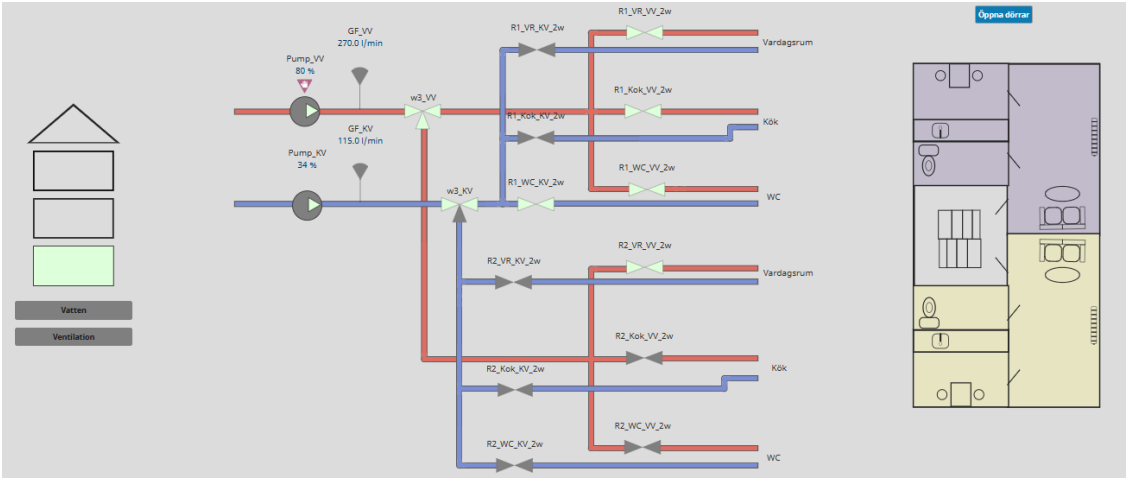
CHALMERS



Utveckling av ett standardbibliotek för fastighetsprojekt i Ignition SCADA

Examensarbete inom högskoleingenjörsprogrammet Mekatronik

FRIDA OLSSON
ALEXANDER STABERG-SAND

INSTITUTIONEN FÖR ELEKTROTEKNIK
CHALMERS TEKNISKA HÖGSKOLA
Göteborg 2025
www.chalmers.se

Examensarbete 2025

Utveckling av ett standardbibliotek för fastighetsprojekt i Ignition SCADA

FRIDA OLSSON
ALEXANDER STABERG-SAND

CHALMERS

Institutionen för Elektroteknik
Chalmers Tekniska Högskola
Göteborg 2025

Utveckling av ett standardbibliotek för
fastighetsprojekt i Ignition SCADA

Handledare: Joel Olsson, Init Sweden AB
Examinator: Veronica Olesen, Elektroteknik

Examensarbete 2025
Institutionen för Elektroteknik
Chalmers Tekniska Högskola
SE-412 96 Göteborg
Telefon +46 31 772 1000

Omslagsbild: En bild på SCADA-systemet som är utvecklat som Proof of Concept i projektet.

Skriven i LaTeX
Printed by Chalmers Reproservice
Göteborg 2025

Förord

Detta examensarbete är utfört av två mekatronikingenjörstudenter från Chalmers Tekniska Högskola under institutionen för elektroteknik. Arbetet utfördes på företaget Init Sweden AB under våren 2025. Arbetet omfattade 15 högskolepoäng per stundent och innehåller kunskaper inom ämnena mekatronik, programmering och styr- och övervakningssystem.

Vi vill först och främst tacka vår handledare på Init, Joel Olsson, för det kontinuerliga stöd vi fått under projektets gång. Vi vill även tacka de övriga anställda på företaget som varit tillgängliga för frågor och gett feedback på projektet. Till sist vill vi också tacka vår examinator och handledare från institutionen, Veronica Olesen som hjälpt oss med utformingen av denna rapport.

Frida Olsson och Alexander Staberg-Sand, Göteborg, Maj 2025

Sammanfattning

SCADA-systemet Ignition SCADA har på senare år blivit allt mer vanlig inom Europa tack vare dess flexibilitet och webbaserade plattform. Med det i åtanke har Init Sweden AB valt att initiera ett examensarbete som ska utveckla ett standardbibliotek för att underlätta utvecklingen av framtida system baserat på Ignition SCADA som platform. Init Sweden AB har även sett ett behov att utveckla ett enklare SCADA-system för att kunna använda som demonstrator för att undervisa i vad SCADA-system är.

Biblioteket är utformat efter Init Swedens AB interna standarder för SCADA system och omfattar de 11 vanligaste processobjekten företaget använder inom fastighetsautomation.

Rapporten behandlar även utvecklingen av ett Proof of Concept som är tänkt att agera som en demonstrator åt företaget för det nya standardbiblioteket och egenutvecklade funktioner för att simulera en mindre fastighetsmiljö.

Abstract

Ignition SCADA has in recent years gained more traction in Europe due to its flexibility and web-based interface. In response to this trend Init Sweden AB has decided to create a degree project centered around developing a standard library to streamline development of future SCADA-systems using the Ignition platform. Init Sweden AB has also seen a need to develop a good demonstrator to make it easier to showcase the core concepts of SCADA-systems.

The standard library follows the internal standards used by Init Sweden AB and includes eleven common elements used by the company within real estate automation.

As a part of the project, a Proof of concept was developed using the developed library meant to be used as a demonstrator. The demonstrator includes features from the developed library and some custom made functions meant to simulate a simplified building.

Innehåll

1
Inledning

Vid större projekt där man använder sig av Programmable logic Controller (PLC) för att styra olika typer av processer, som fabriker, broar och fastigheter används Supervisory Control and Data Acquisition (SCADA) för att övervaka och hantera styrning av alla PLC:er och annan hårdvara vilket underlättar arbetet för exempelvis operatörer. För att ett SCADA-system ska vara funktionllt och bidra till en enklare arbetsmiljö för operatörer och andra som kommer att interagera med SCADA-systemen måste systemen byggas upp på ett korrekt sätt. Genom att skapa systemen i enlighet med standarder, både inom branschen och inom företaget Init, blir systemen ingenkänningsbara för både de som utvecklar dem och för de som ska använda systemen. Ett standardbibliotek som följer en väl utarbetad styleguide och som bygger på etablerade branschstandarder på funktion och utseende underlättar därför för både de företag som utvecklar systemen och för de som ska använda systemen i drift.

1.1 Bakgrund

Init Sweden är ett företag aktivt inom automationssektorn som arbetar mycket med att utveckla SCADA-system, vilket är en form av övervakningssystem som används för att styra och övervaka industriella processer. De vill börja använda SCADA-plattformen Ignition vilket saknar ett standardbibliotek för vanliga processobjekt. Ett processobjekt är en form av bild som används inom SCADA-system där bilden representerar en fysisk enhet, som exempelvis en pump. För att minska startsträckan för utvecklingen av nya SCADA-system i Ignition vill Init Sweden utveckla ett standardbibliotek av processobjekt som används mycket inom fastighetsprojekt.

Ignition SCADA är en SCADA-plattform som används mycket i USA men är inte lika stor i Europa ännu. Ignition används främst för att plattformen är webbaserad och därmed väldigt flexibel. Där tidigare SCADA-system har krävt speciella mjukvaror för att användas behöver Ignition enbart en webbläsare.

