UPPSALA UNIVERSITET

UPTEC STS14 026
Examensarbete 30 hp
Augusti 2014

Utredning och implementering av en prototyp för integration av Prevas FOCS och ABB 800xA

Kristian Bernström
Anders Näsman

UPPSALA
UNIVERSITET

Teknisk- naturvetenskaplig fakultet
UTH-enheten

Besöksadress:
Ångströmlaboratoriet
Lägerhyddsvägen 1
Hus 4, Plan 0

Postadress:
Box 536
751 21 Uppsala

Telefon:
018 – 471 30 03

Telefax:
018 – 471 30 00

Hemsida:
http://www.teknat.uu.se/student

Abstract

Study and implementation of a prototype for integration of Prevas FOCS and ABB 800xA

Kristian Bernström and Anders Näsman

ABB and Prevas have initiated a collaboration to sell a system to optimize steel industry furnaces, called FOCS. The purpose of this thesis is to investigate possibilities for integrating Prevas FOCS and ABB 800xA.

The result of the investigation is used for an implementation of a prototype of the integrated system. The study shows a general method that can be used when implementing two software systems. The prototype of the integrated systems is made with usability principles in mind. This is a very important aspect in order to create a good working environment for the operators of a steel plant. It is also important to follow communication standards when integrating software systems. In an industrial network used in the steel industry OPC is a standard for communication. We recommend ABB and Prevas to follow this standard when possible to make the integration smoother. To keep the cost of the integration to a minimum it is also recommended to reuse existing resources. This can however have a negative effect on usability and it is therefore important to keep a balance between cost and usability.

The prototype made in this thesis accomplishes the goal of transferring the functionalities used by operators of Prevas FOCS to 800xA so that operators can control the processes using only one integrated system.

Handledare: Lars-Göran Uhlander and Rickard Söderlund
Ämnesgranskare: Roland Bol
Examinator: Elísabet Andrésdóttir
ISSN: 1650-8319, UPTEC STS14 026

Populärvetenskaplig sammanfattning

Det har blivit allt viktigare att effektivisera energianvändningen i dagens tunga industrier. Ett sätt att åstadkomma detta är genom skapandet av system som optimerar processer som kräver mycket energi.

ABB och Prevas har inlett ett samarbete för att sälja ett system som används för att optimera styrning av ugnar i stålindustrin, kallat FOCS. I dagsläget är Prevas FOCS ett fristående system som kan användas i detta syfte. ABB tillhandahåller ett annat system för industrimiljöer, kallat 800xA. 800xA är ett mer övergripande styrsystem, med fokus på att övervaka och styra industriella maskiner medan Prevas FOCS primära egenskap är att effektivisera styrningen. För operatörer som styr dessa ugnar är det viktigt att ha programvara som hjälper och ger stöd i processen. Även om båda dessa system är bra var för sig finns det i dagsläget inget bra sätt att använda båda systemen samtidigt. En operatör som vill dra nytta av fördelarna hos både Prevas FOCS och 800xA tvingas använda två separata system, med olika gränssnitt och separata larmlistor.

Syftet med denna studie är att utreda möjligheter för att integrera Prevas FOCS i ABB:s styrsystem 800xA. Med integrering menas att skapa en brygga för att få de två fristående programmen att kunna interagera med varandra. Efter utredningen väljs det alternativ som anses bäst för att göra en prototyp. I denna studie används två huvudsakliga metoder för att åstadkomma en integration av systemen. Den första metoden beskriver hur man kan gå till väga vid en integration av två it-system. Den andra tar upp viktiga principer för användbarhet att tänka på vid design av ett användargränssnitt.

I studien identifieras fem olika alternativ för integration av de båda systemen. De olika alternativen är framtagna med hänsyn till krav som ställdes ifrån både ABB och Prevas, ekonomiska faktorer och hur användarvänlig lösningen är. Av de fem alternativen väljs det som anses möta dessa kriterier bäst för skapandet av en prototyp av det integrerade systemet. Syftet med prototypen är att visa att systemen kan integreras med vald metod och den kommer att användas i marknadsföringssyfte. I prototypen integreras funktioner som operatörer använder för styrande av ugnar så att de kan styra ugnen helt via ABB:s system 800xA.

