7] **Semantik s**orterar under lingvistiken, men ses också som en viktig del inom filosofi, logik och därmed inom matematik och datavetenskap för att beskriva och hantera strukturen hos ett system.

3 OPC – OLE Process Control

En grupp aktörer inom automationsbranschen gick ihop med Microsoft för att utveckla en klient/server baserad kommunikationsarkitektur vid namn av OPC och blev klart under 1995. De kommande tio åren stod OPC för att vara det mest effektiva och användbara sättet inom samtliga industrier till att kommunicera inom automationssektorn. Utvecklingen av OPC fortsatte från grunden med **DA** som gick vidare till **AE** och fortsatte till **HDA** som innehöll mer avancerade protokoll och funktioner än de tidigare. Kontrollsystemen blev allt mer avancerade med tiden och begränsningar som uppstod kunde täckas med hjälp av OPC. Det var ur detta behov gällande modellbaserade data och efterfrågan av ett plattformsoberoende som **OPC UA** skapades [7].

3.1 Klassiska protokoll

OPC:s klassiska protokoll har inget gemensamt med varandra och står helt självständigt, vilket innebär exempelvis att kvalitetsfältet i DA och HDA inte är sammankopplade på något vis eller har någon som helst anknytning till varandra Samtliga protokoll har individuella kommandon för läs och skriv som endast påverkar det individuella protokollet i sig [7]. Till de klassiska protokollen hör följande:

- **DA** – Data Access: Är det mest primitiva protokoll som distribuerar data till befintliga system på fabriksgolvet från kontrollsystemet. Där samtliga enskilda datapunkter/taggar innehåller ett antal informationsfält [7].
    - **Value –** Värde på datapunkten.
    - **Name** – Namnet på datapunkten.
    - **Timestamp –** Beskrivande kontext kring värdet samt tid för dataextraktion, som sker via OPC servern eller i det underliggande systemet.
    - **Quality –** Anger om data är ok.
- **AE** – Alarms & Events: Det som skiljer detta protokoll åt från DA, är att händelser inte innehåller något värde i sig, av den anledning att detta är en prenumerationsbaserad tjänst. Vilket betyder att alla klienter får samtliga händelser som kommer in och utesluter behovet för **Name** och **Tag** fält. Då relevansen är hög

för att hålla reda på samtliga event, finns det både ett **Timestamp** och **Quality** fält med till varje individuellt event [7].

- **HDA** – Historical Data Access: Protokollet innehåller historiska data och stödjer långa data sets med data allt från ett till ett flertal data punkter och kan hämtas från servern i små eller stora delar. Protokollet var designat på ett strukturerat sätt till att få ut och leverera historik data som lagrats i ett **SCADA** eller **Historian system** (OSI-PI eller GE Historian). Introduceringen av OPC UA kommer troligen göra att detta protokoll används ännu mindre än vad det redan görs [7].

- **XML DA** – XML Data Access: Var designad för internetåtkomst och företagsintegration, specifikationen var först med att vara plattformsoberoende och ersatte COM/DCOM med **HTTP/SOAP** och webbservice teknologier. När Denna leverantörs- och plattformsneutrala kommunikationsinfrastruktur introducerades, reducerades funktionaliteten för DA endast till de åtta metoder krävs för att täcka nyckelfunktionerna och var möjligt på grund av att typiska webtjänster är statslösa. Denna applikation var inte så framgångsrik som man förväntat, detta berodde på högresursförbrukning och begränsad prestanda [8, p. 7]. De åtta metoder som utgör nyckelfunktionerna är följande:

    - **GetStatus** – Verifiera servens status.
    - **Read** och **Write –** Läsa/skriva ett eller flera objektvärden.
    - **Browse** och **GetProperties –** Hämta information om tillgängliga objekt.
    - **Subscribe –** Skapa prenumeration från objektlista.
    - **SubscriptionPolledRefresh –** Utbyte av ändrade värden från en prenumeration.
    - **SubscriptionCancel –** Avsluta en prenumeration.

- **DX** – Data eXchange: Detta är ett tillägg för DA och definierar beskrivningen och transporteringen för värden av datatyper med komplexstruktur. Genom att definiera klientbeteendet och konfigurationsgränssnittet hos klienten på servern, specificerar

DX datautbytet mellan data access-servrar och används vid **batch**[8]-processernas speciella behov [8, p. 6].

4 OPC UA

OPC UA är en öppen standard som specificerar informationsutbyte för industriell kommunikation. Mer specifikt på enheter mellan maskiner och från maskiner till system med en sammanlöpning av **IT** och **OT**. Detta protokoll är inte begränsad till någon specifik mjukvaruplattform utan kan användas av de flesta plattformar som Windows, Linux, Mac Os, Android och många fler. Det är inte bara flexibiliteten över mjukvaruplattformarna som gör UPC UA unikt, protokollet kan användas alltifrån små slutna nätverk till massiva moln infrastrukturer med informationsutbyte öppet över nätet, detta uppnås genom att ha säkerheten inbyggd i protokollet i form av **åtkomstkontroll**, **autentisering** och **kryptering**. Denna teknik har blivit eftertraktad inom industrin tack vare flexibiliteten och har tagits i bruk inom ett flertal industriella kategorier som bil, dryck, olja, gas, energi samt många automatiserings uppbyggnader. Utöver flexibiliteten uppfyller OPC UA kraven för **industri 4.0** vilket gör protokollet framtidssäkert [9].

Figur 2 Beskrivning IT & OT (Källa: Otorio.com)

[8] **OPC Batch** – Specifikation som förser ett gränssnitt för utbyte av utrustningskapacitet (motsvarande S88.01 fysisk modell) och de aktuella driftförhållandena.

4.1 Specifikation

Specifikationerna för OPC UA partitioneras in i mindre delar för att uppnå kriterierna till att få en IEC-standardisering. Benämningen OPC UA fick i standardiseringen kallas för **IEC 62541**. Figur 3 visar en översikt på partitionsuppdelningen av samtliga specifikationer [10].

Figur 3 OPC UA-specifikationsdelar

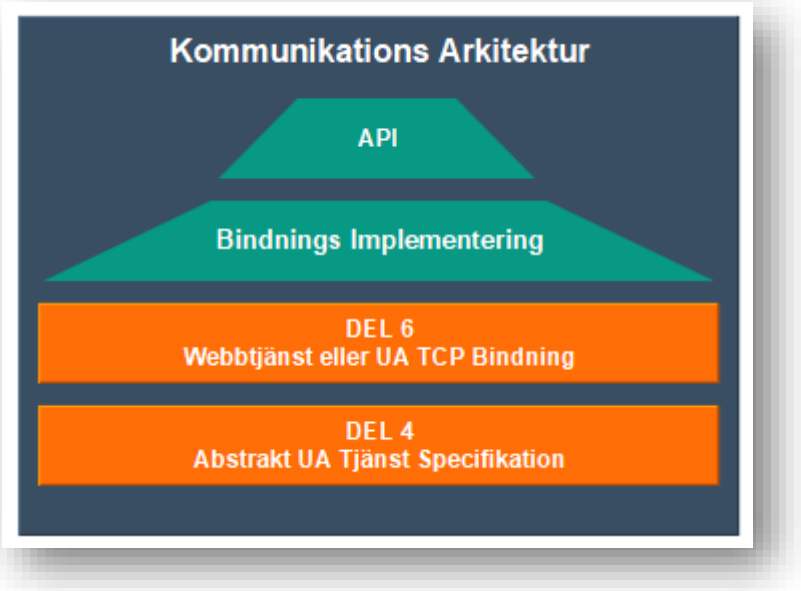
4.1.1 Core Specification Parts

Del 1 ger en översikt över OPC UA, säkerhetsmodellen och säkerhetskraven beskrivs i **del 2**. Dessa två är inte styrande delar [10].

Del 3 och 4 är nyckeldokument för applikationsdesign och utveckling för OPC UA. Modellering och informationsåtkomst görs inom dessa två specifikationsdelar. I **del 3** finns adressutrymmesmodellen (The Address Space Model), i denna modell specificeras grunderna för att visa datamängd och typinformation till *metamodellen*[9] för OPA UA. Detta

[9] **Metamodell**: En modell av en modell, analysen, konstruktionen och utvecklingen av ramverken, reglerna, begränsningarna - modeller och teorier som är tillämpliga och användbara för att modellera en fördefinierad problemklass

används till att uppvisa och beskriva informationsmodeller samt till att konstruera ett OPC UA-serveradressutrymme. **Del 4** företräder eventuella samspel mellan klient och serverapplikationer i UA. När klienten vill hitta eller få åtkomst till information från servern används denna tjänst. Del 4 definieras som den abstrakta delen av UA-tjänster av den orsaken att informationsutbytet mellan applikationer bestäms, men är inte av den verkliga representationen varken på bindningskabeln eller API:n som applikationerna använder sig av. Figur 4 visar **kommunikationsarkitekturens** alla lager för OPC UA [10].



Figur 4 OPC UA-kommunikationsarkitektur

Del 5 består av basinformationsmodellen, vilket är ramverket för samtliga informationsmodeller som använder sig av OPC UA. Del 5 omfattas av följande delar [10]:

- Klientnavigering för åtgärder och typer av en OPC UA-server genom inmatningspunkterna i adressutrymmet.
- Uppbyggnadsgrund för olika typer av hierarkier genom bastyper.
- Inbyggda objekt- och datatyper.
- Serverobjekt som bidrar med kapacitet och diagnostisk information.

Del 6 Beskriver hur information och data mellan OPC UA-klienter och servrar överförs. Denna del täcker följande punkter [11]:

- Data kodning och avkodningsöversikt, standard datatyps regler samt komplexa datatyper och objekt.
- Säkra konversationer, säkrandet av OPC UA-meddelanden.
- Regler för säkerhetsverifiering.
- Mappning av de olika format på transportprotokoll, UA TCP, SOAP/http och HTTPS.

Del 7 av specifikationen beskriver beteendekategorier som kan användas av OPC UA-klienter och servrar. Detta omfattar följande delar [12]:

- Överstämmelseenheter och profilkoncept.
- Beteendekategorier, funktionalitet och säkerhet som stöds mellan UA-servrar och UA-klienter.
- Detaljerade beskrivningar på statiska/dynamiska beteendekrav för varje profil, gäller även kapslade profiler (Nested Profiles).

Kategorierna för del 7 definieras av två nivåer, I den första nivån finns **överensstämmelser**, denna definierar en liten uppsättning av funktionalitet som kan testas och alltid användas tillsammans med **Compliance Test Tools** och bekräftas sedan som enhet. På nivå två finns profiler, som är sammansatt av en lista över överenstämmelseenheter. När en profil har skapats kommer profilen att verifieras som fullständig under certifieringen av OPC UA-produkter. Under anslutning mellan klient och server utbyts listan över de profiler som stöds och används, detta möjliggör för applikationerna att besluta om kommunikationspartnern stöder de funktioner som krävs [10].

4.1.2 Accsess Type Specification Parts

Del 8 beskriver data access (DA)-applikationer, om hur man använder sig av automationsdata och specifika egenskaper i tekniska enheter och omfattas av följande punkter [13]:

- DA översikt och koncept för datatillgång.

- Beteende regler för datatyper och informationsmodells beskrivning.
- Adressutrymmes organisation.
- Beskrivning och regler gällande **Precent Deadband**[10] beteende.
- Detaljerad beskrivning av felkoder specifikt gällande denna specifikation.

Del 9 (AC) specificerar processalarm och tillståndövervakning, denna specifikation omfattas av följande punkter [14]:

- Översikt och koncept gällande larm och tillstånd samt utvecklingen från OPC Classic Alarms & Event-specifikationen.
- Informationsmodellens beskrivning och beteende regler, samtliga datatyper och förväntat beteende.
- Adressutrymmes organisationen.

Del 10 denna del av specifikationen beskriver Informationsmodellen för program och hur de kan implementeras i olika OPC UA-applikationer. Här definieras en bastillståndsmaskin för exekvering, övervakning och manipulation av program och omfattas av följande punkter [15].

- Programkoncept och dess användning, tillstånd och livscykler.
- Programinformationsmodellens beskrivning och beteenderegler, programtyper, orsak och verkan, parametrar och returkoder.
- Diagnostikimplementering.
- Exempel på programinförande

Del 11 HA (Historical Access) beskriver hur data kan hämtas och arkiveras från en databas samt om hur information gällande datakonfigurationen och händelsehistorik kan presenteras. Denna specifikationsdel omfattas av följande punkter [16]:

[10] Parameter som anger den procentuella förändring i data som krävs för att meddela klienten om en dataändring.

- Översikt och koncept för datahistorik gällande event och utveckling från OPC Classic Historical Data Access.

- Instruktioner och beteenderegler för noder, samtliga datatyper och event inom informationsmodellen.

- Detaljerad beteendebeskrivning för arkiverade data gällande skapa, hämta, uppdatera, radera samt kommentarer/anteckningar.

- Detaljerade beskrivningar gällande kategorierna, säkerhet, åtkomsträttigheter och revision.

4.1.3 Utillity Type Specification Parts

Del 12 Discovery and Global Services beskriver om hur servrar kan upptäckas inom nätverket och hur UA produkterna hanteras på datorn, inom närverksinfrastrukturen samt om hur klienten får tillgång till den nödvändiga informationen för att kunna etablera en anslutning till en specifik server. Specifikations delen omfattar följande ämnen [17]:

- Upptäckprocess.

- Koncept för **LDS** (Local Discovery Server).