1.2 Syfte

Syftet med examensarbetet är att skapa ett enhetligt och standardiserat bibliotek för Ignition SCADA, för att minimera uppstartstiden vid framtida fastighetsprojekt. Projektet har även i syfte att utveckla ett Proof of Concept för att ha som stöd vid utbildning om SCADA system vid till exempel mässor.

1.3 Avgränsningar

Projektet behandlar inte alla processobjekt som finns inom branschstandarden eftersom arbetet skulle blivit för stort för projektet. Därför tas enbart ett antal utvalda processobjekt fram. Dessa objekt valdes ut i samråd med erfarna systemutvecklare på företaget Init som jobbar med utveckling av fastighetsprojekt. I listan nedan presenteras de 11 processobjekt som ska skapas.

- Pump
- Fläkt
- Kompressor
- Spjäll
- Tvåvägsventil
- Trevägsventil
- Givare
- Differensegivare
- Timer
- Lampa
- Frekvensomriktare

Processobjekten som används i Proof-of-Conceptet är inte kopplade till någon fysisk motsvarighet utan alla objekt är simulerade versioner som programmerats i PLC,

1.4 Mål

Målet med projektet var att utveckla ett standardbibliotek, bestående av ett antal processobjekt anpassade för plattformen Ignition SCADA. De objekt som tagits fram är främst för användning inom fastighetsprojekt men kan även användas inom andra områden.

För att redovisa standardbibliotekets funktion genomfördes ett Proof of Concept där standardbiblioteket nyttjas. Konceptet består av en simulerad fastighetsmiljö som drivs av en PLC som är kopplad till Ingition där det skapade biblioteket används. Det ska vara möjligt att använda konceptet som demonstartor av SCADA-system, exempelvis vid mässor.

- Processobjekten ska följa standarderna satta i FlexFas och Inits HMI style guide.
- Processobjekten ska vara skalbara för att ha möjligheten att användas på olika skärmstorlekar.
- Konceptet ska vara interaktivt och skalbart, samt vara ett bra visningsmaterial för mässor och liknande.

2
Teknisk bakgrund

I deet här avsnitt presenteras den tekniska bakgrund som är nödvändig för resten av rapporten. Programmet Ignition SCADA har använts i projektet vilket presenteras nedan med tillhörande teori.

2.1 SCADA

Supervisory Control And Data Acquisition (SCADA) är ett mjukvarusystem som används för styrning och övervakning av större processer i realtid [2]. SCADA-systemet samlar in data från flera olika enheter som styrenheter och sensorer i övervakningssyfte. SCADA-systemet sammanställer datan på en central server där systemet även lagrar historisk data. Systemet har även förmågan att skicka signaler till styrenheter för att påverka processer.

Informationen som SCADA-systemet samlar in visas för en operatör genom ett Human-Machine interface (HMI) som är ett interaktivt lager mellan server och operatör och möjliggör visualisering av datan samt låter operatören påverka parametrar hos det större systemet.

2.2 OPC UA

OPC UA (Open Platform Communications Unified Architecture) är ett plattformsoberoende kommunikationsprotokoll som utvecklats med syftet att möjliggöra kommunikation mellan enheter och plattformar av olika fabrikat [3]. Det finns flera äldre OPC protokoll men OPC UA protokollet är en sammanslagning och uppgradering av tidigare OPC protokoll såsom OPC Data Access (DA), OPC Alarms and Events (AE) och OPC Historical Data Access (HDA). Genom denna sammanslagning skapar det nya protokollet, OPC UA, ett enhetligt, flexibelt och säkert system, vilket har lett till att det blivit branschstandard inom industriell automation.

2.3 UDT

User defined types (UDT) är en typ av objekt definierade enligt IEC 61131-3 standarden[8]. Detta är en standard för programspråk för PLC:er. UDT:er används för att gruppera ihop flera olika minnesvärden till en form av objekt som sedan kan användas som mall för att kunna skapa flera komponenter i ett system med samma konfiguration[4].

2.4 Processobjekt

Processobjekt är en form av bild inom SCADA-system som används för att snabbt kunna införa flera likadana enheter med samma symbol samt signaler. Symbolen inom processobjektet brukar representera en fysisk enhet som till exempel en pump. Signalerna som kopplas till processobjektet i SCADA-systemet kommer från PLC:n som det fysiska objektet i sin tur är kopplad till. Signalerna till ett processobjekt brukar samlas ihop i en UDT.

2.5 Flexbox

Nu för tiden måste webbsidor vara flexibla och anpassade för många olika typer av skärmar och enheter av olika storlekar, såsom mobiltelefoner och surfplattor. För att underlätta designen av webbsidor skapades verktyget Flexbox som positionerar objekt på sidan utan att veta objektens storlek [5]. Flexbox använder sig av olika positioneringsattribut för att justera hur objektet agerar när flexboxen ändrar storlek.

2.6 Ignition SCADA

Ignition SCADA är en utveklingsplattform för SCADA-system. Det finns två primära moduler för att skapa HMI och SCADA-system i Ignition: Vision och Perspective. Vision fungerar som de flesta andra utvecklingsprogram för SCADA medan Perspective använder sig av webbaserad mjukvara [6] för att skapa operatörsgränssnitt eller HMI. Det gör det möjligt att använda det skapade systemet på mobila enheter och kräver bara att enheten har tillgång till en webbläsare.

2.6.1 Views

I modulen Perspective i Ignition används "views" eller vyer för att skapa HMI-bilderna. Dessa vyer kan agera dockat fönster, popup-fönster, och som huvudsida, men de skapas på samma sätt[7]. Vyerna kan även läggas i andra vyer vilket gör dem till "embedded views", det gör det möjligt att skapa mindre objekt som en egen vy och sedan lägga in dessa på en huvudsida vilket är den större vyn. För att vyerna ska vara dynamiska i storlek kan de byggas med hjälp av flexcontainrar vilket är en typ av container som bygger på vektyget Flexbox. Containrarna kan kapslas in i varandra för att få alla komponenter i vyn att agera som man vill.

2.6.2 Script

I Ignition finns det olika sätt att manipulera data för att få ett fungerande HMI. Huvudsakligen görs det i form av olika typer av script som manipulerar olika parametrar och taggar, där en tagg är ett datavärde som antingen är statiskt eller dynamiskt. De två huvudscripten är "change-scripts" och "Event-scripts" där change-script exekveras när taggen, parametern eller objektet som scriptet är kopplat till ändrar

värde och event-scripts exekveras på ett objekt när ett förbestämt event inträffar som scriptet är kopplat till. Exempelvis kan en knapp trigga ett event-script när den blir nedtryckt. Båda dessa script skrivs med hjälp av programmeringsspråket python. Sedan finns även script som skrivs med hjälp av expressions. Dessa är till för att manipulera värdet från en tag och presentera det på andra sätt, ett exempel på det är ifall man har en tag som följer ett temperaturvärde givet i Celsius kan man använda ett expression-script för att få värdet i Fahrenheit istället.