Slutsatserna dragna av studien är att det ur ett ekonomiskt perspektiv är bra att återanvända redan befintliga resurser så långt som möjligt. En resurs kan i detta fall vara både hårdvara i form av exempelvis en server eller mjukvara i form av exempelvis programkod. Återanvändning av resurser går dock till sist ut över användarvänligheten och en kompromiss mellan en billig lösning och en användarvänlig lösning måste göras. För framtiden rekommenderar författarna att en utförlig användbarhetsstudie utförs på operatörerna. Detta är viktigt för att på bästa sätt kunna underlätta arbetet för systemets användare genom en vältänkt design.

Ytterligare ett problem vid integration av mjukvara är att deras kommunikationer inte alltid är kompatibla. Med detta menas att information inte kan skickas från en komponent till en annan eftersom att de inte talar samma språk. En faktor som underlättar för integration av två it-system är därför att använda sig av kommunikationsstandarder. I industriella nätverk finns en sådan standard som heter OPC. Författarna rekommenderar därför att denna används i så stor utsträckning som möjligt för integrationen av de två systemen.

Förord

Med detta examensarbete avslutar vi våra studier på civilingenjörsprogrammet System i Teknik och Samhälle på Uppsala universitet. Examensarbetet utfördes på ABB och Prevas under vårterminen 2014.

Vi vill rikta ett stort tack till våra handledare: Lars-Göran Uhlander på ABB och Rickard Söderlund på Prevas för deras stöd och hjälp under arbetets gång. Vi vill även tacka Torbjörn Ottosson på ABB och Johan Bäckman på Prevas för att de gjort detta examensarbete möjligt. Tack även till alla medarbetare på både ABB och Prevas för trevliga fikastunder genom hela projektet.

Ett stort tack till vår ämnesgranskare Roland Bol för hjälp att hålla arbetet på rätt akademisk nivå och stöd under arbetets gång.

Kristian Bernström och Anders Näsman

Uppsala, augusti 2014

Innehållsförteckning

1 Inledning 3
1.1 Syfte och mål 4
1.2 Avgränsningar 4
1.3 Uppdelning 4
1.4 Rapportens struktur 5

2 Systembeskrivning 5
2.1 Industriella nätverk 5
2.2 Kommunikation i industriella nätverk 7
2.2.1 OPC 7
2.2.1.1 OPC Data Access 7
2.2.1.2 OPC Alarm & Events 8
2.2.2 Windows Communication Foundation 8
2.3 800xA 8
2.3.1 800xA struktur och kommunikation 9
2.3.1.1 Operatorsystem 10
2.3.1.2 Aspektserver 10
2.3.1.3 Konnektivitetsserver 10
2.3.1.4 Kontroller 11
2.3.2 Larmhantering i 800xA 11
2.4 Prevas Furnace Optimization Control System 12
2.4.1 Prevas FOCS struktur och kommunikation 13
2.4.1.1 Prevas FOCS server 13
2.4.1.2 Prevas FOCS klient 14
2.4.2 Larmhantering i Prevas FOCS 15

3 Systemintegrering 16
3.1 Integrationsprocessen 16
3.1.1 Steg 1: Definiera systemets funktion och ta fram en kravspecifikation 17
3.1.2 Steg 2: Identifiera vilka gemensamma åtgärder och resurser som behövs 19
3.1.3 Steg 3: Tilldela systemroller till resurser 20
3.1.3.1 Dataintegration 21
3.1.3.2 Teknisk integration 21
3.1.4 Steg 4: Implementera det integrerade systemet 21
3.2 Test, validering och verifikation 22
3.3 Designprinciper vid skapande av ett användargränssnitt 23

4 Framtagning av integrationsalternativ 25
4.1 Systemdefinition och framtagning av kravspecifikation 25
4.2 Resurser och roller 29
4.2.1 800xA:s resurser 29
4.2.2 Prevas FOCS resurser 29
4.2.3 Roller för det integrerade systemet 29
4.3 Systemroller och saknad funktionalitet 30
4.3.1 Dataintegration och teknisk integration av 800xA och Prevas FOCS 31
4.3.2 Val av integrationsmetod 35