- Koncept för **GDS** (Global Discovery Server).

- Certifikatshantering för *Pull* och *Push* [11]-metoder.

- Konfiguration och distribution.

- **KeyCredential Management**, tillåter hantering som OPC UA-applikationerna använder för att få åtkomst till **Authorization Services**. Applikationer som förser hanteringsfunktioner för KeyCredential kallas för **KeyCredentialService** och kombineras oftast med GDS för att få en enda applikation [4].

- Tillståndstjänster (Authorization Services).

[11] **Pullkodning** är en nätverkskommunikationsstil där den första begäran om data kommer från klienten och sedan besvaras av servern. **Push** (push-teknik) är det motsatta, där servern driver data till klienter.

Del 13 är en övergripande OPC UA-del bland specifikationsdelarna som definierar informationsmodellen och användningen av **Aggregate**-funktioner inom UA applikationer [18]. Denna specifikationsdel omfattas av följande punkter:

- Konceptet för ett aggregat, samt vart och hur det ska appliceras på en allmän applikation.
- Detaljerade beskrivningar och uppförandekrav för samtliga 37 aggregat.
- Informationsbeskrivning och beteenderegler för varje individuell datatyp.
- Ett brett bibliotek med referensmaterial med exempel på frågor och resultat för varje individuellt aggregat.

4.1.4 Övriga specifikationsdelar

Del 14 PubSub, en kommunikationsmodell som definierar ett OPC UA publicerings-prenumerationsmönster istället för klientservermönstret som är definierat av tjänsterna i specifikationsdel 4 [19]. Denna specifikationsdel omfattas av följande punkter:

- En allmän introduktion av koncepten.
- Definering av PusbSub kommunikationsparametrar.
- En konfigurationsmodell för PubSub.
- Mappningar för meddelanden och protokoll.

Del 100 Device Information Model, en informationsmodellsspecifikation för enheter [20]. Denna specifikationsdel inkluderar följande:

- TopologyElementType
- FunctionalGroupType
- Identification Functional Group
- DeviceType, DeviceSet Entry Point
- ProtocolType
- Uses Reference Type
- Block Type

- Configurable Components

OPC UA Companion Specification Template: Är en mall som samtliga organisationer kan använda sig av för att skapa en OPC UA informationsmodell.

4.2 Säkerhet

Ramverket inom säkerheten som OPC UA använder sig av för att skydda data, är en standard nätteknologi med både autentisering och kryptering. Detta kräver en handskakning med **X.509 certifikat** mellan klient och server innan någon kommunikation kan ske.

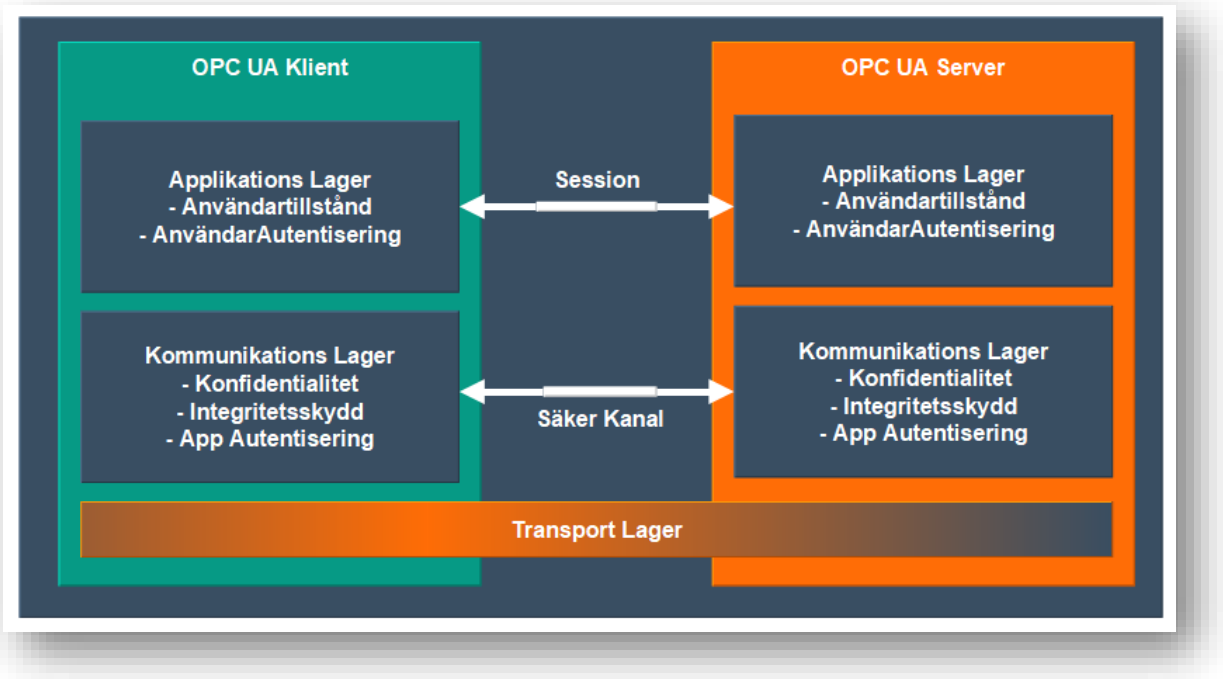
Figur 5 PC UA använder X.509 certifikat

OPC UA stödjer **PKCS12 (Public-Key Cryptography)** standard för att kunna stödja X.509:s privata nycklar samt certifikatfiler som innehåller de publika nycklarna. Val av publika- och privata nycklar kan avgöras av både server och klient. Användaren kan använda sig av tre typer av kommunikations inställningar; **Ingen**, **Sign**, **Sign och Kryptering**. Det finns två typer av säkerhetspolicyn som användaren kan använda sig av, **Basic256** och **Basic128Rsa15**, dessa är algoritmgrunderna till att signera eller kryptera data mellan klient och server [21].

4.2.1 Applikationslagret

Applikationslagret använder sig av ett flertal säkerhetsmekanismer. Identifikation av användare eller operatör sker via ett **användarcertifikat** eller en kombination av **användare/lösenord**. Varje enskild nod kan justeras för åtkomsträttigheter gällande data.

Detta gör det möjligt att dela upp användare i olika åtkomstklasser som till exempel administratör, användare och gäst, där administratören har skrivrättigheter, användaren har endast läsrättigheter och gästen får inte ens tillgång att bläddra/söka igenom nodkatalogen. Applikationslagrets revisionsmekanism gör att en server kan logga alla värden som ändrats vid specifik tidpunkt samt av vilken person. Dessa säkerhetsfunktioner är integrerade delar av specifikationen och UA-stacken [22].



Figur 6 OPC UA-kommunikationslager

4.2.2 Transportlagret

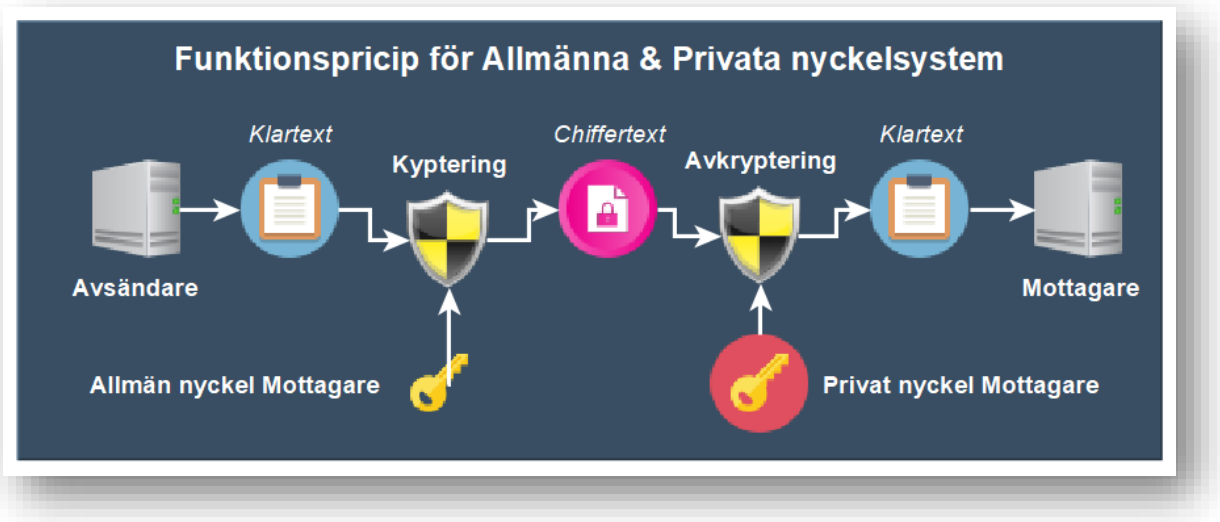
Meddelandesäkerheten för det binära och webbtjänstprotokollet definieras av OPC UA. Hybridvarianten använder sig av **TLS** (Transport Layer Security) till att säkra transportvägen. Säkerhetsmekanismen följer säkerhetsspecifikationerna för webbtjänster (**WSS**) och är inte OPC-stiftelsens skapelse helt från grund. **WS Secure Conversation** används för webbtjänster och säkerställer kompatibiliteten med **.NET** och andra **SOAP**-implementationer. Till den binära varianten så har algoritmer från WS Secure Conversation anpassats och konverterats till en binär likvärdighet som går under namnet **UA Secure Conversation.**

Figur 8 visar att det finns en hybridversion som använder sig av binärkodning och HTTPS gällande transport, vilket är en kompromiss mellan binäreffektiv kodning och

brandvägsvänlig överföring. Till autentisering används endast X.509 certifikat. Applikationsutvecklaren avgör vilket certifikat som är bundet till UA-applikationen [22].

4.2.3 Autentisering – X.509

X.509 certifikatet är en del av X.500 katalogen och skapades 1988 för att hjälpa användare identifiera en säker anslutning. X.509 certifikatet binder en specifik användare till ett specifikt certifikat genom att skapa ett individualiserat nyckelpar. Detta garanterar användare inom stora organisationer eller företag **laglig rätt** och **integritet**. När parning sker via certifikatet, kontrollerar den identiteten för den länkande användaren. En suverän två-polig strategi till att säkra webbplatser och anslutningar. Ytterligare säkerhetsfunktioner mot **Phishing** och **Malware** lades till efter 2011, då det skedde en del attacker och andra typer av säkerhetskompromisser [23].



Figur 7 Funktionsprincip nyckelsystem

En summering av allt som ingår i en standard X.509 certifikat:

- Ett **DN** (distiguished name) används för att styrka användarens identitet.
- En **offentlig** nyckel bunden till användaren.
- X.509 certifikats **version** information och **serienummer**.
- En digital **signatur**.

- Detaljerad information gällande certifikatets **algoritm**.
- Möjligheter för tillägg gällande **utökad säkerhet**.

4.2.4 WS-SecureConversation

Ws-SecureConversation är optimerad för säkrandet av XML-data genom XML-kryptering och signering och är en specifikationsextension till **WS-Security** som genom definition av koncept och teknologi säkerställer data som utbyts via **Web Services** [8, p. 195].

4.2.5 UA-SecureConversation

UA-SecureConversation är OPC Foundations egna säkerhetsprotokoll som består av tekniker kombinerat med mekanismer godkända av TLS[12] standard och WS-SecureConversation. UA-SecureConversation använder de kodade servicemeddelandena som nyttolast och tillägger en säkerhetsrelaterad information både framför och bakom denna last [8, pp. 196-197].

4.3 Protokoll

OPC UA har för närvarande två protokollbeskrivningar (HTTPS, UA TCP) och två typer av kod (Binär, XML) som stöds. En tredje fås via kombination av de två befintliga. Ifall en ny teknik inom datakommunikation skulle börja användas så kan OPC UA anpassas till denna teknik genom en ny definition av mappning. Anledningen till att OPC UA stöder dessa två protokolltyperna och kodningar är att det kommer att användas i olika typer av applikationsdomäner med varierande krav, detta ger ett mer bredare tillämpningsområde att arbeta inom [8, p. 317].

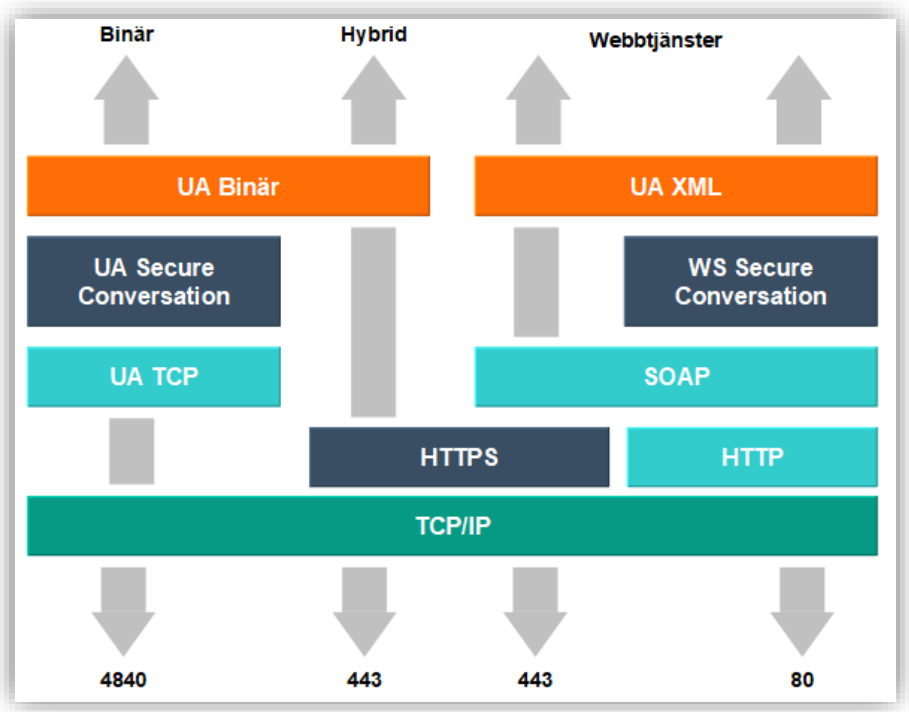
HTTPS används till att köra olika typer av webapplikationer, med möjlighet att korsa brandväggar.