2.6.3 Taggar

Ignition använder sig av taggar för att spara och manipulera data från olika källor. Det finns sex huvudtyper av taggar i Ignition: memory-, OPC-, expression, query-, reference- och derived-taggar. Minnestaggar används för att spara ett värde och är inte kopplade till någon annan hårdvara eller mjukvara, utan värdet på taggarna kommer antingen från att vara manuellt skrivet på dem eller som en input från programmet. OPC-taggar är de taggar som är kopplade till PLC:er och får sitt värde via en OPC-adress som är kopplat till en minnestagg i PLC:n. Expression-tags fungerar på samma sätt som expression-script men här sparas det nya värdet på en egen tagg. Query-taggar är kopplade mot en databas och har en speciell query som den ligger och söker efter i databasen. Reference-taggen refererar till en annan tagg och derived-tag fungerar på liknande sätt som en reference-tagg men de har även funktionaliteten att kunna använda expressions på värdet från huvudtaggen.

2.7 Datablock (DB)

Datablock (DB) är en form av datastruktur som defineras i IEC 61131-3 [8] och används för att organisera och lagra data i PLC-program. Informationen som lagras i datablock lagras där för att kunna användas av olika delar av styrsystemet.

Det finns två varianter av datablock: instanserade datablock och globala datablock. Instanserade datablock används i samband med ett funktionsblock. Det går att jämföra med en lokal variabellista. Globala datablock används för att lagra information som hela programmet behöver ha tillgång till och går att jämföra med en global variabellista.

3
Metod

En kombination av mjukvara och hårdvara användes för att fullfölja projektet. På begäran av företaget Init användes Ignition SCADA som utvecklingsprogram för SCADA-systemet. SCADA-systemet kombinerades med en PLC från Siemens som progammerades i TIA-portal för att göra ett Proof of Concept.

3.1 PLC

En PLC eller Programmable logic controller är en form av industridator som används för att styra industriella processer. PLC:er kör en typ av programmeringspråk som liknar relä styrning och är tänkta att vara mer tillförlitliga och mer robusta än en vanlig dator. Ett grundläggande krav hos PLC:er är att en PLC skall kunna vara igång i flera år utan omstart. I projektet användes en PLC från förertaget Siemens för att simulera och styra en fastighet.

3.2 HMI Style guide

I projektet har en HMI style guide från Init använts för att bland annat färgsätta processobjekten och även SCADA-systemet i Proof-of-Conceptet. Style guiden är skapad i enlighet med standraden ISA101 som riktar in sig på just HMI-utveckling. Det genomgående temat är att använda så lite färg som möjligt och jobba mycket med gråskala för objekt, text och bakgrunder. Den färg som faktiskt används är främst för att uppmärksamma att något inte är normalt för systemet, exempelvis de ikoner som visas ifall ett objekt har ett aktivt larm (röd, orange eller gul), eller om objektet är i manuellt läge (lila).

3.3 FlexFas

FlexFas är ett ramverk utvecklat för SCADA-systemet Citect SCADA (Plant SCADA) vilket är det program som Init använder sig av, när projektet genomförts, för utveckling av fastighetsprojekt. I projektet används ramverket som utgångspunkt vid skapandet av processobjektsymbolerna eftersom det har en etablerad standard på utfomningen av objekten som skapades i projektet. Användningen av FlexFas bidrar till högre igenkänning för både företaget och framtida användare av SCADA-system som använder det utvecklade standardbilbioteket i projektet.

3.4 Inkscape

För att förverkliga processobjekten i Ignition SCADA skapades bilder med Scalable Vector Graphics (SVG) -format som representerar de olika processobjekten och indikatorflaggorna. Dessa skapades i programmet Inkscape och följde FlexFas-standarden för objektens utformning för att efterlikna de symboler som används tidigare på företaget.

3.5 Ignition Designer

I projektet utnyttjades SCADA-systemet Ignition SCADA från Inductive Automation, vilket var från begäran av Init. För att utnyttja Ignition användes deras Integrated Development Environment (IDE), Ignition Designer.

3.6 TIA Portal

Totally Integrated Automation Portal (TIA portal) är en utvecklingsmiljö som används för att programmera PLC:er tillverkade av leverantören Siemens. I projektet används TIA portal V19 för att utveckla den simulerade miljön som används i Proof of Concept.

4

Resultat

Projektet har enligt de krav, mål och syfte som ställts upp inom projeket uppnått ett gott resultat. Projektet har tagit fram ett användbart standardbibliotek för Ignition SCADA som gör utvecklingen av nya SCADA-system enklare och biblioteket är utformat för att följa den praxis som råder på företaget. I projektet ingick det även att framställa ett Proof of concept som förevisar biblioteket och kan användas som visningsmaterial vid utbildning om SCADA-system och dess funktion, samt som förevisningsmatrial vid till exempel mässor för att förtydliga vad ett SCADA-system är.

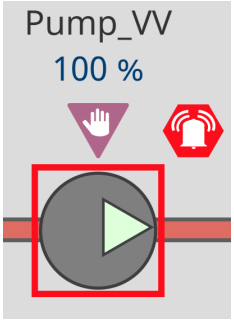
4.1 Standardbiblioteket

Det framtagna standardbiblioteket består av de elva processobjekten som företaget efterfrågade innehållande de standardenliga styrsignalerna, olika statussymboler, samt korrekt visuellt beteende. Biblioteket finns tillgängligt för de anställda på företaget genom det skapade projektet i Ignition som sedan kan ärvas till nya fastighetsprojekt. Det har även skapats tillhörande dokumentation till standardbiblioteket som går att hitta i Appendix A.

4.1.1 Processobjekt som view

Bilderna för symbolerna och flaggorna som bildar processobjekten är skapade i SVG-format eftersom bilderna inte får synliga pixlar ifall bilden skalas upp som det blir ifall man använder rastergrafik. De skapade SVG-bilderna implementerades i Ignition som "embedded views". Vilket gjorde det möjligt att manipulera specifika element på bilden, exempelvis om bilden består av en cirkel, en triangel och en fyrkant, kan man ändra deras egenskaper individuellt såsom storlek, färg och synlighet. Med hjälp av "views"eller vyer programmerades objekten för att agera på rätt sätt. I vyerna kapslades symbolerna in i flexkontainrar för att göra dem dynamiska och skalbara. Dessa vyer kopplades även ihop med en UDT för respektive processobjekt.

Genom att koppla ihop taggarna från PLC:n mot bildernas olika element kunde elementen manipuleras i SCADA-systemet beroende på objektets tillstånd. Exempelvis kan en del av symbolen för en pump göras grön när pumpen är i aktivt läge, vilket man ser i figur 4.1 nedan där triangeln på pumpsymbolen är grön.