5 Resultat och prototypbygge 36
5.1 Implementering av det integrerade systemet 37
5.1.1 Implementering av OPC-server 37
5.1.2 Modifikation av Prevas FOCS 38
5.1.3 Design av bilder i 800xA 38
5.2 Test och utvärdering av systemet 48

6 Diskussion 53

7 Slutsatser 54

8 Vidare forskning 55

9 Referenser 56
9.1 Böcker 56
9.2 Tidskriftsartiklar 56
9.3 Elektroniska källor 56
9.4 Rapporter och övriga dokument 57
9.5 Personlig kommunikation 57

Bilaga 1: Prevas FOCS funktioner 58

1 Inledning

ABB och Prevas har inlett ett samarbete för att marknadsföra och sälja ett optimeringssystem för styrning av ugnar i stålbranschen, kallat FOCS (Prevas 2010 [www]). Båda företagen ABB AB och Prevas AB är företag som specialiserat sig på att effektivisera industriers produktion. Prevas är ett konsultföretag som bland annat jobbar för att öka effektiviteten på industriers produktionsprocesser. En ugn som styrs av Prevas FOCS använder vanligtvis 5–20 procent mindre energi för att värma upp ämnet till rätt temperatur (ibid). ABB har en produkt som heter 800xA som är framtagen för att kunna reglera och övervaka hela industriers produktioner från början till slut, och på så sätt effektivisera produktionen (ABB 2014a [www]).

Energieffektiviseringen i svenska industrier har aldrig varit så viktig som den är i dagsläget. Energieffektiviseringsdirektivet som Europaparlamentet enades om i oktober 2012 innebär för Sveriges del att energiförbrukningen ska minskas med 20 procent fram till 2020. Detta i kombination med att miljöbalken kan ställa krav på att företag både ska ha kunskap om den bästa tekniken för energibesparing och använda densamma gör att det nu är viktigare än någonsin för industrier att effektivisera sin produktion (Naturvårdsverket 2014 [www]).

Energiåtgången i svenska industrier står för cirka en tredjedel av hela landets energiåtgång och järn- och stålindustrin är två av de industrier som drar mest energi av alla industrier. Detta betyder att det är just dessa industrier som måste minska sin energiåtgång mest för att Sverige ska klara av energieffektiviseringsdirektivet. Fleming (refererad i Vikström 2013, 41) från Svenskt Näringsliv menar att det är viktigt att företagen inte ser detta som att man ska skära ner på energianvändningen utan att man ska effektivisera den för att behålla sin konkurrenskraft gentemot lågkostnadsländerna, speciellt de utanför EU, som inte berörs av energieffektiviseringsdirektiven.

I dagsläget finns det inget sätt att använda både ABB:s system 800xA och Prevas FOCS utan att tvinga operatörerna att använda två olika system. Ur en arbetsmiljömässig synvinkel är detta inte önskvärt då det finns stora forsknings- och utvärderingsstudier som binder hälsobesvär till dåligt utformade IT-system. Vicente (2006) tar till exempel upp vikten av att utforma system utifrån användarens både fysiska och psykologiska förmågor. Med större fokus på den mänskliga faktorn i samspel med det tekniska systemet som en helhet kan man se till att operatörerna kan utföra sitt jobb på ett mer effektivt sätt. Karasek och Theorell (1990) publicerade på 1970-talet en modell för hur anställda kan uppleva stress på sin arbetsplats beroende på två faktorer, kontroll och krav. En anställd som upplever att den har en viss nivå av krav på sig måste även nå till en viss nivå av kontroll. Har den anställda en högre nivå av krav än vad som stöds skapas det negativ spänning som på lång sikt kan leda till psykiska och fysiska påfrestningar. Deras bok beskriver också att så länge en anställd har bra kontroll över sin tillvaro störs den inte alls på samma sätt av höga krav utan istället för att få negativa spänningar blir den mer produktiv. Modellen är senare vidareutvecklad av House (1981) genom att ta med socialt stöd som en tredje faktor. Med denna faktor menar House att man kan kompensera en del av kontrollbehovet med ett socialt stöd, men om detta stöd inte heller är tillräcklig tillfredställande riskerar personerna ifråga allvarliga hälsorisker som till exempel utbrändhet.