UA TCP används när man med begränsande resurser behöver köra optimerade applikationer via kabel.

Optimeringen handlar inte om jämförandet av HTTPS och UA TCP, utan mer om vilken kodnings typ som körs (binär, XML). Det som skiljer OPC UA från andra protokoll är dess

[12] **TLS** – Transport Layer Security, definierad hos [DR06]

förmåga att köra binära data direkt utan att behöva konvertera data först till XML och sedan till binär, vilket gör det väldigt effektivt.



Figur 8 OPC UA-protokoll

4.4 Datakodning

Datakodning betyder att tjänstemeddelandena (in- och utgångsparametrar) serialiseras[13] till ett nätverksformat. För närvarade specificerar OPC UA två typer av kodningar, **OPC UA Binary** och **XML**. Nätverksrepresentationen i de bägge kodningstyperna använder grupper av primitiva typer som **Boolean**, **Byte** och **Float** till att skapa strukturer och mer komplexa typer. Uppsättningen av dessa specifika primitiva typer kallas även för **inbyggda datatyper** och definieras i del 6 av OPC UA-specifikation [8, p. 192].

4.4.1 OPC UA Binary

OPC-UA är ett dataformat som erbjuder snabb kodning och avkodning av data genom ett effektivt format av kodade data som endast upptar en liten plats. Konceptet bakom *OPC UA Binary* är att en specifik uppsättning inbyggda datatyper (Primitiva data typer) översätts till

[13] **Serialisering** är en process inom datavetenskapen som innebär att en datastruktur eller ett objekttillstånd sparas till ett format som kan lagras i, eller överföras till, ett datorminne eller en annan datamiljö.

en binär representation genom användandet av definierade regler. Det mest effektiva sättet för datautbyte mellan enheter och olika system är att använda sig av **serialisering** och **deserilisering** av serviceparametrar till en binärström [8, p. 193].

Figur 9 Exempel på Stringdata i OPC UA Binary Encoding

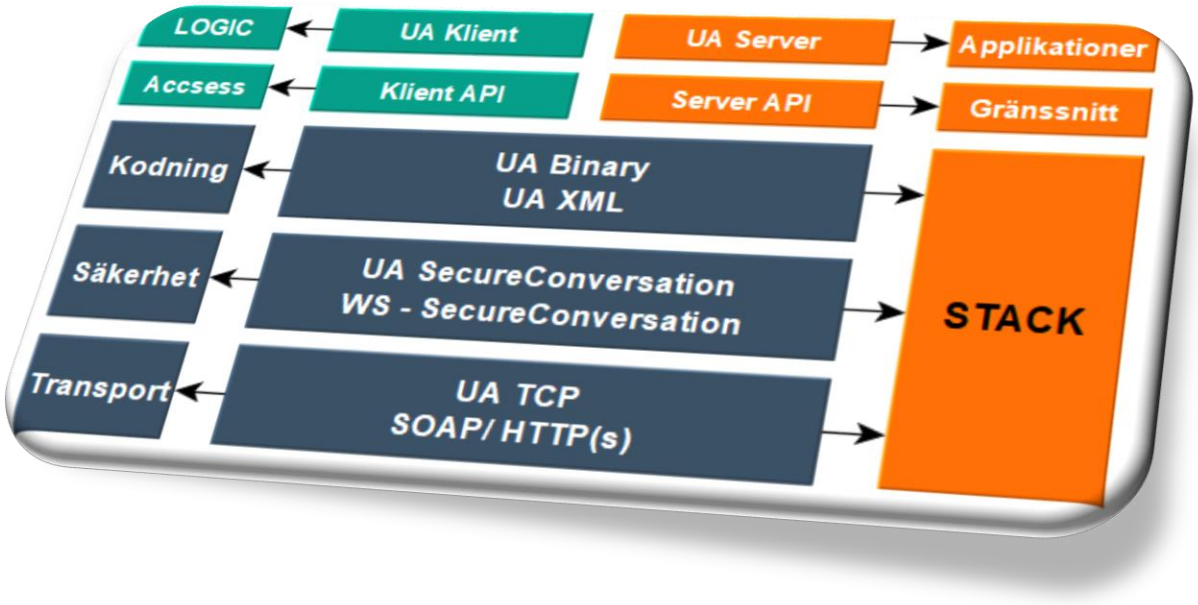
Figur 9 visar hur datatypen **String** ”OPC UA” kodas enligt OPC UA Binär, som använder en sekvens av UTF-8 tecken. Längden på datatypen är fem och blir det första tecknet. ”Null”-strängar kodas genom värdet ”-1”. Mer komplexa datatyper kan skapas genom att kombinera en uppsättning inbyggda datatyper. Sedan sker en sekventiell översättning av de omfattade primitiva datatyper för att få ett binärtformat. Servicemeddelandena som är av abstrakt typ [UA del 4] kodas med samverkan av **ExtensionObjects**.

4.4.2 XML

XML-strukturen är standardiserad och gör det lätt att använda i olika typer av applikationer och plattformar. OPC UA-applikationer förväntas att utbyta data mellan system på operations- (MES) och företagsnivå (ERP) därav är ett XML-stöd ett måste för OPC UA. Vanligtvis kodas datatyperna efter XML specifikationerna [W3C04a] och [W3C04b] men i vissa fall kan det behöva införas speciella användningar och begränsningar [8, p. 193].

4.5 Funktionslager för applikationer

För att ge en interoperalitet mellan OPC UA-produkter och säkra för framtida teknologier definieras tjänster och koncept på ett abstrakt sätt genom en tekniskkartläggning för implementering. Mappningarna riktar sig mot de tre nödvändiga uppgifter (**datakodning**, **kommunikationssäkerhet** och **datatransport**) som krävs för datautbyte mellan OPC UA applikationer. Figur 10 illustrerar en uppdelning av OPC UA-applikationens funktionella lager.

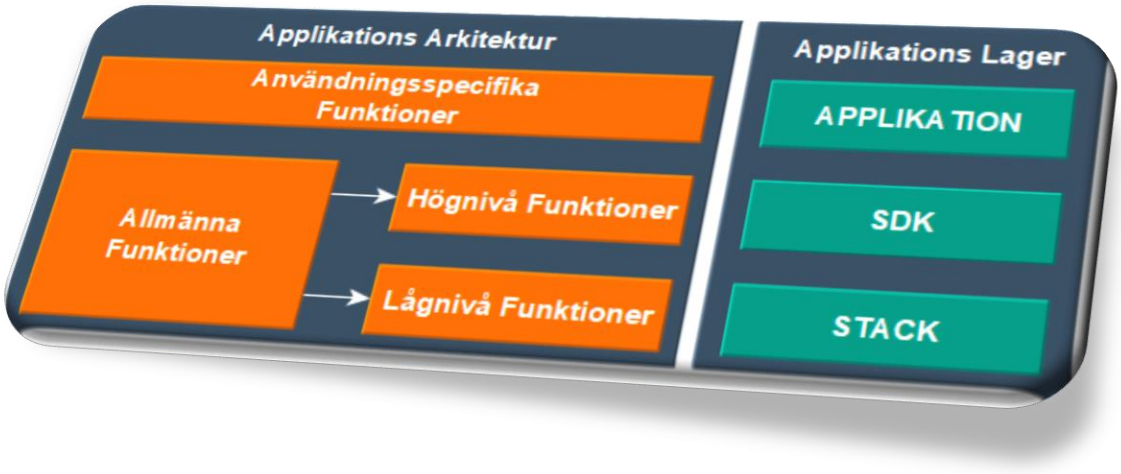


Figur 10 Teknisk översikt av applikationsfunktioner

När det finns fler lager som ansvarar för kodning, säkerhet och transport brukar de komponeras till en stack. En stack består av allmänna komponenter som står separat ifrån den verkliga applikationen. Detta möjliggör att andra applikationer kan återanvända komponenter [8, p. 191].

4.6 Applikationsarkitektur

När man designar applikationsarkitekturen delas den huvudsakligen in i tre delar för att underlätta återanvändning då detta är ett mål. Detta görs genom att separera den användningsspecifika- och allmänna funktionalitetsdelen. Den allmänna delen klyvs ytterligare en gång för att sära på hög- och lågnivå funktioner. Kombinationen av dessa block skapar en arkitektur av mycket hög nivå som figur 11 visar [8, p. 255].

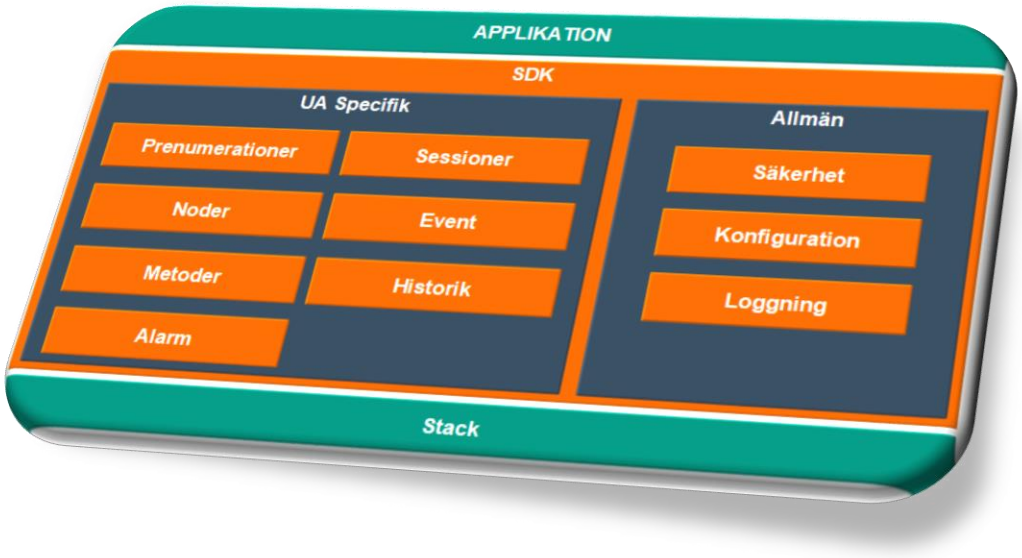


Figur 11 Applikationsarkitektur & lager

Innehållet i **applikationslagret** består av två typer: **klienter** och **servrar**. Strukturerna på dessa varierar efter målsättning.

4.7 SDK

SDK:n hamnar över mjukvarustacken som figur 12 visar och omfattar högnivåfunktionerna. SDK kan i allmänhet delas in i tredelar, **Klient/Server API**, **UA specifik** och **allmän**. Den UA-specifika delen utför OPC UA-koncept och tjänster medan den allmänna delen står för säkerhet, konfiguration och loggning.



Figur 12 SDK-arkitektur

UA SDK innehåller två typer av bibliotek i ***C#***, den ena ett serverbibliotek och den andra ett klientbibliotek. Bägge bibliotek förser basfunktionaliteten till hanteringen av UA-protokollet samt behandlingen av uppgifter gällande samtliga underkategorier i den allmänna delen.

Serverbiblioteket använder sig av ett special gränssnitt till integreringen av underliggandesystem (läsa och skriva vissa värden). **Klientbiblioteket** utför en NodeCache[14] för buffring[15] av noder och referenser. Utöver dessa bibliotek förser SDK även en ***C#***-baserad upptäcktsserver som klienter använder sig av för att identifiera aktiva server slutpunkter till anslutning [8, pp. 258-263].

4.7.1 SDK-utvecklarverktyg

Det finns en del alternativ för utvecklare när det kommer till att arbeta med OPC UA, alltifrån standard OPC-aktiverade produkter till utvecklande av OPC-teknik via specifikationer och verktygssatser. Användning av OPC verktygssatser är det mest optimala sättet att minimera utvecklingskostnaderna. För alla större OPC-specifikationer finns det att hämta utvecklarverktyg från ett flertal OPC Foundation medlemmar. Detta görs via att söka efter ”Toolkits” i OPC Foundations produktbibliotek [24]. Nedan listas några utvecklarverktyg som Prosys OPC erbjuder på deras nätsida.

SDK för Java: är ett högnivå programmeringsgränssnitt som tar hand om alla OPC UA-kommunikationsdetaljer när det kommer till att utveckla OPC UA-servrar/klienter och multiplattforssystem. Detta bidrar till en mer effektiv applikationsutveckling och snabbare utvecklingsprocess [25].

C/C++ SDKs: finns i fyra olika utföranden.