Figur 4.1: En pump som processobjekt skapat i projektet. Objektet visar namn, aktuellt processvärde, manuell styrning och kritiskt larm

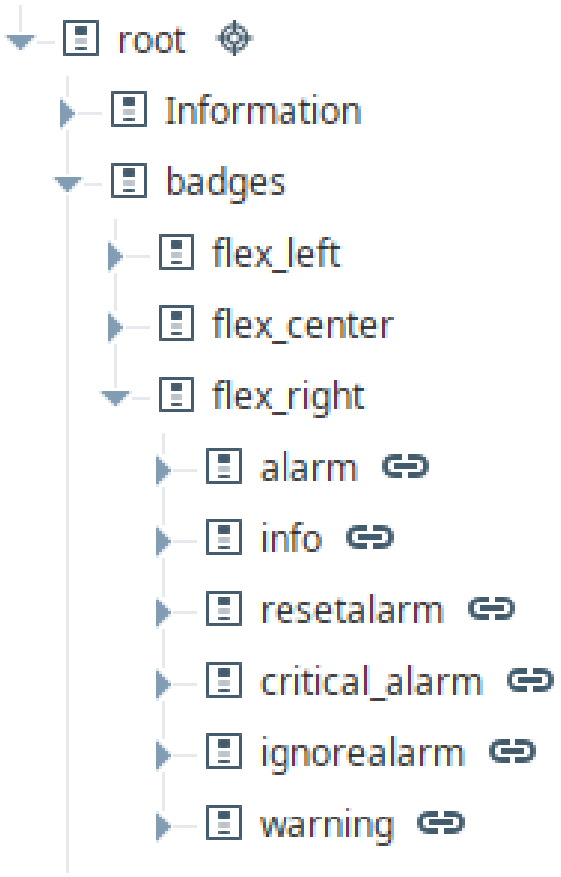
Flaggorna som visar processobjektens status har tre olika platser ovanför processobjekten, en till vänster, en till höger och en i mitten. Det är nio olika statusflaggor, vilka visas i fig 4.2, som är programmerade på de olika objekten. Alla statusflaggor är skapade som SVG-bilder i projektet och är baserade på referensbilder från Inits HMI style guide .

Figur 4.2: Alla flaggor från vänster till höger: Kritiskt larm, larm, varning, information, blockera larm, återställ larm, manuellt läge, simulering, intelock

I figur 4.2 vissas de nio olika flaggorna. Flaggorna är manuellt läge, simulering, kritiskt larm, larm, varning, information, blockera larm, återställ larm och interlock. Interlock-flaggan visas om det är något annat i flödet som gör att objektet inte kan användas på rätt sätt, exempelvis om en pump inte startar vilket kan göra att flödet genom en ventil inte finns även om den är öppen. Då hade Interlock-flaggan visats på ventilen.

Området där flaggorna visas består av en flexcontainer med tre nästlade flexcontainrar, en för varje förutbestämd plats ovanför objektet. I varje nästlad flexcontainer finns det ytterligare flera nästlade flexcontainrar, där det i varje flexcontainer placerades en flagga, se figur 4.3 för trädsturkturen av containrarna. Flexcontainers har ett element som kallas "visibility"som tillåter att hela containern döljs och inte tar någon plats, vilket gör att flera symboler kan dela på en plats genom att elementen

aktiveras och avaktiveras i varje individuell flexcontainer för de tre flaggplatserna.



Figur 4.3: Trädet av flexcontainers

För att förtydliga statusen hos processobjekteten implementerades en ram runt objektsymbolen, se den röda ramen runt pumpen i fig 4.1. Eftersom det kan vara fler aktiva larm samtidigt implementerades en prioriteringsordning med hjälp av ett expression-script på ramen runt processobjektet för att säkerställa att den högst prioriterade statusen alltid visades. Varje flaggflexcontainer fick ett specifikt värde associerat med sig och även en prioriteringsordning. Värdet kommer från en tagg kopplad till PLC:n som varje flaggcontainer läser från när värdet ändras på taggen, om taggvärdet stämmer överrens med ett flaggvärde kommer den flaggan att visas ovanför processobjektet.

4.1.2 Taggar och UDT

UDT:er implementerades för varje processobjekt för att standardisera taggarna som kopplades till varje processobjektsvy samt möjliggjorde användningen av "drag-and-drop"av processobjekten in i vyer. De taggar som lades till i UDT:erna var tagna delvis från FlexFas. Majoriteten av objekten har taggarna _AL, _CMD, _V, _MODE och _interlock. Några av objekten har även taggar för deras bör- och ärvärde. De flesta taggarna är av typen integer där de olika siffrora betyder olika saker. Som

exempel används taggen _AL för att skicka information om vilken nivå av larm som är aktivt på objektet, där 0 är inget larm, 1 är kritiskt larm, 2 vanligt larm och 3 är varning. _CMD är till för information om någon av flaggorna "blockera larm", "information"eller "återställ larm"är aktiv och ska visas över objektet. De olika taggarna finns dokumenterade för varje objekt i appendix A.

Alla dessa taggar är skapade som OPC-taggar. Eftersom taggarna är skapade i en UDT måste OPC-adressen vara dynamisk, annars kommer alla instanser av UDT:n att vara kopplade till samma adress. För en Siemens PLC kopplat till Ignition kan en OPC-adress se ut på följande sätt:*[S7_Example]DB0,I2.0*, där texten inom hakperentes är namnet på PLC:n. *DB0* beskriver var objektet finns i PLC:n. I det fallet finns adressen till objektet i datablock noll. *I2.0* beskriver dels vilken datatyp taggen kommer ha, här är det en integer, och dels vilken offset taggen har i det specificerade datablocket, i exemplet är den 2.0.

OPC-adresserna är gjorda dynamiska genom att använda en parameter i UDT:n på följande sätt: *[S7_Example]DB0,I{Offset+2*.0}. Här är "Offset"parametern och om den är satt till 0 kommer adressen länka till offset 2.0 i PLC:n. För att den aktuella lösningen ska fungera måste taggarna ligga i samma ordning när de skapas i PLC:n. För att säkerställa att det stämmer skapades även UDT:er på PLC-sidan för de olika processobjekten.

4.1.3 Popup-fönster

Vissa av processobjekten får även upp ett popup-fönster ifall man klickar på symbolen för objektet. Ett popup-fönster för en pump presenteras i fig 4.4. Det möjliggjordes med hjälp av ett event-script på processobjektets symbolbild i processobjektvyn, där det programmeras att en specifik vy ska öppnas som ett popup-fönster om symbolen blir klickad på. I fönstret kan man se information om objektet och även skicka kommandon till det. För att undvika att popup-fönstret bara visar information om ett och samma objekt oberoende av vilket man klickar på skickas tag-pathen för det specifika objektet med i event-scriptet från objektets vy. Detta gör att även popup-fönstren är dynamiska ifall man använder sig av "drag-and-drop"för ett processobjekt.

Figur 4.4: Popup-fönster för en fläkt som är aktiv där manuellt läge är aktiverat och är- och börvärdet är 50%