Sandblad (2012) menar att nya eller förändrade IT-stöd så gott som alltid upplevs som en ökning av krav. Detta kan förklaras med att det oftast är en specifik anledning varför

det nya IT-stödet levereras som till exempel minskad energiförbrukning eller ökad produktivitet. Då är det också viktigt att samtidigt öka stödet för användarna så inte uppgraderingen känns som en ytterligare stressfaktor.

I dagens samhälle där krav för effektivisering trycker på av både konkurrens- och miljömässiga anledningar kan det vara enkelt att glömma bort användarna av ett IT-stöd. Påföljderna av att negligera sina anställda i produktionsindustri är väl dokumenterade redan sedan Henry Ford införde fordismen i sina verkstäder. Bortfall och sjukskrivningar leder till försämrad kvalitet och högre krav på resterande anställda – en negativt nedåtgående spiral. Detta examensarbete kommer fokusera på och påvisa möjligheter på hur man kan effektivisera produktion genom integration samtidigt som man höjer stödet för operatörerna med relativt enkla medel.

1.1 Syfte och mål

Syftet med examensarbetet är att utreda möjligheter för integrering av Prevas ugnsreglersystem FOCS i ABB:s styrsystem 800xA. Olika alternativ analyseras utifrån teknisk genomförbarhet och hänsyn tas till ekonomiska faktorer och marknadsläge. Det alternativ som efter utredningen bedöms som bäst implementeras i en prototyp där de två systemen kan samspela. I implementeringen av prototypen tas hänsyn till användbarhetsprinciper för att förbättra användbarheten i det integrerade systemet. Målet är att underlätta konstruktions- och projektgenomförandet och att utredningen ska kunna användas som mall inför framtida integrationsproblem. Målen kan således sammanfattas i tre övergripande punkter:

- Utredning av olika alternativ till integrering av de båda systemen.
- Skapande av prototyp utifrån vald integreringsmetod.
- Förbättra användbarheten där möjlighet finns.

1.2 Avgränsningar

Prevas FOCS är ett stort system som riktar sig till många olika användare med behov av olika information. Vi kommer dock i den här rapporten att avgränsa oss till operatörers informationsbehov och integreringen kommer därför att rikta sig till de funktioner som är relevanta för just operatörer.

1.3 Uppdelning

Eftersom att rapporten har två författare har vissa uppdelningar gjorts mellan oss. Kristian har i prototypbygget haft större ansvar för ABB:s system 800xA. Det innefattar design och byggande av bilder och att se till att alla relevanta OPC-signaler kan kopplas till dem. Anders har haft större ansvar för modifikation av Prevas FOCS. I detta innefattas omprogrammering så att relevanta funktioner även skickas via OPC, databasuppdateringar och implementering av OPC-server. Författarna har tillsammans studerat litteratur inom området för att få en uppfattning om systemen och miljön där de verkar. Författarna har även tagit fram krav för systemintegreringen och skrivit rapporten tillsammans.

1.4 Rapportens struktur

I kapitel 2 beskrivs vad ett industriellt nätverk är för att ge en förståelse för miljön där de båda systemen verkar. I kapitlet beskrivs även ABB:s styrsystem 800xA och Prevas styrsystem för ugnar FOCS ingående.

I kapitel 3 ges teori och metoder för systemintegrering. Här presenteras ett tillvägagångssätt för systemintegrering som används i studiens resultat och hur ett system kan testas och valideras. Även teori kring användbarhet presenteras som senare används vid design av operatörsbilder i 800xA.

I kapitel 4 beskrivs hur framtagning av olika integrationsalternativ har gått till och de olika integrationsmöjligheterna diskuteras. Här presenteras även den metod som vi bedömer som bäst. Tillvägagångssättet bygger på metoden som presenteras i teori- och metodkapitlet.