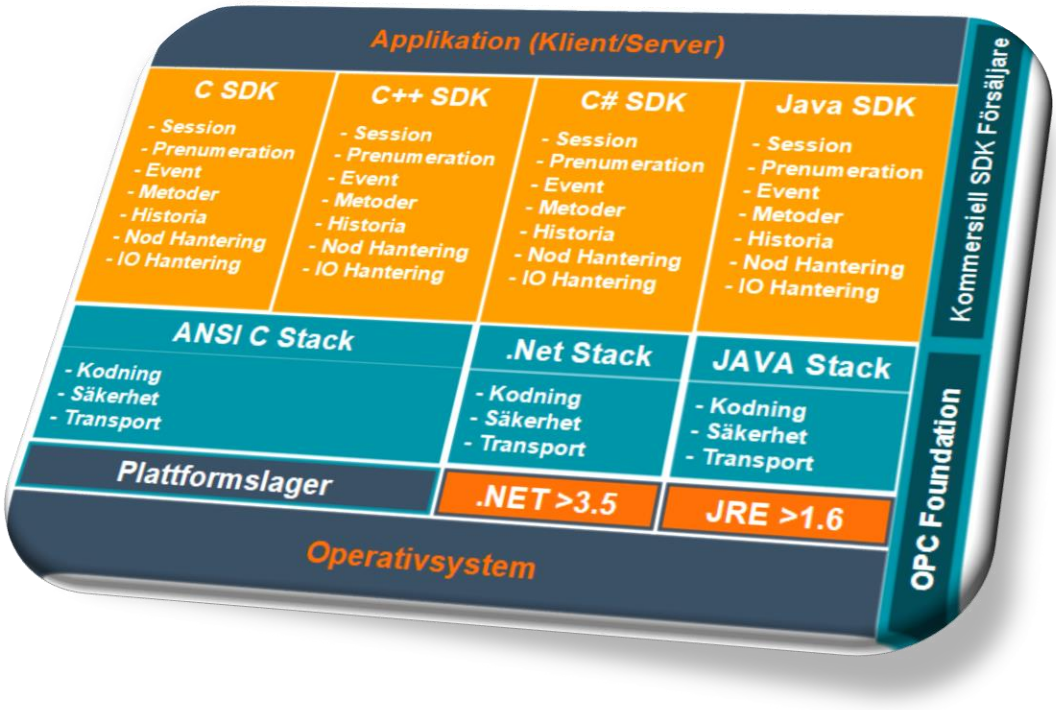
- **ASNI** baserad OPC UA klient SDK
- ***C++*** baserad OPC UA klient SDK
- **ASNI** baserad OPC UA klient/server SDK paket
- ***C++*** baserad OPC UA klient/server SDK paket

[14] **Node/Nod –** En punkt i ett nätverk där linjer korsar eller grenar, **Cache –** En samling data/instruktioner som lagras på en dold eller otillgängligplats.
[15] **Buffra/Buffert -** anordning som mellanlagrar sådant som ska användas inom kort. Buffertens uppgift är att jämna ut skillnader mellan källans och mottagarens förmåga att leverera respektive att ta emot och använda.

ANSI C är främst designad för användning i inbyggda enheter. Denna SDK är skapad att kunna arbeta med begränsade resurser genom att avge minimala minnesavtryck med maximal prestanda. Plattformarna denna SDK riktar sig in på är **Windows**, **Linux**, **vxWorks**, **QNX**, **EUROS** och **RTOS** [26].

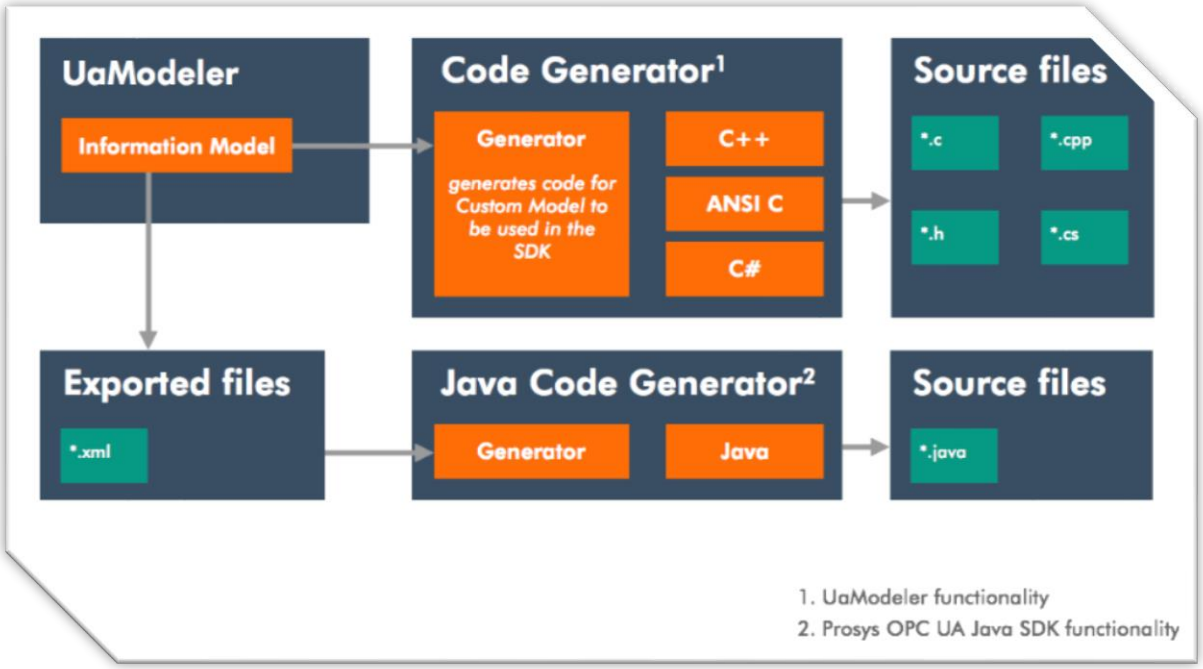
C++ är ett bibliotek som har stöd för att skriva till funktionella och bärbara OPC UA-klienter och servrar. Denna SDK är designad för att passa PC-plattformar samt inbyggda system och består i själva verket av två separata SDK:n (server och klient) som använder samma UA-basbibliotek. Plattformarna denna SDK riktar sig in på är **Windows 32**, **Windows 64**, **Linux**, **vxWorks**, och **QNX** [26].

.NET SDK: är en verktygssats baserad på .NET som används till att utveckla OPC UA servar och klienter kvickt. Verktygssatsen innehåller dokumentation, prover och det som behövs för en komplett sammanställning. Användning av denna SDK kräver minst en **.NET Framework 3.5** och har stöd för alla säkerhetsdefinitioner och funktioner till att skapa .NET applikationer. Det som skiljer Prosys SDK från OPC Foundations SDK, är deras API, som är enkel och användarvänlig samt att de förser med bra dokumentation och användningsexempel [27].



Figur 13 SDK-verktyg

OPC UA Modeler: erbjuder en grafisk design av adressutrymmet med tillägg av noder och referenser. Den grafiska representationen följer OPC UA:s notationer och syntax. Koden genereras sedan via ett knapptryck som påskyndar hela implementeringen, koden produceras välstrukturerad och ”felfri” vilket bidrar till en ökad mjukvarukvalitet [28]. Figur 14 visar en hierarkisk översikt av modelldesignen för OPC UA Modeler.

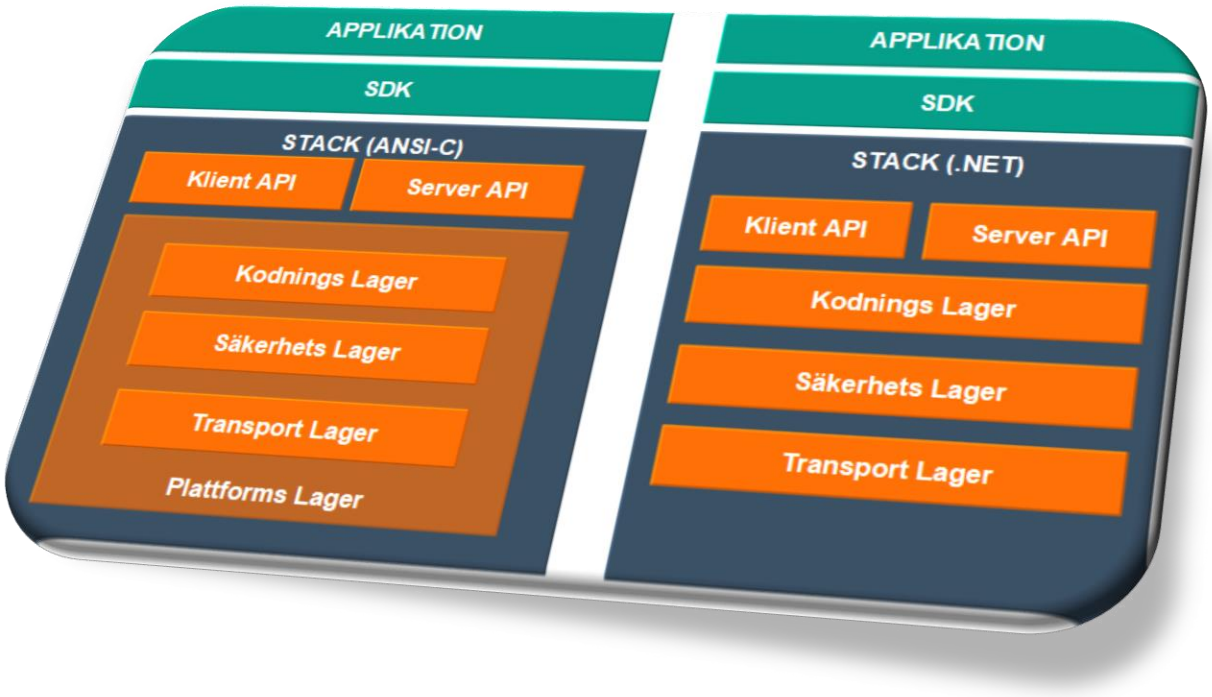


Figur 14 Prosys OPC UA Modeler (Källa. Prosysopc.com)

OPC UA Modeler kan generera programkod för följande produkter; ***C++*** och **ANSI C** baserad OPC UA Server SDK samt **.NET** baserad OPC UA Klient & Server SDK. Utöver dessa, kan Modeler spara modeller till XML-filer som sedan kan användas till att generera kod till **Prosys OPC UA SDK för Java Klient och Server**.

4.8 Stackar

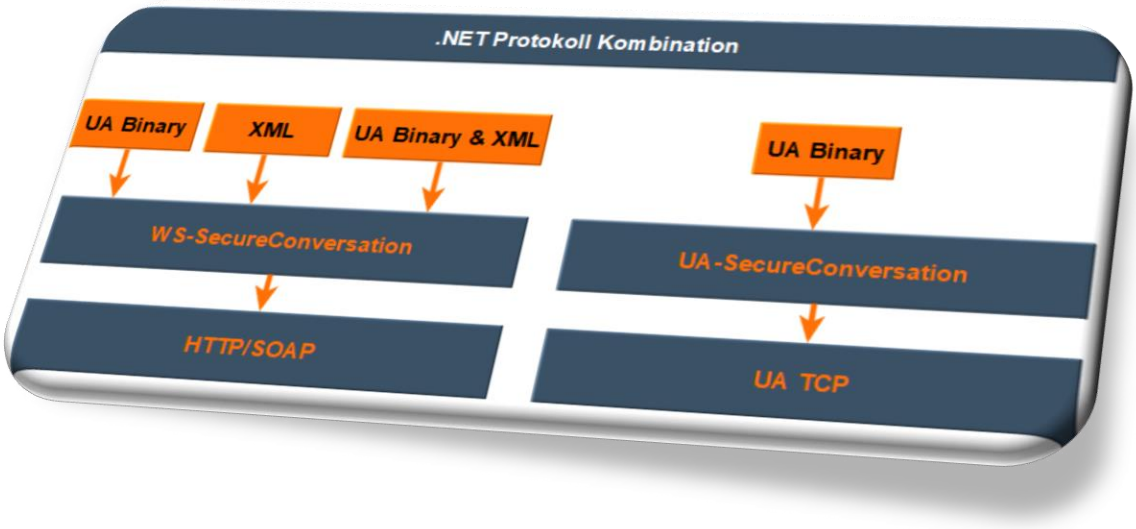
Det rekommenderas att använda OPC Foundations stackar för att säkerställa kompabilitet av applikationer i varierande utvecklingsmiljöer. Stacken är en vanlig del som omfattar den lägre nivåns funktionalitet. Figur 15 visar en mer detaljerad vy av stackens arkitektur [8, p. 258].



Figur 15 ANSI-C & .NET stack arkitektur

ANSI-C stacken implementeras enligt Figur 15 arkitektur, där kodningslagret stöder **binärkodning**, säkerhetslagret använder sig av **SecureConversation** och **UA TCP** för transportlagret. Till säkrandet av meddelanden och validering av certifikat, integreras ett **OpenSSL** kryptobibliotek i den plattformsspecifika delen av stacken [8, p. 262].