4.1.4 Dokumentation

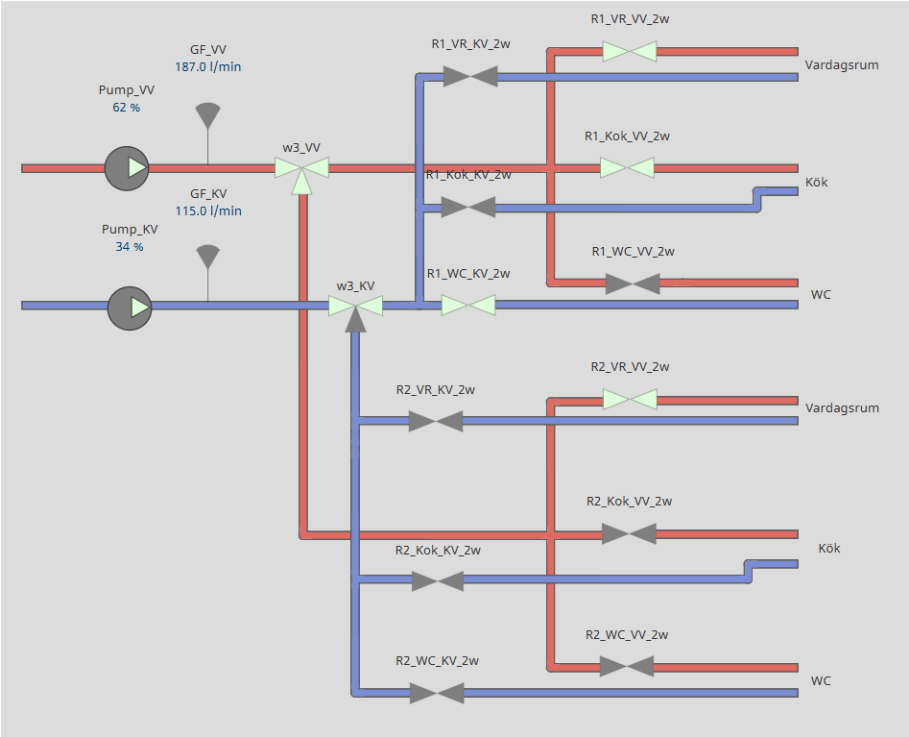
Projektet framställde även tillhörande dokumentation för processobjekten enligt samma upplägg som återfinns i FlexFas. Dokumentation finns tillgänglig i appendix A.

4.2 Proof of Concept

Fokuset lades på att ta fram ett koncept som visade hur ett SCADA-system kan se ut för att kunna användas som förevisningsmaterial på mässor och liknande. Eftersom de processobjekt som skapades i projektet är till för fastighetsprojekt, var det självklart att konceptet skulle visa upp en fastighet. Konceptet bygger därför

på en simulerad fastighetsmiljö där en PLC simulerar en fastighet i form av dess ventilation- och vattensystem och ett SCADA-system utvecklades med hjälp av det framtagna standardbiblioteket.

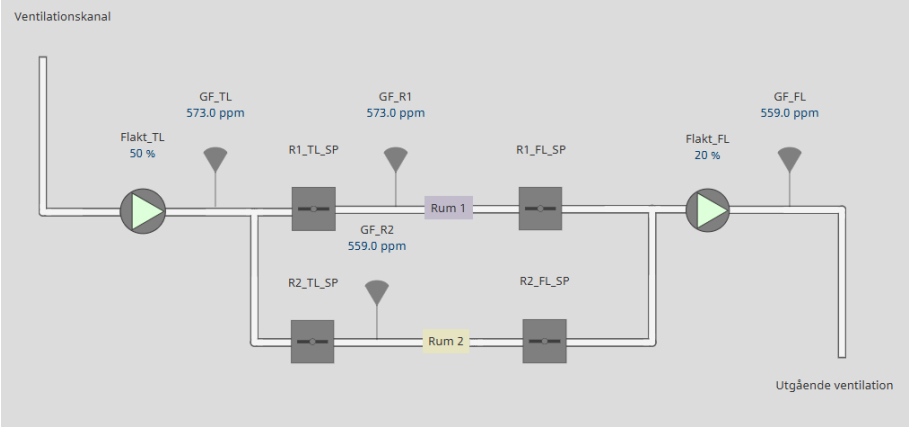
Konceptet består av tre våningar där det finns ventilation och vatten på alla våningar. Varje våning består av två lägenheter med vardagsrum, kök och toalett där det finns varm- och kallvattenledningar dragna till alla rum. Vattensystemet för en våning visas i fig 4.5.



Figur 4.5: Skapad HMI-bild för vattensystemet

Systemet i fig 4.5 består av en kallvattenpump, en varmvattenpump, var sin flödesgivare, samt ett antal ventiler för att styra vattenflödet till de olika rummen.

Ventilationssystemet i konceptet presenteras i fig 4.6 och ser till att lägenheterna har tillräckligt med friskluft genom att mäta koldioxidhalten.



Figur 4.6: Skapad HMI-bild för ventilationssystemet

Tillförseln av friskluft styrs av en tilluftsfläkt och en frånluftsfläkt som finns på varje våning. Luften styrs även med hjälp av ett antal spjäll mellan fläktarna och de olika lägenheterna.

4.2.1 Simulering av en fastighet

För att simulera fastighetsmiljön nyttjades en PLC som styrsystem som hade i huvudsaklig uppgift att styra de olika vatten- och ventilationssytemen i det fiktiva huset.

För att implementera det i PLC:n programmerades funktionsblock för varje processobjekt för att simulera styrsignaler. Vilket gjordes främst med funktionsblock i TIA Portal med tillhörande DB. För de processobjekt som krävde varierande värden implementerades en egenutvecklad PI-regulator. PI-regulatorn valdes i det här fallet för att försöka återskapa ett realistisk värdesförändring i Proof-of-Conceptet genom att agera som ett dynamiskt element. Nedan är de funktionsblock som utvecklades:

- Flödesgivare
- Trevägs-ventil
- Pump
- Koldioxidgivare
- PI-regulator
- Fläkt
- Spjäll
- Tvåvägs-ventil

Funktionsblocken nyttjades i kombination för att bygga större funktionsblock som i tur simulerade hela våningsplan. Våningsplanen programmerades för att vara så automatiserade som möjligt och vara så självreglerande som möjligt.

Genom SCADA-systemet implementerades det även en manuell styrning. Där en

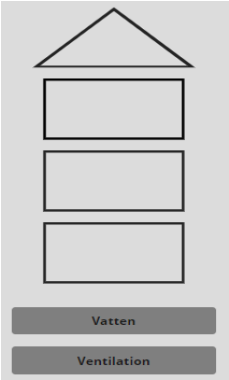
förfrågning från SCADA till PLC:n som sen löste förfrågningen genom att till exempel öppna rätt ventiler.

4.2.2 SCADA-systemet

De simulerade processobjekten i PLC:n kopplas ihop med Ignition och visas med hjälp av HMI-bilder. Dessa HMI-bilder ska föreställa en fastighet bestående av lägenheter på tre våningar där deras vatten- och ventilationssystem visas.

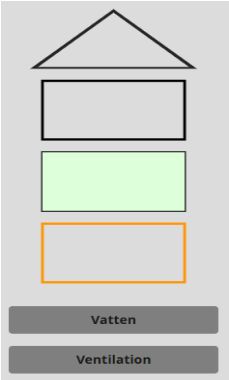
HMI-bildernas utseende bygger på hur fastigheter brukar vara uppbyggda i riktiga SCADA-projekt där man ofta använder sig av en layout som liknar ett driftkort för att enkelt få en övergripande bild av förloppet. För att realisera det i Ignition användes ett "pipe-vektyg"för att skapa kopplingar mellan de olika processobjekten precis som på ett driftkort, se fig 4.5 och 4.6.

Navigeringen mellan våningarna sköts med hjälp av en förenklad version av ett flervåningshus , se figur 4.7, samt två knappar för att skifta mellan ventilationssystemet och vattensystemet.