Kapitel 5 beskriver det andra målet med denna uppsats; skapandet av en prototyp till integrering av de båda systemen. Här beskrivs de olika modifikationerna av systemen samt vilka resurser som har behövts läggas till. Utvärdering av det färdiga systemet görs även.

I kapitel 6 diskuteras den använda metoden och resultatet av studien.

I kapitel 7 dras slutsatser från studien och rekommendationer för systemintegrering ges.

Kapitel 8 sammanfattar studiens viktigaste resultat och ger förslag på områden att fokusera på för vidare forskning.

2 Systembeskrivning

I detta kapitel finns inledningsvis en generell beskrivning av vad ett industriellt nätverk är och hur det fungerar för att ge en bakgrund till området där de två systemen verkar. Därefter beskrivs hur kommunikationen går till inom det industriella nätverket och mellan dess komponenter. Slutligen beskrivs de två systemen som ska integreras: ABB:s system 800xA och Prevas system FOCS mer detaljerat.

2.1 Industriella nätverk

Ett industriellt kontrollnätverk är ett system av sammankopplade enheter för att övervaka och styra fysisk utrustning i industriella miljöer. Med uppfinnandet av mikroprocessorer och integrerade kretsar har styrningen av tillverknings- och processanläggningar gått från att vara helt mekanisk till att idag skötas till stor del digitalt. Denna flytt till det digitala har lett till att kommunikationsprotokoll mellan digitala kontrollers har behövts tas fram och standardiseras. Industriell utrustning befinner sig idag i en nätverksmiljö som är lik traditionella nätverk, men med andra behov. Skillnaden på ett industriellt nätverk och ett vanligt kommersiellt nätverk är att ett industriellt nätverk är sammankopplat med fysisk utrustning som på något sätt övervakar och styr verkliga processer och villkor (Galloway & Hancke 2013:860).

Industriella nätverk finns i många tillverknings- och processindustrier såsom elproduktion, mat- och dryckesberedning, transporter, vattendistribution, vattenrening, kemisk förädling (till exempel oljeraffinaderi) och inom stålindustrin. I de flesta fall där maskiner måste styras finns ett industriellt nätverk installerat i någon form. Arkitekturen

i ett industriellt nätverk skiljer sig också från i ett kommersiellt på så sätt att det krävs kommunikation på fler nivåer. Ofta är instrument ihopkopplade med kontrollers på en nivå, kontrollers sammankopplade med varandra på nästa, överföring av information från kontrollers till ett Human-Machine Interface (HMI) på nivån över detta och slutligen ett nätverk för datainsamling och extern kommunikation. Olika typer av protokoll för kommunikation och olika fysisk media finns ofta på de olika nivåerna, vilket kräver gatewayutrustning för att facilitera kommunikation. Krav på säkerhet och snabbhet i dataöverföring tenderar också att vara högre i ett industriellt nätverk än i ett kommersiellt nätverk. Detta beror på att fysisk utrustning är kopplad till nätverket, vilket innebär att fel kan få större konsekvenser. Fel i industriella processer kan i värsta fall leda till katastrofer om till exempel utrustningen används för att styra ett kärnkraftverk eller inom andra industrier som innefattar kritiska processer. Snabbhet i respons krävs för att säkert kunna styra tidskritiska industriella processer, där information exempelvis kan komma från rörelsedetektorer eller annan tidskritisk utrustning (Galloway & Hancke 2013:861–862).

En viktig komponent i industriella nätverk är Programmable Logic Controllers (PLC:er). Dessa är specialiserade, datorbaserade elektroniska komponenter och är centrala i ett industriellt nätverk. De skapades för att ersätta reläkretsar och för att vara enkla att omprogrammera. De är även mindre i storlek och billigare att producera. Moderna PLC:er kan reglera signaler genom exempelvis PID (en proportionell del, en integreringsdel och en deriveringsdel) och kan utföra både binär och analog input och output. De består generellt av en processor, en strömkälla och utbytbara input/output-moduler och kommunikationsmoduler. Detta medför att det lätt kan anpassas efter behovet som en specifik industri och ett specifikt företag har. För att hantera PLC:ers behov har ett antal standarder ifråga om design och programmeringsspråk skapats. Den vanligaste är International Electrotechnical Commission (IEC) som definierar fem olika programmeringsspråk för PLC:er: Ladder Diagram, Sequential Function Chart, Function Block Diagram, Structured Text och Instruction List. I många fall finns i moderna utvecklingsmiljöer grafiska gränssnitt för att programmera PLC:er (Galloway & Hancke 2013:863).