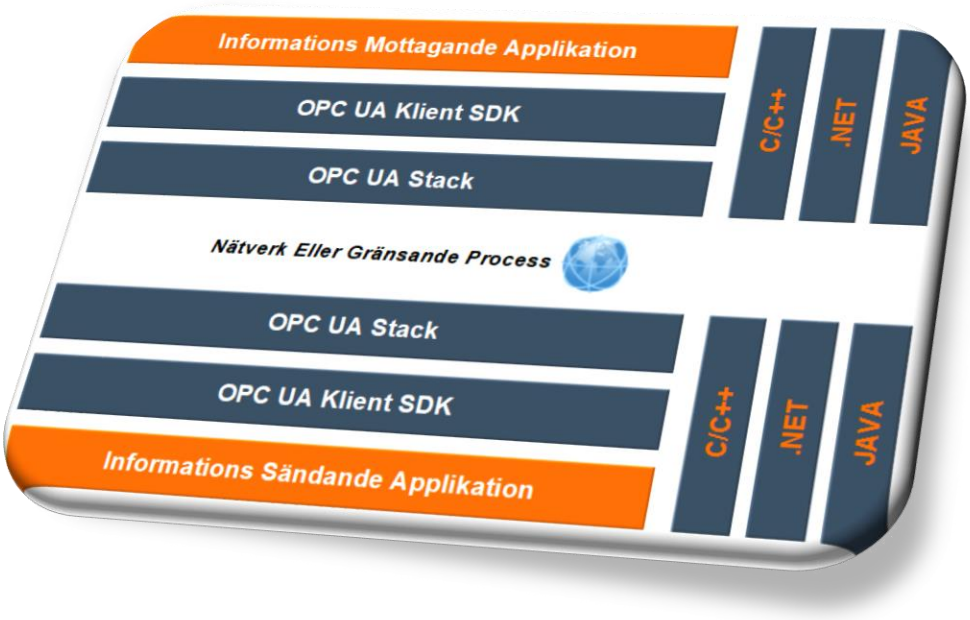
.NET stacken är skriven i *C#* och skiljer något från den ovannämnda, då den inte har ett plattformslager. Skillnaden i arkitekturen är att kodningslagret stöder både **binärkodning** och **XML**, säkerhetslagret använder sig av protokollen **UA-SecureConversation** och **WS-SecureConversation** och transportlagret använder sig av **TCP** och **SOAP/HTTP** protokoll som kan kombineras på följande sätt [8, p. 262]:



Figur 16 .NET protokollkombination

4.9 Mjukvarulager

OPC UA använder sig av ett klient-serversystem, definitionen för en UA-server är när en applikation vill utbyta eller uppvisa sin data för andra applikationer och vice versa när det kommer till UA-klient, en applikation som vill använda sig av data från andra applikationer. Det kommer däremot finnas applikationer som fungerar som både server och klient, anledningen till detta är att allt fler UA-servar kommer att integreras rakt in i enheterna och möjliggör för direktkommunikation mellan enheter. Ännu en fördel med detta är att man kan använda sig av OPC UA som ett konfigurationsgränssnitt för konfigurering av UA-servrar och UA-klienter [8, p. 14].



Figur 17 OPC UA mjukvarulager

Figur 17 visar OPC UA-applikationens tre skikt av mjukvara och de programmeringsspråken som används. Mjukvarustacken[16] kan köras med C/C++, JAVA och .NET, men är inte begränsat specifikt till dessa programmeringsspråk eller utvecklingsplattformar. OPC Foundation använder sig av endast dessa tre i nuläget för att köra OPC UA stack-leveranser [8, p. 14].

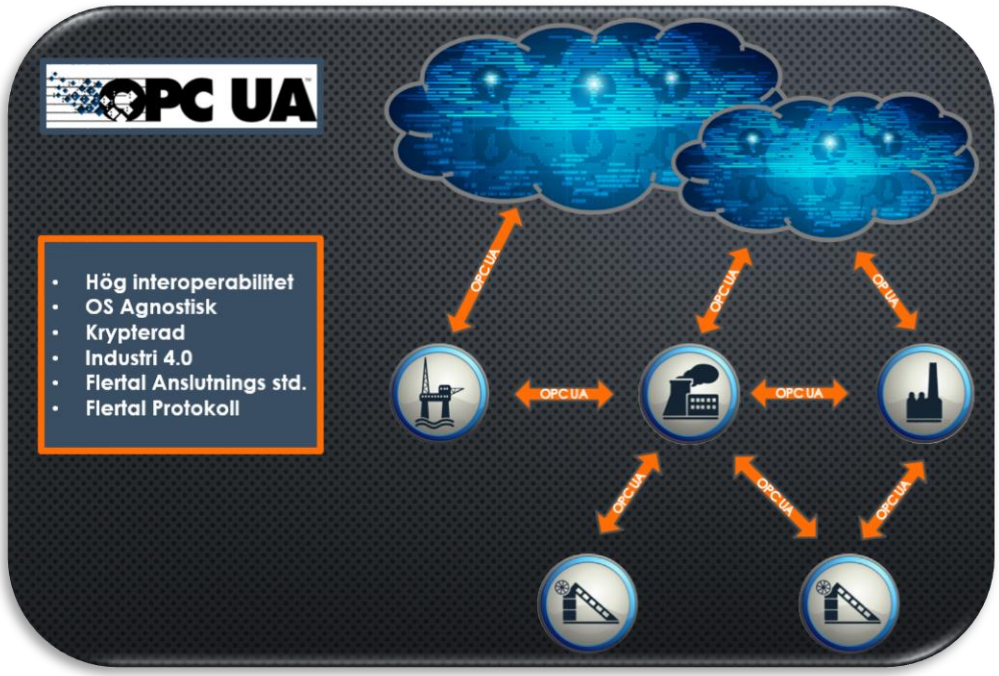
OPC UA-applikationen använder sig av en OPC UA-stack och en OPC UA SDK för att få den specifika applikationsfunktionaliteten och funktionalitetsmappningen till OPC UA. En server SDK eller OPC UA-klient genomför OPC UA-funktioner i applikationsskiktets del och UA-stacken kör enbart kommunikationskanalerna. Utvecklingsinsatsen minskar med hjälp av SDK och bidrar till en snabbare interoperabilitet[17] för en OPC UA-applikation [29].

[16] **software stack** eller **solution stack** – **mjukvarustack, mjukvarustapel***:* en komplett uppsättning program som tillsammans behövs för att utföra ett arbete. En stack innehåller all mjukvara som behövs, från operativsystemet till den applikation som användaren använder för att utföra sin arbetsuppgift.

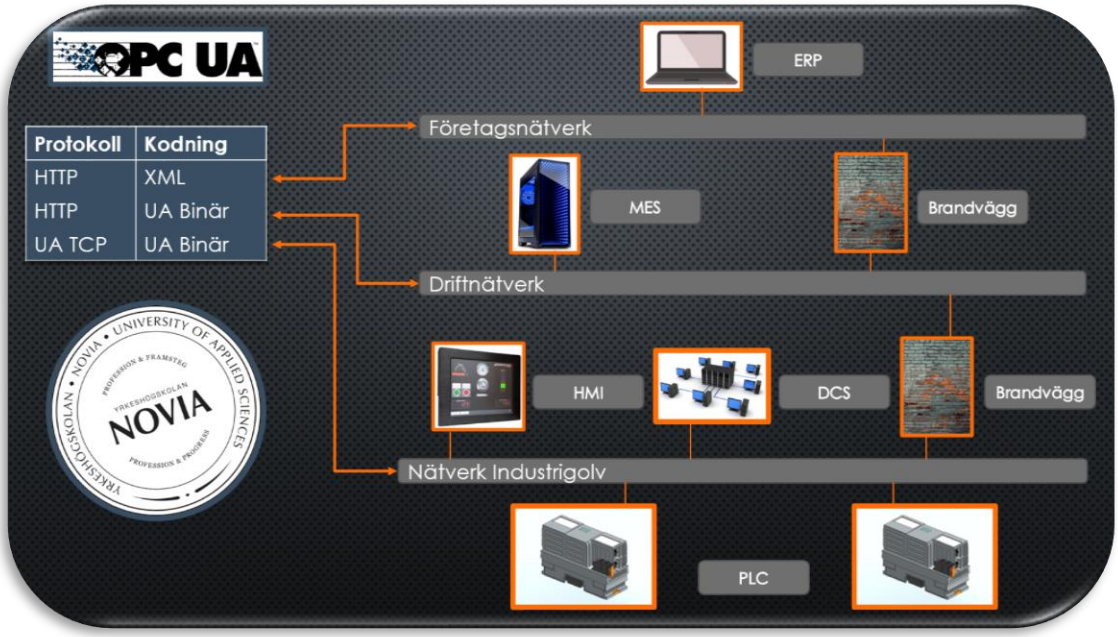
[17] **Interoperabilitet -** förmåga hos olika system att fungera tillsammans och kunna kommunicera med varandra

5 Funktionsöversikt OPC UA

Detta kapitel behandlar en grafisk översikt kring hela OPC UA-strukturen. Figur 18 visar en förenkling av den yttre nätverksstrukturen.

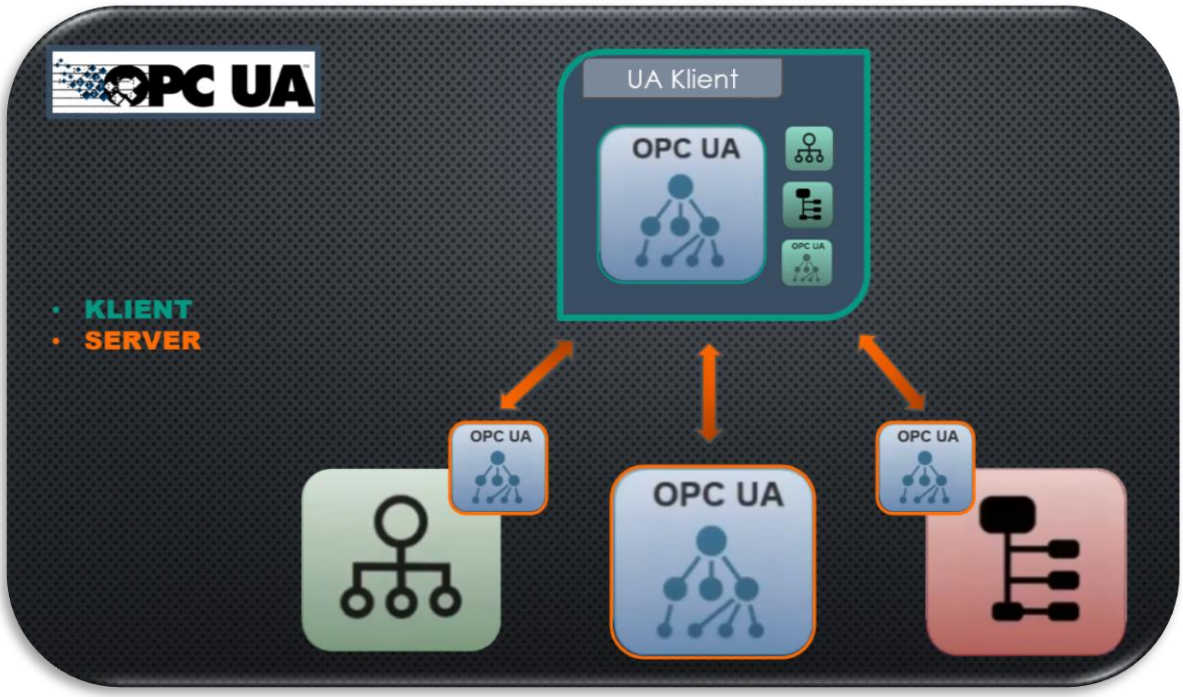


Figur 18 OPC UA översiktnätverk



Figur 19 OPC UA nätverkslager, transportprotokoll och kodningar

Figur 19 visar de olika användningsområden för samtliga protokoll och kodningar i varje nätverkslager samt vilka enheter som körs på varje individuellt nätverk.



Figur 20 OPC interoperalitet enheter

Interoperalitet har i första hand att göra med data - förmågan att skapa informationsmodeller eller modellstrukturer efter olika produkter.

Den nedre delen av figur 20 representerar enheter med individuella typer av datastrukturer som är länkad till en OPC UA-aktiverad klient. Genom att ha en OPC UA-server på varje individuell enhet, möjliggör det att skapa en modell av informationen med relationen av den data som enheten använder inhemskt och sedan uttrycka den relationen med OPC UA-metoden av att modellera information [30].

Fördelarna med detta är att den bevarar och håller intakt det som krävs för den ursprungliga informationsmodellen, då enheterna kan vara designade till en specifik industri (ex. inom automation används BACnet[18]) [30].

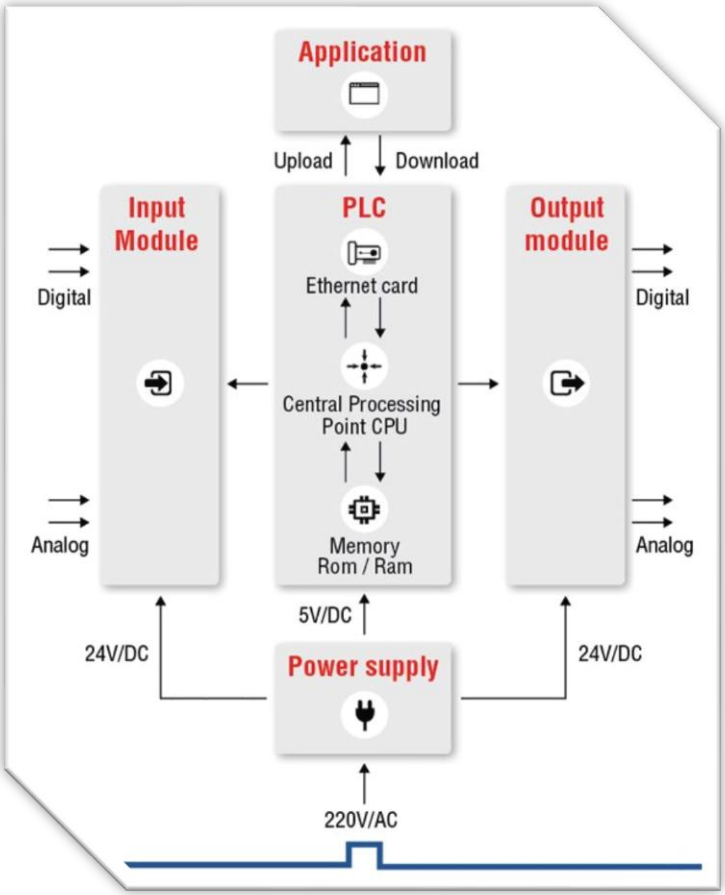
Dessutom blir OPC UA-kommunikationen helt fristående av enheternas växlande protokoll komplexitet och ger frihet till ett tredjepartssystem (UA klient) att förstå eller identifiera relationerna mellan de olika informationsmodellerna [30].