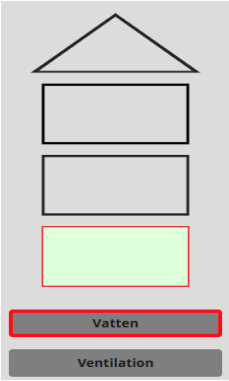


Figur 4.7: Navigationsobjekt i form av ett förenklat flervåningshus med tillhörande knappar för att byta vy mellan vatten och ventilation

Navigeringen sköts alltså visuellt, vilket gör det möjligt att se vilken våning man är inne och tittar på eftersom den våningen blir ifylld med grönt som i fig 4.8 och 4.9. Man kan även gå till en annan våning genom att klicka på önskad våning i huset.



Figur 4.8: Navigationsobjekt som indikerar att den aktiva vyn är våning 2 och det är larm på våning 1



Figur 4.9: Navigationsobjekt som indikerar att den aktiva vyn är våning 1 och det är larm på vatten-delen på våning 1

Navigeringshuset fyller även funktionen som larmindikator för hela husets alla system, liknande hur processobjekten får en färgad ram runt sig när det finns aktiva larm på objektet, får den våningen med aktivt larm också en ram runt sig med samma färg som det aktiva larmet, vilket också visas i fig 4.8 och 4.9. Den visuella navigeringen gör det möjligt att navigera sig till det aktiva larmet genom att följa färgramen på navigeringshuset och de tillhörande knapparna som byter mellan våningens vatten- och ventilationssystem.

SCADA-systemet innehåller även en "öppna dörrar"-knapp och en översiktbild där man ser våningarnas layout. Knappen är till för att simulera att dörrar öppnas i lägenheterna och släpper in ny luft vilket påverkar ventilationssystemet.

5
Slutsats och diskussion

Överlag har projektet resulterat i ett användbart bibliotek och ett väl fungerande Proof of Concept, samt bidragit till bättre hållbarhet inom automationsbranschen genom effektivisering. I projektet har förbättringsområden identifierats samt framtida utvecklingsmöjligheter.

5.1 Biblioteket

Projektet har uppnått alla krav som ställts från företagets sida med gott resultat och skapat ett bibliotek som underlättar utvecklingen av fastighetsbaserade SCADA-system i Ignition men det finns några förbättringsområden.

I första hand har det framtagna processobjekten några egenskaper som avviker från FlexFas och HMI-standarden. En av dessa avvikelser är ramen som uppstår runt processobjekten. Den blinkar inte, vilket det enlig standarden ska göra för att underlätta identifieringen av vilket processobjekt som larmar eftersom ögat dras till saker som rör på sig. Det är något som aldrig implementerades under projektet eftersom andra funktioner prioriterades i längden. Syftet bakom funktionen hade varit en ökad användarvänlighet för operatören.

En styrka med Ignition SCADA är att det är en webbaserad platform och går att köra på både mobila och stationära skärmar. I projektet har bibliotekets processobjekt enbart använts på datorskärmar och inte testats på mindre skärmar som till exempel en mobilskärm. Vilket kan medföra svårigheter vid skalning av objekten för att passa in på en mindre skärm. Samt har biblioteket i nuläget inget stöd för att kunna rotera ett processobjekt utan objektet måste alltid vara på "rätt"håll vilket medför begränsningar i hur objekten kan placeras i ett SCADA system.

5.2 Proof of Concept

Framtagning och precisiering av Proof of Concept var svårare än vad som var tänkt från början och tog ett flertal tankevändor att lära sig. Syftet bakom Proof-of-Conceptet var att förevisa biblioteket och dess ingående delar samt att kunna agera förevisningsmaterial till mässor och liknande. Just det senare innebär att det fanns ett behov att avväga mellan att bygga ett komplicerat system och ett mer lättförståligt system. Det mer komplicerade systemet hade medfört att mer processobjekt

ingått i konceptet och därmed visat upp mer av det skapade standardbilioteket men det hade blivit mer svårförstått och inte ett så bra förevisningsmaterial för personer utan kunskap om SCADA-system. Därför valdes ett mer lättförstått system med färre processobjekt för att underlätta vid visning av systemet. Dock gjordes det ett val att bygga en mer komplicerad PLC-sida med regulatorer för att försöka skapa ett mer realistisk och interaktivt Proof of Concept.

Proof-of-Conceptet uppfyller samtliga mål och kravställningar som sattes på det. Konceptet visar upp delar av det skapade standardbilioteket samt är interaktivt för att kunna användas som förevisningsmaterial. Mycket av fokuset kring Proof-of-conceptet lades i bakgrunden där arbetet inte synts genom nyttjadet av PI-regulatorer och PLC-kod. Dock är det som gjorts uppbyggt på ett sådant sätt att Proof-of-Conceptet går att skala upp och ner med hjälp av UDT:er.

5.3 Hållbarhet

Utvecklandet av standardbibliotek för system bidrar till en bättre hållbarhet genom minskad utecklingstid för nya system. Fastigheter som det den skapade Proof-of-Conceptet föreställer, högt moderniserade fastigheter med flera automatiserade system är mer hållbara eftersom energiförbrukningen minimeras i den mån de går genom automatiserade processer. Nya standardbibliotek minskar tiden det tar att installera liknande system i nya fastigheter vilket bidrar till en miljövänlig fastighetsautomation då fastigheterna blir miljövänliga på grund av minskad energiförbrukning.

Projektet bidrar även till en minskad arbetslast för de anställda som jobbar på företaget med ingition. Det bidrar även genom en förbättrad arbetsmiljö för de anställda med mindre stress och mer tid samt bättre möjligheter att fullfölja sina arbetsuppgifter. Samt lägre arbetskostnader genom effektivisering av arbete.

Litteraturförteckning

[1] M. Gustaver, "A Chalmers University of Technology Master's thesis template for LaTeX," Unpublished, 2020.

[2] W. C. Hayes, O. P. Malik, A. Bose, H. M. Merrill, L. Goel, and C. W. Gellings, "Electric power systems," *AccessScience*, Jan. 2020. [Online]. Available: https://doi.org/10.1036/1097-8542.216900

[3] OPC Foundation, "Unified Architecture – Landingpage," OPC Foundation. [Online]. Available: https://opcfoundation.org/about/opc-technologies/opc-ua/. [Accessed: May 13, 2025].

[4] Inductive Automation, "User-Defined Types (UDTs) | Ignition 8.1 Docs," Inductive Automation. [Online]. Available: https://www.docs.inductiveautomation.com/docs/8.1/platform/tags/user-defined-types-udts\protect\penalty-\@M. [Accessed: May 13, 2025].

[5] Chalmers University of Technology, "Skillport," Chalmers. [Online]. Available: https://chalmers.skillport.com/skillportfe/main.action?path=summary/BOOKS/163644. [Accessed: May 13, 2025].

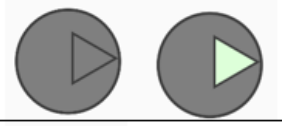
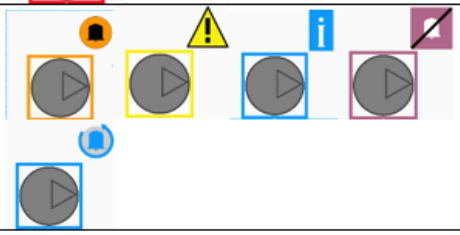
[6] Inductive Automation, "Perspective | Ignition User Manual," Inductive Automation. [Online]. Available: https://www.docs.inductiveautomation.com/docs/8.1/ignition-modules/perspective. [Accessed: May 13, 2025].