Industriella nätverk har delats upp i två olika klasser: level 1-system och level 2-system. Level 1-systemen har hand om den data som är mest processkritisk, det vill säga de första nivåerna i det industriella nätverket som nämnts i stycket ovan. Informationen som går genom ett level 1-system är oftast rådata som till exempel temperatur eller hastighet som är uppmätta av givare och skickade genom kontrollers vidare till ett HMI, detta för att användare både ska kunna övervaka vad som händer med sin utrustning och att de ska kunna skicka förändrade börvärden till maskinerna. Level 2-system är däremot processoptimerande och använder oftast flera olika parametrar från olika nivåer och delar av processen för att kunna skapa en helhet i industrins produktion. Nackdelen med ett level 2-system är liknande de i ett kommersiellt nätverk, nämligen att systemen oftast körs i en Windows-baserad miljö och risken för att systemet kraschar är högre än i ett level 1-system som ska ha en drifttid på flera år utan avbrott. (Uhlander 2014).

Datapaket som skickas i ett level 1-system i ett industriellt nätverk är ofta mindre än de som skickas i level 2-system och kommersiella nätverk, då de ofta i princip bara behöver innefatta ett uppdaterat mätvärde från något instrument. Det är dock viktigare med tidsangivelser och i vilken ordning händelser sker, vilket kräver synkroniserade klockor och tidsangivelser inbyggda i protokoll på låga nivåer. Dessutom kräver den

fysiska miljön där utrustningen är installerad att utrustningen tål hårda miljöer, där exempelvis värme eller fukt förekommer (Galloway & Hancke 2013:862).

2.2 Kommunikation i industriella nätverk

Kommunikation i industriella nätverk sker på många olika nivåer, med flera olika kommunikationsprotokoll med olika komplexitet och krav på bland annat säkerhet och hastighet. Ett av de viktigaste kommunikationsprotokollen i ett industriellt nätverk är idag OPC, vilket är en relativt ny standard som utgör grunden för både 800xA och Prevas FOCS kommunikation med PLC:er. Prevas FOCS, som är ett level 2-system, använder sig även av Microsofts .NET Framework och WCF som även kort presenteras i detta kapitel.

2.2.1 OPC

Det är vanligt att använda sig av Windows-baserade PCs för att styra och visualisera i industriella automationssystem. Det är därför viktigt att kunna standardisera automationsmjukvara så att olika system, protokoll och gränssnitt kan kommunicera med varandra oberoende av tillverkare. Detta problem löstes när en grupp återförsäljare av HMI och SCADA gick samman för att ta fram en Plug&Play-standard för drivrutiner för att standardisera åtkomsten av automationsdata i Windows-baserade system. Denna standard kom att kallas OPC Data Access och hålls uppdaterad av OPC Foundation, en ideell förening där de flesta företagen i automationsbranschen är medlemmar (Mahnke et al. 2009:1). OPC stod ursprungligen för Object Linking and Embedding (OLE) for Process Control, men betydelsen har ändrats på grund av att det idag används bortom processkontroll (OPC Foundation 2014a [www]). Sedan 2011 står förkortningen enligt OPC Foundation för Open Productivity & Connectivity och det finns idag sju standarder, färdiga och under utveckling (ibid). Två av dessa standarder är för vår del relevanta att mer utförligt beskriva, Data Access (DA) och Alarm & Events (A&E). Först kommer en förklaring av OPC:s struktur.