[18] **BACnet** - ett standardprotokoll för kommunikation inom fastighetsautomation

6 PLC

En PLC är ett programmerbart styrsystem likt en dator som används främst till att styra maskiner och processer inom industrin. Därav delar PLC:n en del terminologi med den vanliga datorn som processorenhet, minne, mjukvara och kommunikation. PLC:n är designad för en extrem hård industriell miljö med en hög driftsäkerhet och flexibel när det kommer till samverkan mellan in- och utgångar anknutna till den verkliga världen [31].

PLC är en dedikerad styrenhet som oftast kör endast ett program kontinuerligt. Skanningstid är vad en cykel genom programmet kallas och detta innebär att ingångarna läses av från de andra modulerna och sedan exekvera den logik som baserats på de ingångarna samt uppdatera utgångarna. Processorminnet lagrar programmet och håller den aktuella statusen för in- och utgångarna [31].



Figur 21 PLC – Funktionsöversikt (Källa: Unitronics)

Beroende på in- och utgångar så kan en PLC övervaka och spara drifttidsdata för maskinproduktivitet och drifttemperaturer. Automatiskt starta och stoppa processer samt

generera olika typer av alarm för tekniska fel och processavbrott. En PLC är en flexibel och stabil kontrollösning som kan anpassas till de flesta applikationer. Det som skiljer PLC:n åt från mikrokontrollers, industriella datorer och andra kontrollösningar utöver lagringen i processorminnet, är att användaren kan blanda och matcha en PLC:s in- och utgångar för att få rätt konfiguration för sin applikation. Under kommunikationsdelen kan en PLC behöva ansluta till andra typer av system för exempelvis en systemövervakningskontroll med datainsamling (SCADA) som övervakar ett flertal system samtidigt. En PLC erbjuder en serie portar och kommunikationsprotokoll för att säkerhetsställa kommunikationen med dessa andra system. Interaktionen med en PLC i realtid görs via en HMI, dessa operatörsgränssnitt kan vara allt ifrån enkla skärmar med en textavläsning och knappsstats till en mer konsumentlik digitalmiljö. Samtliga versioner gör det möjligt för användaren att granska och mata in information till PLC:n i realtid [32].

6.1 Signaler

En signal används till att transportera data från ett system eller nätverk till ett annat och är en elektromagnetisk eller elektrisk ström. Inom elektronik och telekommunikation refererar signalen till en tidsvarierande spänning som är en elektromagnetisk informationsbärande våg. En signal kan även definieras som en observerbar förändring i kvalitet liksom kvantitet [33].

6.1.1 Analog signal

En analog signal är en kontinuerlig signal med en tidsvarierande kvantitet som representerar en annan tidsbaserad variabel. Denna typ av signaler fungerar med fysiska värden och naturfenomen som jordbävning, vindhastighet, blixt etcetera. Den analoga signalen betecknas med en **sinusvåg** och fungerar med kontinuerliga data och kan vara antingen periodisk eller icke periodisk, samt med en lägre noggrannhet i jämförelse med den digitala signalen [33].

6.1.2 Digital signal

En digital signal är en signal som används för representation av data som en sekvens av separata värden vid vilken tidpunkt som helst. Digitala signaler betecknas med **fyrkantsvågor** och har en hög noggrannhet som kan överföras och behandlas bättre jämfört med analoga signaler [33].

7 Klient – Server definition

Klient – server representerar ett förhållande mellan samarbetsprogram i en applikation. Vissa maskiner är avsedda som servrar för att betjäna behoven hos klientparten. En server kan vara en stordator, PLC eller en persondator. Klienter är fristående datorer som förlitar sig på servrar för att hantera de primära uppgifterna, nätverksadministration, säkerhet och andra kritiska funktioner [34].

7.1 Klient – Server modell

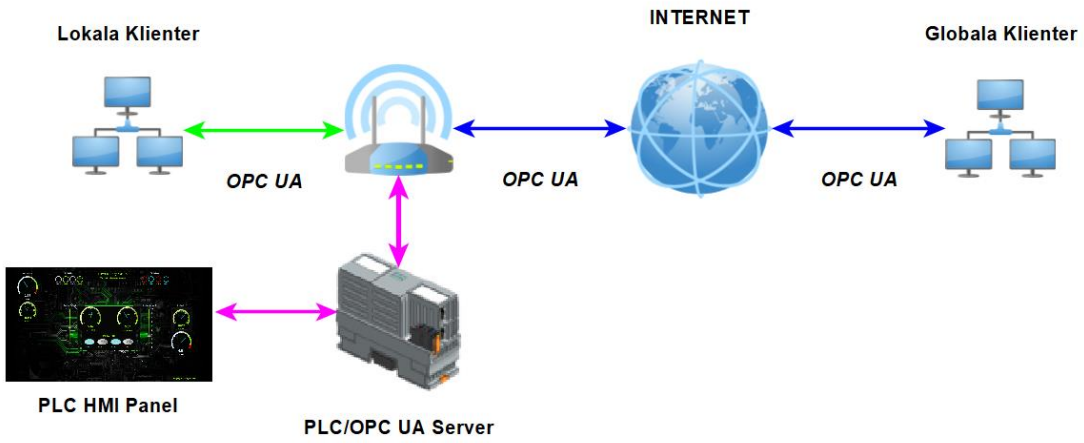
Klient – server arkitekturen är ett applikationsramverk som distribuerar uppgifter mellan klienter och servrar som kan finnas i samma system eller att kommunikationen sker via ett datornätverk eller internet. Klienten skickar en begäran till ett program för att få tillträde till en tjänst som tillgängliggörs av servern [35].

Förhållandet mellan server och klient är att de kommunicerar i ett meddelandemönster i form av handskakning[19] och följer ett gemensamt kommunikationsprotokoll, som formellt definierar de regler, språk och kommunikationsmönster som ska användas. Klientförfrågningar organiseras och prioriteras inom ett schemaläggningssystem, detta hjälper servrar med att ta emot förfrågningar från olika klienter på kort tid [35].

[19] **Handskakning** en procedur med vilken elektroniska apparater etablerar kommunikation.

8 Projektet i sin helhet

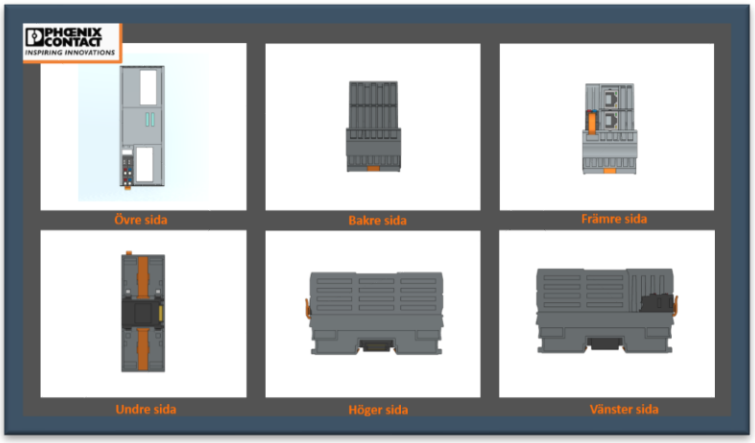
Teorin i detta examensarbete är avgränsat till fokus på OPC UA-kommunikationen med dess funktions principer och en grundförståelse om PLC och signaler. Projektet i sig är mycket mer omfattande och kräver konfigurering av PLC-enhet, nätverk, OPC UA server och OPC UA klient samt skapande av program och HMI, för att kunna etablera en OPC UA kommunikation med funktioner.



Figur 22 Projektets kommunikations översikt

8.1 Hårdvara

Hårdvara som använts för detta projekt är en Phoenix Contact PLC AXC F 2152 med tillhörande moduler för digitala (AXL F DI16/1 DO16/1 2H) och analoga (AXL F AI2 AO2 1H) in- och utgångar.



Figur 23 Phoenix PLC 2152 översikt

8.2 Mjukvara

Mjukvaran som använts för detta projekt är PLCnext Engineer och UaExpert klient från Unified Automation.

PLCnext Engineer är programmet som använts till att skapa programvara och konfigurera PLC portar för in- och utkommande signaler. PLC kontrollern har en inbyggd OPC UA server som konfigureras via denna mjukvara. En HMI dashboard[20] för kontrollern skapas även via PLCnext Engineer och används till för övervakning och interaktion med programmet.

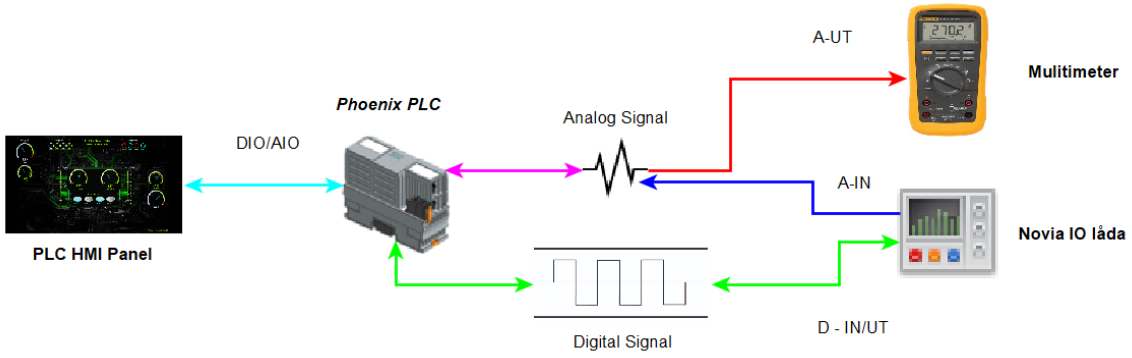
UaExpert är en fullständig OPC UA testklient som kan använda sig av ett flertal OPC UA profiler och funktioner. UaExpert användes till att testa det skapade prototypprogrammet för OPC UA-kommunikation med funktioner som alarm och kryptering.

[20] **Dashboard** – en visuell och lättfattlig display som innehåller de nyckeltal som är viktiga för den person som läser av dem.

8.3 Planering och utförande

Projektet separerades in i mindre delmål för att göra det mer lätt hanterligt. Den första delen var att samla all teoretiskbakgrund gällande OPC UA-kommunikationsprotokoll. Det gav en förståelse kring dess uppbyggnad och funktionalitet.

Den andra delen var att konfigurera PLC-kontrollern och den inbyggda OPC UA-servern. Ett testprogram skapades i PLCnext Engineer för att testa in och utgående signaler av analog och digital typ.

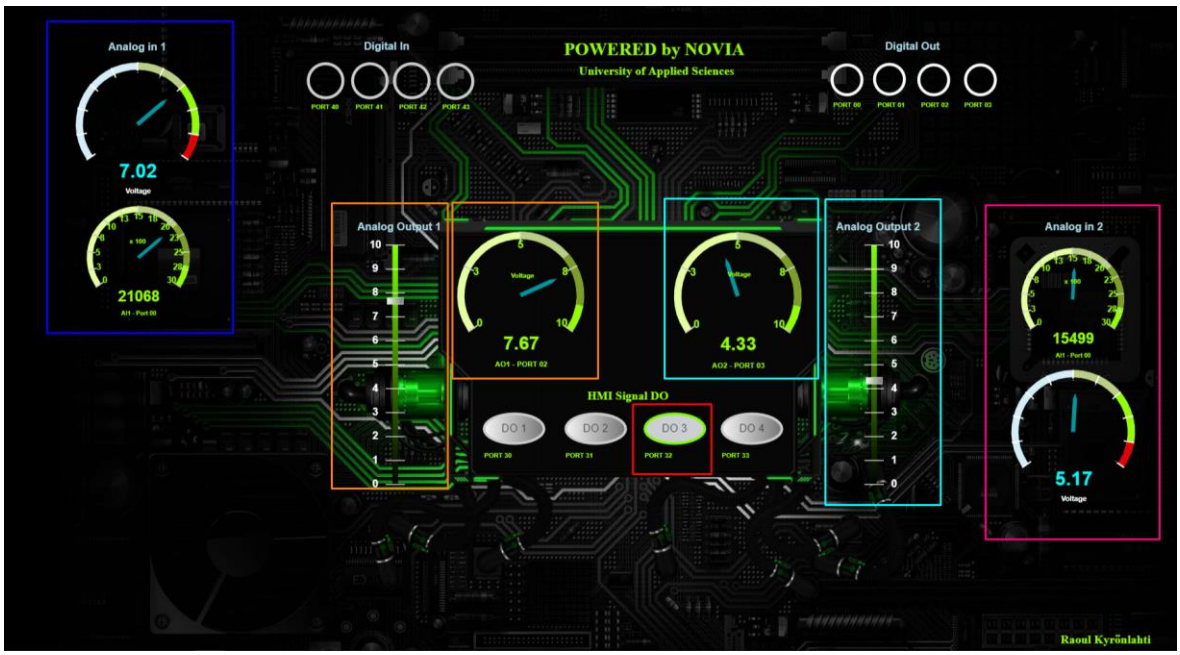


Figur 24 Utförande av signaltester

Skapandet av HMI för PLC kontrollern gjordes även under detta skede. Användningen av HMI:n underlättade testet för signalprocessen och gav en grafisk visualisering som sedan kunde jämföras med Multimetern eller Novias IO låda.