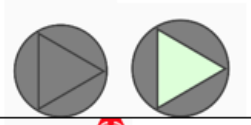
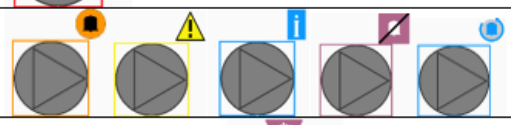
[7] Inductive Automation, "Views in Perspective | Ignition User Manual," Inductive Automation. [Online]. Available: https://www.docs.inductiveautomation.com/docs/8.1/ignition-modules/perspective/views-in-perspective\protect\penalty-\@M. [Accessed: May 13, 2025].

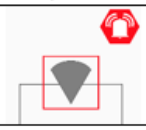
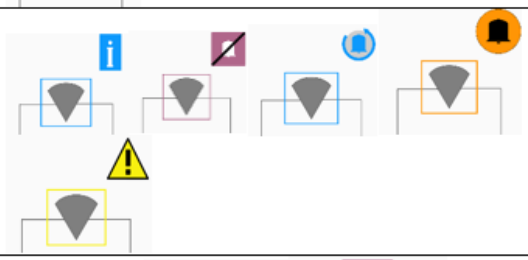
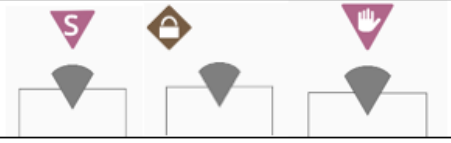
[8] IEC, "IEC 61131-3: Programmable controllers – Part 3: Programming languages," International Electrotechnical Commission, Geneva, Switzerland, 2013. [Online]. Available: https://www.iec.ch/. [Accessed: May 13, 2025].

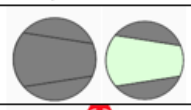
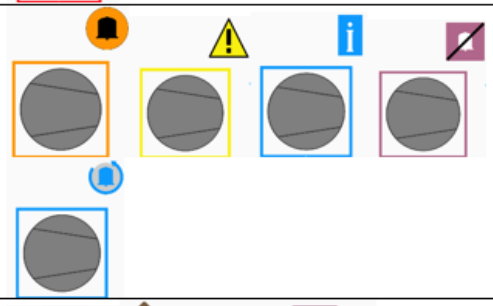
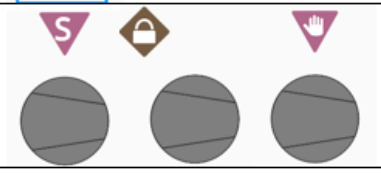
A

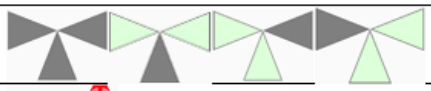
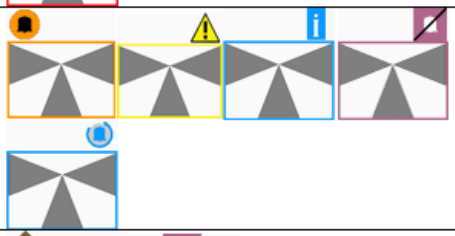
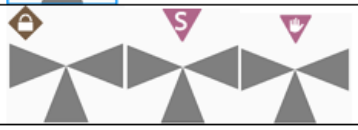
Dokumentation av processobjekt

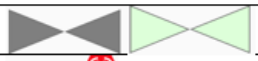
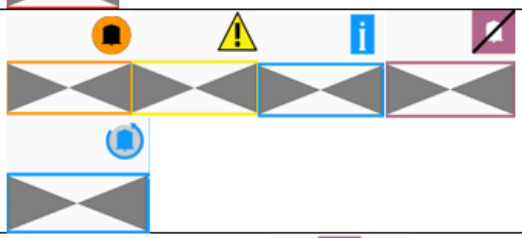
Version 1.0	Datum för senaste version 2025-04-16
Pump	Processobjekt av en pump. Finns möjlighet att indikera drift, olika larm och handstyrning av objektet.
_V	Pumpens indikator blir grön och indikerar aktivt läge
_AL	1, 2 eller 3 beroende på prioritet (1 är kritiskt larm)
_CMD	4=info, 5=ignore alarm, 6=reset alarm
_MODE	1=Handstyrning, 2=simulering,
_Interlock	1= interlock
_PV	Process value
_SP	Setpoint
Normalt läge	
Vid kritiskt-larm	
Vid Övriga Statusar (Larm, varning,Info, Larm ignorerat, Larm återställ/kvittera)	

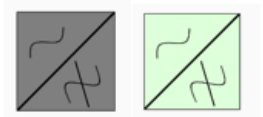
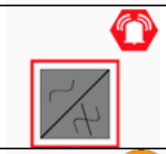
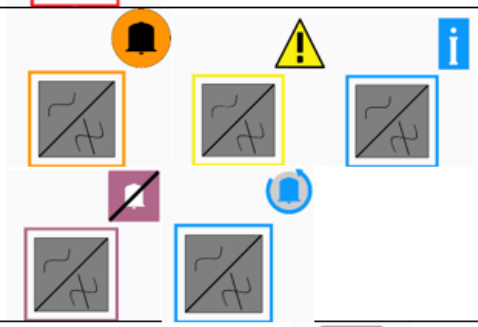
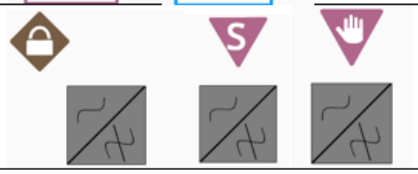
Version 1.0	Datum för senaste version 2025-04-02
Fläkt	Processobjekt av en fläkt. Finns möjlighet att indikera drift, olika larm, simuleringsläge och handstyrning av objektet.
_V	Indikator blir grön och indikerar aktivt läge
_AL	1, 2 eller 3 beroende på prioritet (1 är kritiskt larm)
_CMD	4=info, 5=ignore alarm, 6=reset alarm
_MODE	1=Handstyrning, 2=simulering,
_Interlock	1= interlock
_PV	Process value
_SP	Setpoint
Normalt läge och aktivt läge	
Vid kritiskt-larm	
Vid Övriga larm (Larm, varning,Info, Larm ignorerat, Larm återställ/kvittera)	
Interlock, simulering och 1=Handstyrning	

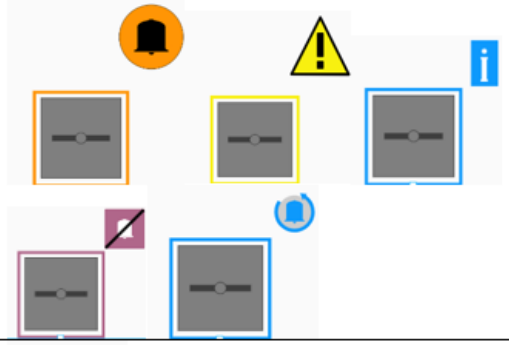
Version 1.0	Datum för senaste version 2025-04-02
Diff_Givare	Processobjekt av en diff-givare. Finns olika larm, simuleringsläge och handstyrning av objektet.
_V	Indikator blir grön och indikerar aktivt läge
_AL	1, 2 eller 3 beroende på prioritet (1 är kritiskt larm)
_CMD	4=info, 5=ignore alarm, 6=reset alarm
_MODE	1=Handstyrning, 2=simulering,
_Interlock	1= interlock
_PV	Process value
_SP	Setpoint
Vid kritiskt-larm	
Vid Övriga larm (Larm, varning,Info, Larm ignorerat, Larm återställ/kvittera)	
Interlock, simulering och 1=Handstyrning	