Efter att alla signaler fungerade korrekt, etablerades det en kommunikation till UaExpert-testklient. Där de signaler som var aktiverade för OPC UA-kommunikation i PLC-servern kunde läsas av med klienten. Se kap. 5.1.1 del 4, 6, 7 och 5.2.3

Under detta stadie testades det även att skapa en egen prototypklient med programmet Node-RED. Det är ett användarvänligt program där man kan läsa/skriva data och skapa en HMI på klientsidan utan att behöva fördjupad programmeringskunskap.



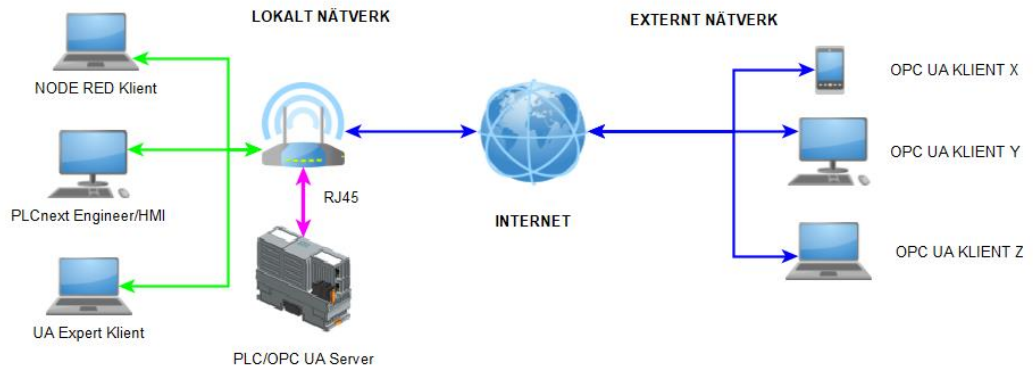
Figur 25 Phoenix PLC HMI

Data Access View

#	Server	Node Id	Display Name	Value	Datatype	Source Timestamp	Server Timestamp	Statuscode
1	eUAServer@192...	NS5|String|Arp....	Analog_out01	23000	UInt16	12:56:42.958	12:56:42.995	Good
2	eUAServer@192...	NS5|String|Arp....	Analog_out01_Scaled	7.66667	Float	12:56:42.958	12:56:42.995	Good
3	eUAServer@192...	NS5|String|Arp....	Analog_out02	13000	UInt16	12:56:38.208	12:56:38.247	Good
4	eUAServer@192...	NS5|String|Arp....	Analog_out02_Scaled	4.33333	Float	12:56:38.208	12:56:38.247	Good
5	eUAServer@192...	NS5|String|Arp....	DO_1	false	Boolean	12:40:21.111	12:40:21.111	Good
6	eUAServer@192...	NS5|String|Arp....	DO_2	false	Boolean	12:40:22.839	12:40:22.839	Good
7	eUAServer@192...	NS5|String|Arp....	DO_3	true	Boolean	12:56:40.459	12:56:40.495	Good
8	eUAServer@192...	NS5|String|Arp....	DO_4	false	Boolean	12:40:25.418	12:40:25.418	Good
9	eUAServer@192...	NS5|String|Arp....	Analog_in	21067	Int32	13:02:21.459	13:02:21.497	Good
10	eUAServer@192...	NS5|String|Arp....	Analog_in_Scaled	7.02367	Float	13:02:21.459	13:02:21.498	Good
11	eUAServer@192...	NS5|String|Arp....	Analog_in2	15495	Int32	13:02:21.459	13:02:21.497	Good
12	eUAServer@192...	NS5|String|Arp....	Analog_in_Scaled2	5.16267	Float	13:02:21.459	13:02:21.498	Good

Figur 26 UaExpert klient

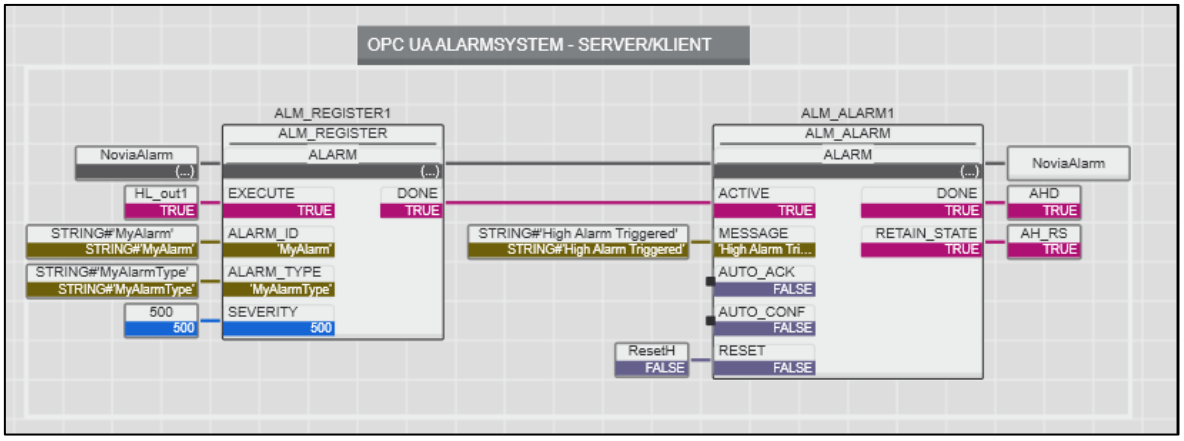
Den tredje delen var det fokus på nätverk och OPC UA alarm funktioner. I nätverksdelen konfigurerades det både lokala och globala nätverk för OPC UA kommunikation mellan klient och server, se kap. 5.1.3 del 12 och 5.2.3



Figur 27 Konfigurering av nätverk

För den globala kommunikationen mellan server och klient aktiverades krypteringen via test klienten UaExpert, samtliga former av kryptering testades. Se kap 5.2

Implementeringen av alarmsystemet (se kap. 5.1.2 del 9, 10) gjordes via färdiga alarmblock som fanns tillgängliga i ett utbyggnadsbibliotek som installerades till PLCnext Engineer som sedan kopplades samman med det manuella alarmsystemet som skapats i testprogrammet för PLC:n.



Figur 28 Registrering av OPC UA alarmsystem – PLCnext Engineer

Den fjärde delen i projektet var att ta den teoretiska och praktiska kunskapen från de tidigare delarna och skapa en grund för ett läromaterial och en laboration för utbildningssyfte till Yrkeshögskolan Novia. Målet med laborationen var att sammanlänka kommunikationsstandarden för industri 4.0 med ett simulerat verklighetsscenario. Detta ska ge ett intryck till en så nära verklig ingenjörsutmaning som ska kunna relateras till utmaningar för ingenjörsarbeten inom automationssektorn. Se bilaga 1.

9 Resultat

Projektet resulterade i ett grundläggande ramverk för läromedel kring industriell kommunikation med OPC UA. Följande dokument skapades som referensmaterial för konfigurering av PLC, HMI, nätverk, server, klient samt programmering av OPC UA-funktioner för industriell kommunikation.

- Konfigurering av Phoenix PLC & Design av intern HMI – Fas 1
- Alarmsystem & Blockprogrammering – Fas 2
- Laboration 1 – Industriell kommunikation

Det första dokumentet (Fas 1) har all grundläggande information som behövs, från att skapa ett eget automationsprogram med analoga och digitala signaler med tillhörande HMI styrning, till att etablera en krypterad global kommunikation mellan OPC UA-server och klient, samt tillvägagångssätt för testning av prototypprogram och kommunikation.

Det andra dokumentet (Fas 2) har fokus på funktionerna gällande OPC UA protokollet, specifikt alarmfunktionerna. Detta dokument var menat att innehålla mer men var tvunget att kapas på grund av projektets omfattning. Det som inte tagits med är en mer fördjupning av alarmsystemsfunktioner, event typer och historisk datatillgång.

Laborationen skapades och baserades på grundkunskapen från dokumenten Konfigurering av Phoenix PLC & Design av intern HMI – Fas 1 och Alarmsystem & Blockprogrammering – Fas 2, med en utgångspunkt kring frågor ur ett pedagogiskt perspektiv.

- Hur kan man göra laborationen mer intressant?
- Hur kan en laboration med endast tre signaler ge ett intryck av hög betydelsefullhet?
- Hur kan denna laboration länkas till verkliga situationer och processer?
- Hur kan laborationen anpassas till elever på olika kunskapsnivåer?
- Hur kan lektorn dynamiskt justera materialet efter behov?
- Hur kan man trigga elevens kritiska tänkande, ansvarstagande och bättra på teamegenskaperna?

Kombinationen med tekniskgrundkunskap och aspekter ur ett pedagogiskt tänkande resulterade till en högkvalitativ laboration med utbyggnads och anpassningsmöjligheter, som skiljer sig något från den abstrakta norm som används idag inom Yrkeshögskolan Novia.

Grunden för läromaterialet är av en modulär uppbyggnad vilket gör den flexibel och framtidssäker. Flexibel i den form att delar av materialet går att använda i kurser utöver industriell kommunikation, som till exempel Industriella automationssytem och styrsystem, genom detta stärka den röda tråd som binder automationsteknikens kurser samman. Framtidssäkert, då materialets utgångspunkt är från industri 4.0 och bjuder in för vidareutveckling och påbyggnation från varje elevkull under de kommande åren.

10 Diskussion

Projektet har varit en riktigt utmanade process med många delmoment att ta i beaktande. Jag har alltid intresserat mig för nya teknologier och när det uppstod en möjlighet för ett projekt med OPC UA-kommunikation vart det en självklarhet att ta an utmaningen som ett examensarbete.

Till en början var projektomfattningen endast inriktat på OPC UA-kommunikationen och dess funktionalitet mellan server och klient. Detta ändrades fort då det inte fanns någon färdigkonfigurerad hårdvara till bruk och projektomfattningen var tvungen att expanderas med PLC-konfigurering och programmering, klientkonfigurering samt konfigurering av nätverksdel för global kommunikation.

Det mest utmanande med projektet var att det fanns väldigt begränsat med material och källor gällande information både för OPC UA och PLC-kontrollern från Phoenix Contact. OPC UA för att det är relativt nytt och utvecklas kontinuerligt. Siemens har den större delen av den industriella marknaden när det kommer till PLC-enheter och majoriteten av information för PLC är om just olika typer av Siemens enheter. Den information det fanns om Phoenix Contacts produkter var lite mer riktad mot de som har tidigare erfarenhet med PLC-enheter. Detta blev ett projekt i sig att lära sig programmera och konfigurera en Phoenix PLC från grunden med minimal kunskap och inga referenser att vända sig till.

Jag är oerhört tacksam om att fått vara med och sätta samman ett projekt av denna magnitud med endast några få riktlinjer tillhands. Detta gav en utmaning för mitt kreativa tänkande och testade mitt tekniska kunnande som jag lärt mig under åren på Yrkeshögskolan Novia. Resan under projektets gång har varit otroligt lärorikt och givande även då det funnits stunder som känts att man tagit lite vatten över huvudet när projektet expanderats eller då man kört fast ordentligt. Jag är väldigt nöjd över projektets slutresultat, men nådde inte riktigt dit jag satt mina personliga mål. Det som skulle tagits med är ett mer fördjupande av de olika funktioner OPC UA erbjuder. Men detta öppnar upp möjligheter för någon annan att bygga vidare på projektet om man känner sig vilja ta an utmaningen.

En tanke jag fick under planeringsstadiet var att projektet skulle kunna användas utöver den kurs den var skapad för. En möjlighet att kunna bygga gemensamma projekt som sträcker sig över alla de inriktningarna el- och automationsteknik erbjuder, om Yrkesskolan Novia skulle känna att det finns ett behov för det.

Exempelvis de som valt it som inriktning skulle kunna ha ett gemensamt projekt med de som valt automation där it bygger en OPC UA-klient från grunden med en tillhörande databas med de olika utvecklarverktyg som finns tillgängligt, som sedan länkas samman med ett automationsprojekt. Kraft inriktningen kan även vara med att delta, genom att koppla en utomstående OPC UA-server till något av deras projekt och sedan etablera en kommunikation mellan it och automation.

Fördelarna med en OPC UA-standard är att den är både hårdvaru- och plattformsoberoende vilket leder till en enorm flexibilitet, Skolan behöver inte låsa sig till något specifikt märke av produkter eller införskaffa dyra licenser för mindre projekt då det finns en del open source produkter redan på marknaden.