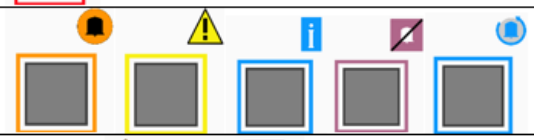
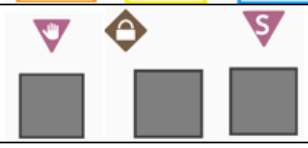
Version 1.0	Datum för senaste version 2025-04-02
Kompressor	Processobjekt av en kompressor. Finns möjlighet att indikera drift, olika larm, simuleringsläge och handstyrning av objektet.
_V	Indikator blir grön och indikerar aktivt läge
_AL	1, 2 eller 3 beroende på prioritet (1 är kritiskt larm)
_CMD	4=info, 5=ignore alarm, 6=reset alarm
_MODE	1=Handstyrning, 2=simulering,
_Interlock	1= interlock
_PV	Process value
_SP	Setpoint
Normalt läge och aktivt läge	
Vid kritiskt-larm	
Vid Övriga larm (Larm, varning,Info, Larm ignorerat, Larm återställ/kvittera)	
Interlock, simulering och 1=Handstyrning	

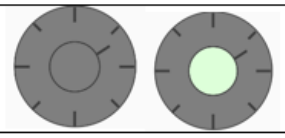
Version 1.0	Datum för senaste version 2025-04-016
Trevägsventil	Processobjekt av en trevägsventil. Finns möjlighet att indikera drift, olika larm, simuleringsläge och handstyrning av objektet.
_V	Indikator blir grön och indikerar aktivt läge
_AL	1, 2 eller 3 beroende på prioritet (1 är kritiskt larm)
_CMD	4=info, 5=ignore alarm, 6=reset alarm
_MODE	1=Handstyrning, 2=simulering,
_Interlock	1= interlock
Normalt läge och aktivt läge	
Vid kritiskt-larm	
Vid Övriga larm (Larm, varning,Info, Larm ignorerat, Larm återställ/kvittera)	
Interlock, simulering och 1=Handstyrning	

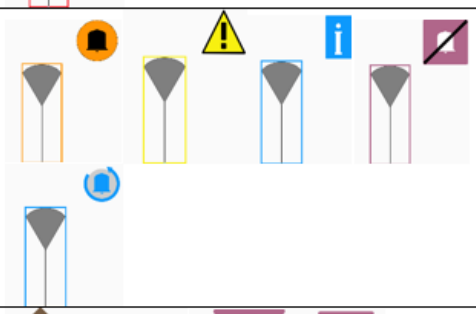
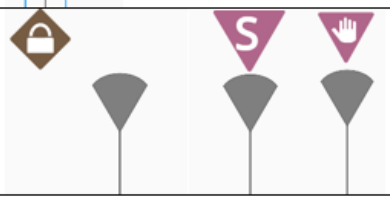
Version 1.0	Datum för senaste version 2025-04-16
Tvåvägsventil	Processobjekt av en tvåvägsventil. Finns möjlighet att indikera drift, olika larm, simuleringsläge och handstyrning av objektet.
_V	Indikator blir grön och indikerar aktivt läge
_AL	1, 2 eller 3 beroende på prioritet (1 är kritiskt larm)
_CMD	4=info, 5=ignore alarm, 6=reset alarm
_MODE	1=Handstyrning, 2=simulering,
_Interlock	1= interlock
Normalt läge och aktivt läge	
Vid kritiskt-larm	
Vid Övriga larm (Larm, varning,Info, Larm ignorerat, Larm återställ/kvittera)	
Interlock, simulering och 1=Handstyrning	

Version 1.0	Datum för senaste version 2025-04-16
Konverterare	Processobjekt av en konverterare. Finns möjlighet att indikera drift?, olika larm, simuleringsläge och handstyrning av objektet.
_V	Indikator blir grön och indikerar aktivt läge
_AL	1, 2 eller 3 beroende på prioritet (1 är kritiskt larm)
_CMD	4=info, 5=ignore alarm, 6=reset alarm
_MODE	1=Handstyrning, 2=simulering,
_Interlock	1= interlock
OP	Output
Normalt läge och aktivt läge	
Vid kritiskt-larm	
Vid Övriga larm (Larm, varning,Info, Larm ignorerat, Larm återställ/kvittera)	
Interlock, simulering och 1=Handstyrning	

Version 1.0	Datum för senaste version 2025-04-16
Spjäll	Processobjekt av ett spjäll. Finns möjlighet att indikera drift?, olika larm, simuleringsläge och handstyrning av objektet.
_V	Indikator blir grön och indikerar aktivt läge
_AL	1, 2 eller 3 beroende på prioritet (1 är kritiskt larm)
_CMD	4=info, 5=ignore alarm, 6=reset alarm
_MODE	1=Handstyrning, 2=simulering,
_Interlock	1= interlock
Öppet-, stängt- och mellanläge	
Vid kritiskt-larm	
Vid Övriga larm (Larm, varning,Info, Larm ignorerat, Larm återställ/kvittera)	
Interlock, simulering och 1=Handstyrning	

Version 1.0	Datum för senaste version 2025-04-16
Light_Basic	Processobjekt av en lampa. Finns möjlighet att indikera drift?, olika larm, simuleringsläge och handstyrning av objektet.
_V	Indikator blir grön och indikerar aktivt läge
_AL	1, 2 eller 3 beroende på prioritet (1 är kritiskt larm)
_CMD	4=info, 5=ignore alarm, 6=reset alarm
_MODE	1=Handstyrning, 2=simulering,
_Interlock	1= interlock
Normalt läge och aktivt läge	
Vid kritiskt-larm	
Vid Övriga larm (Larm, varning,Info, Larm ignorerat, Larm återställ/kvittera)	
Interlock, simulering och 1=Handstyrning	

Version 1.0	Datum för senaste version 2025-04-16
Timer	Processobjekt av en timer. Finns möjlighet att indikera drift?, olika larm, simuleringsläge och handstyrning av objektet.
_V	Indikator blir grön och indikerar aktivt läge
_CMD	
Normalt läge och aktivt läge	

Version 1.0	Datum för senaste version 2025-04-16
Givare	Processobjekt av en givare. Finns möjlighet att indikera, olika larm, simuleringsläge och handstyrning av objektet.
_AL	1, 2 eller 3 beroende på prioritet (1 är kritiskt larm)
_CMD	4=info, 5=ignore alarm, 6=reset alarm
_MODE	1=Handstyrning, 2=simulering,
_Interlock	1= interlock
_PV	Process value
_SP	Setpoint
Normalt läge	
Vid kritiskt-larm	
Vid Övriga larm (Larm, varning,Info, Larm ignorerat, Larm återställ/kvittera)	
Interlock, simulering och 1=Handstyrning	

INSTITUTIONEN FÖR Elektroteknik
CHALMERS TEKNISKA HÖGSKOLA
Göteborg, Sverige
www.chalmers.se

CHALMERS