Programmet PLCnext Engineer från Phoenix Contact som använts till detta projekt för att konfigurera PLC:n och OPC UA-servern är gratis att ladda ner och kräver inga licenser, den är dock begränsad till Phoenix Contacts produkter, men kan installeras på vilken Windows 10 enhet som helst, som inte har en version äldre än 1709. Detta gör det enkelt för studerande och andra med begränsade resurser att kunna lära sig och experimentera med olika lösningar på egen hand, tillskillnad från Siemens TIA portal som är licensbaserad och endast kan installeras på Windows 10 Enterprise.

Jag kan se att skolor och mindre automationsföretag skulle kunna dra stor nytta av den typen av program, då den är ekonomisk och inte är begränsad till en specifik typ av miljö och kriterier. Även under en pandemi skulle forskning och utveckling kunna fortsätta i samma takt, med ett minimalt krav på hårdvara.

11 Referenser

[1] M. Skilton och F. Hovsepian, The 4th Industrial Revolution: Responding to the impact of artificial intelligence on business, Switzerland: Palmgrave Mcmillian, 2018, p. 322.

[2] E. G. Popkova, Y. V. Ragulina och A. V.Bogoviz, Industry 4.0: Industrial Revolution of the 21st Centurty, 1:a red., Springer, 2018, p. 249.

[3] ”Siemens,” 1996 - 2020. [Online]. Available: https://new.siemens.com/global/en/company/about/history/people/founders.html.

[4] ”OPC UA Online Reference,” 19 Decmber 2019. [Online]. Available: https://reference.opcfoundation.org/v104/GDS/docs/8.1/.

[5] P. Seeberg, ”OPC UA as a Bridge between IT and Automation,” Softing Industrial Automation GmbH.

[6] OPC Foundation, ”OPC Unified Architechture: Pioneer of the 4th industrial (r)evolution”.

[7] ”Novotek,” 2020. [Online]. Available: https://www.novotek.com/sv/l-sningar/kepware-opc-kommunikationsplattform/opc-och-opc-ua-en-foerklaring/.

[8] W. Mahnke, S.-H. Leitner and M. Damm, OPC Unified Architecture, Ladenburg: Springer, 2009, p. 339.

[9] ”OPC Foundation,” 15 Jun 2015. [Online]. Available: https://opcfoundation.org/.

[10] U. A. GmbH, ”Unified Automation,” [Online]. Available: https://documentation.unified-automation.com/uasdkhp/1.0.0/html/_l2_opc_ua_specifications.html.

[11] ”OPC Foundation,” 22 Nov 2017. [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-6-mappings/.

[12] ”OPC foundation,” 22 Nov 2017. [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-7-profiles/.

[13] ”OPC Foundation,” 22 Nov 2017. [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-8-data-access/.

[14] ”OPC Foundation,” 22 Nov 2017. [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-9-alarms-and-conditions/.

[15] ”OPC Foundation,” 22 Nov 2017. [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-10-programs/.

[16] ”OPC Foundation,” 09 Jan 2018. [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-11-historical-access/.

[17] ”OPC Foundation,” 07 Feb 2018. [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-12-discovery-and-global-services/.

[18] Foundation, OPC, ”Part 13: Aggregates,” 2015.

[19] OPC Foundation, 06 februari 2018. [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-14-pubsub/.

[20] OPC Foundation, 19 April 2019. [Online]. Available: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-100-device-information-model/.

[21] ”National Instruments,” 14 Mars 2019. [Online]. Available: https://www.ni.com/fi-fi/innovations/white-papers/12/why-opc-ua-matters.html.

[22] ”Ascolab,” Automation Systems Communication Laboratory, [Online]. Available: http://www.ascolab.com/en/technology-unified-architecture/technology-security.html.

[23] SSL Authority INC, ”SSL Authority - Gobal Encryption Leader,” 2017. [Online]. Available: https://www.sslauthority.com/x509-what-you-should-know/.

[24] OPC Foundation, 2020. [Online]. Available: https://opcfoundation.org/developer-tools/.

[25] Prosys OPC LTD, 2020. [Online]. Available: https://www.prosysopc.com/products/.

[26] Prosys OPC LTD, ”prosysopc,” 2020. [Online]. Available: https://www.prosysopc.com/products/opc-ua-cplusplus-sdk/.

[27] Prosys OPC, 2020. [Online]. Available: https://www.prosysopc.com/products/opc-ua-dotnet-sdk/.

[28] Prosys OPC, 2020. [Online]. Available: https://www.prosysopc.com/products/opc-ua-modeler/.

[29] U. A. GmbH, ”Unified Automation,” [Online]. Available: https://documentation.unified-automation.com/uasdkhp/1.0.0/html/_l2_opc_ua_software_layers.html.

[30] Matrikon OPC, ”Youtube/Matrikon Videos,” 9 April 2018. [Online]. Available: https://www.youtube.com/watch?v=PV_T-CGtDII&t=2702s.

[31] PLCdev - Tools for PLC programming, ”plcdev,” 2005 - 2020. [Online]. Available: http://www.plcdev.com/how_plcs_work.

[32] UNITRONICS, ”unitronicsplc,” 2020. [Online]. Available: https://unitronicsplc.com/what-is-plc-programmable-logic-controller/.

[33] Guru99, ”guru99.com,” 2020. [Online]. Available: https://www.guru99.com/analog-vs-digital.html#2.

[34] Your Dictionary, ”yourdictionary.com,” 1996-2020. [Online]. Available: https://www.yourdictionary.com/client-server#computer.

[35] OMNI - SCI, ”omnisci.com,” 2020. [Online]. Available: https://www.omnisci.com/technical-glossary/client-server.

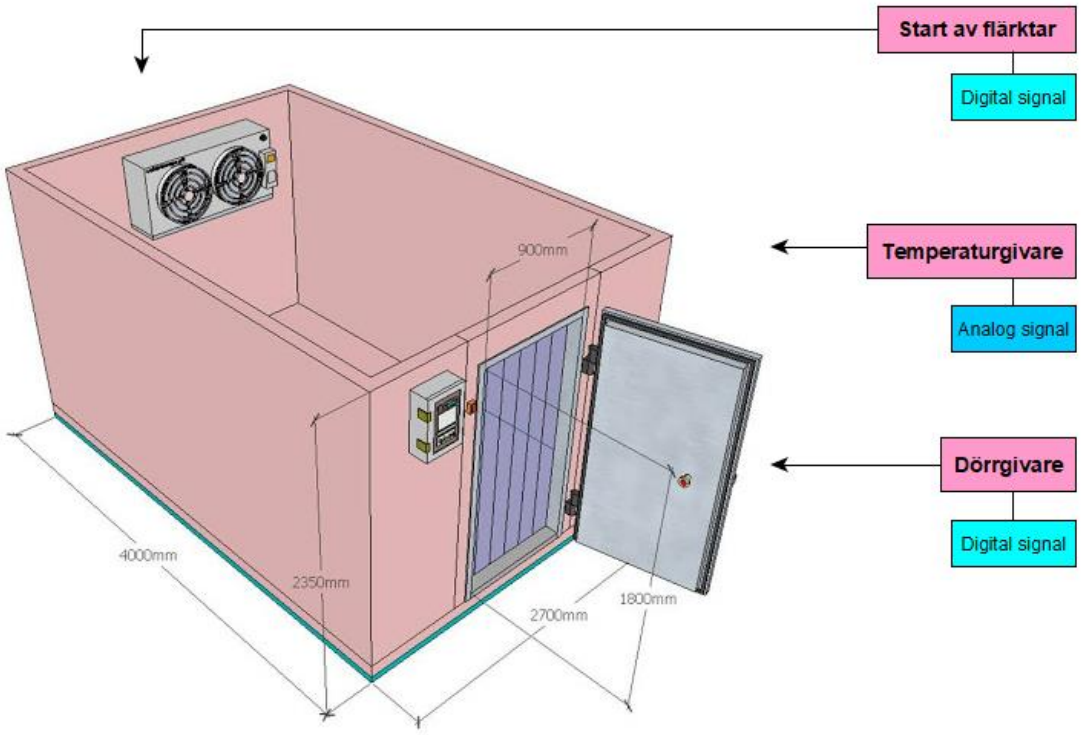
Bilaga 1 laboration

Laboration 1 – Kylrumsfunktioner

Corona pandemin har börjat lägga sig och det som återstår är att börja återuppbygga den trasiga samhällsstrukturen tillbaka till dess storhetstid. Diskussionsämnen som export, import, global ekonomi, arbetsmarknad samt utbildningar tar plats vid G20 mötet där alla världsledare möts för att hitta lösningar. Till mötet kommer det att flygas in exklusiva matråvaror från alla världens håll. Varje land som deltar kommer att skicka en grupp av experter som har hand om funktionaliteten för G20 mötet.

Finland är globalt kända för sina högkvalitativa ingenjörer och fick i uppdrag att se över förvaringen av kaviar som flygs in från Ryssland, Iran och Azerbajdzjan.

GRATULERAR! Du kvalificerade för ingenjörsteamet som representant för Finland! Nu är det upp till dig att testa dina färdigheter.



Figur 1 – Special kyl X1

Då kaviar måste förvaras i en temperatur mellan $-2\ och\ +2℃$ eller när det kommer till specifikt exklusiva sorter vars förvaringstemperatur ska ligga mellan $28-30$ F, finns det varken en kyl eller frys som klarar detta, en special kyl måste tillverkas.

Det svenska teamet står för konstruktionen av själva kylrummet och det finländska teamet ansvarar för automatiseringen och övervakningen.

Kylrummets position kommer att vara cirka 5 min från anläggningen där G20 mötet hålls och kaviaren tas ut 10 min innan servering.

Tabell 1 Kravspecifikation X1

Kravspecifikation X1	
Styrenhet	**PLC**
Övervakning	**HMI**
Reglering	**Manuell** & **HMI**
Kylfläkt	**Digital Signal**
Dörrgivare	**Digital Signal**
Temperaturgivare	**Analog Signal**
Alarm	**HMI & OPC UA Klient**
Protokoll	**OPC UA Kommunikation**

Fas 1 - HMI & Konfiguration av PLC

Konfigurera de PLC-portar som behövs och skapa en intern HMI i **PLCnext Engineer**, testa sedan att alla signaler fungerar korrekt, både för hårdvara och HMI. Hårdvarutestet kan utföras med hjälp av Novias IO-låda.

Ta hjälp av dokumentet **Konfigurering av Phoenix PLC fas 1**

Fas 2 – Program för Kylsystem

Skapa programmet för kylsystemet X1 i **PLCnext Engineer**. I programgrunden bör fokus endast vara på de **tre** signaler och givare som figur 1 illustrerar samt länkningen till **HMI** från dessa. Några saker att ta i beaktande.

- Ska kylfläkten vara aktiv under upphämtning av råvaror?
- Om kyldörren glömts öppen?
- Manuell styrning av kylfläkt, **HMI**, **Hårdvara**?
- Säkerhetsaspekter krig manuellstyrning, **Av**, **PÅ**, **AUTO**

Ta hjälp av dokumentet **Konfigurering av Phoenix PLC fas 1**

Fas 2.1 – Övervakning via Klient

Då du bär huvudansvaret för övervakningen och sköter det via en HMI, så bör en sekundär övervakning etableras både vid Kylrumsanläggningen och G20 anläggningen för de övriga internationella funktionärerna som arbetar inom sektorn.

Tabell 2 Kravspecifikation Övervakning

Kravspecifikationstillägg - Övervakning	
OPC UA Klient 1 - Primär	**READ**
OPC UA Klient 2 - Tillägg	**READ & Write**

1. Ladda ner och installera testklient från **Unified Automation**
2. Aktivera **OPC UA Server** i Phoenix PLC
3. Etablera en säker OPC UA kommunikation mellan **server** och **klient**.
4. Välj ut de signaler **du** anser är lämpliga att övervaka via klient
5. Välj rättigheter (Read, Write) för klient om möjligt

Ta hjälp av dokumentet **Konfigurering av Phoenix PLC fas 1**

Fas 3 – Alarm

Då anläggningarna ligger 5 min ifrån varandra att det viktigt att alarm och felmeddelande uppstår vid avvikelse och i olika grader av tillstånd.

Ta hjälp av dokumentet **Alarmsystem & Blockprogrammering fas 2**

Tabell 3 Kravspecifikation Alarm

Kravspecifikation Alarm	
Låg-larm/Felmeddelande	Öppen dörr <x min
Hög-larm/Felmeddelande	Öppen dörr >x min
Ställbara Alarm Låg	Ställbara värden av temperaturskiftning
Ställbara Alarm Hög	Ställbara värden av kritiska temperaturer

1. Konfigurera ett **Alarmsystem** som uppfyller kraven från tabell 3
2. Länka alarmen till din **HMI**
3. Länka alarmen till **OPC UA klienten**

Fas 4 – Egna funktioner

I det finländska teamet märkte vi av i ett tidigt skede att den ursprungliga kravspecifikationen har sina brister gällande optimal funktionalitet. Varav en gäller funktionen för kylfläkten. Det skulle vara mer optimalt att kunna reglera hastigheten på fläkten efter behov.

Figur 2 Konceptfläkt

1. Byt ut **fläktens** digitala signal mot en analog, hitta en lämplig lösning för styrning och reglering av fläkt, exempelvis genom **PID** reglering eller någon annan funktion du anser vara lämplig för optimering.

I Finland gillar vi att tänka utanför ramarna och ligga i framkant när det kommer till utvecklade, det grundar sig i vårt **SISU** som representerar styrka, envishet och uthållighet.

1. Hitta **brister** på X1 som du själv skulle vilja förbättra.
2. Förverkliga **lösningar** på dessa brister