OPC använder en server-klient struktur för informationsutbyte. En OPC-server gör källan till processinformation tillgänglig via sitt gränssnitt och en OPC-klient ansluter till den och kan komma åt och använda dess data. Applikationer som använder OPC kan vara både klient och server för att både konsumera och skicka vidare data. Klassisk OPC använder sig av Microsofts teknologier Component Object Model (COM) och Distributed COM (DCOM) vilket är smidigt för att det finns på alla Windows-baserade enheter. Nackdelar är istället att OPC blir beroende av Windowsplattformen och att DCOM är svår att konfigurera, har långa timeouts och inte kan användas för internetkommunikation (Mahnke et al. 2009:3–4).

2.2.1.1 OPC Data Access

Gränssnittet för OPC Data Access ger tillgång till läsning, skrivning och övervakning av variabler innehållandes aktuella processdata. Dess främsta uppgift är att föra över realtidsdata från PLC:er, DCS och annan kontrollutrustning till HMIs och andra displayklienter. Data Access är OPC:s viktigaste standard och används i de flesta produkter som använder sig av OPC. En OPC DA klient väljer de variabler (OPC items) som den vill läsa, skriva eller övervaka i servern. OPC-klienten skapar sedan en uppkoppling till servern genom att skapa ett OPCServer-objekt. Servern erbjuder metoder för att navigera genom adressutrymmen för att hitta items och dess egenskaper såsom datatyp och åtkomsträttigheter. För att komma åt data grupperar klienten OPC

items med likadana egenskaper såsom uppdateringstid i ett OPCGroup-objekt. När items är grupperade kan de läsas och skrivas av klienten. Genom att övervaka dessa grupperingar med viktiga items kan klienten cykliskt läsa av ändringar i värden med en förinställd tidsram. Servern skickar då de uppdaterade värdena till klienten efter varje sådan cykel. OPC tillhandahåller realtidsdata som kan hindras av att en enhet temporärt störs. Genom att sätta en tidsstämpel och kvalitet på data (bra, dålig eller osäker) kan detta hanteras (Mahnke et al. 2009:4–5). Data lagras i en så kallad tagg, som kan representera en output från en PLC eller ett minnesregister i en enhet. Detta ger realtidsaccess till datavärdena (Control Global 2010 [www]).

2.2.1.2 OPC Alarm & Events

Gränssnittet för OPC Alarm & Events möjliggör mottagandet av händelser och alarm. Händelser är meddelanden som informerar klienten att en händelse har skett och alarm meddelar klienten om en ändring i tillstånd i processen. Detta tillstånd kan exempelvis vara att nivån i en tank har sjunkit under en viss tillåten nivå. Alarmen kan ställas in så att de måste bekräftas av användaren. För att få meddelanden kopplar OPC A&E klienten till servern och prenumererar på meddelanden som servern då skickar. OPC-klienten kopplar upp genom att skapa ett OPCEventServer-objekt i A&E servern och genererar sedan en OPCEventSubscription för att kunna ta emot händelser. Till skillnad från OPC DA efterfrågas inga specifika värden, men alla processhändelser skickas och det går att ställa in mängden händelser som tas emot genom att filtrera på exempelvis händelsetyp, prioritet eller händelsekälla (Mahnke et al. 2009:5–6).

2.2.2 Windows Communication Foundation

Microsofts .NET Framework är teknologi för att designa nästa generations applikationer och XML Web services (Microsoft 2014a [www]). Med version 3.0 av .NET introducerades ett API designat specifikt för att bygga distribuerade system: Windows Communication Foundation (WCF). Med WCF blir det möjligt att kommunicera med andra distribuerade API som använts tidigare (exempelvis DCOM) genom en förenad objektorienterad programmeringsmodell (Troelsen 2012:985). WCF använder meddelandebaserad kommunikation mellan server och klient där servern väntar på att en klient initierar kommunikation och därefter svarar (Microsoft 2014b [www]). Ytterligare fördelar med WCF är att data kan skickas över flera protokoll. För in-house applikationer kan TCP användas och för att kommunicera över internet kan exempelvis HTTP användas (Troelsen 2012:992).

I en Windows-baserad in-housemiljö är det fördelaktigt att använda sig av TCP som protokoll eftersom att det ger högre prestanda än web service-protokoll. Det är också en mer pålitlig överföring som erbjuder säker kommunikation (Troelsen 2012:1000–1001).

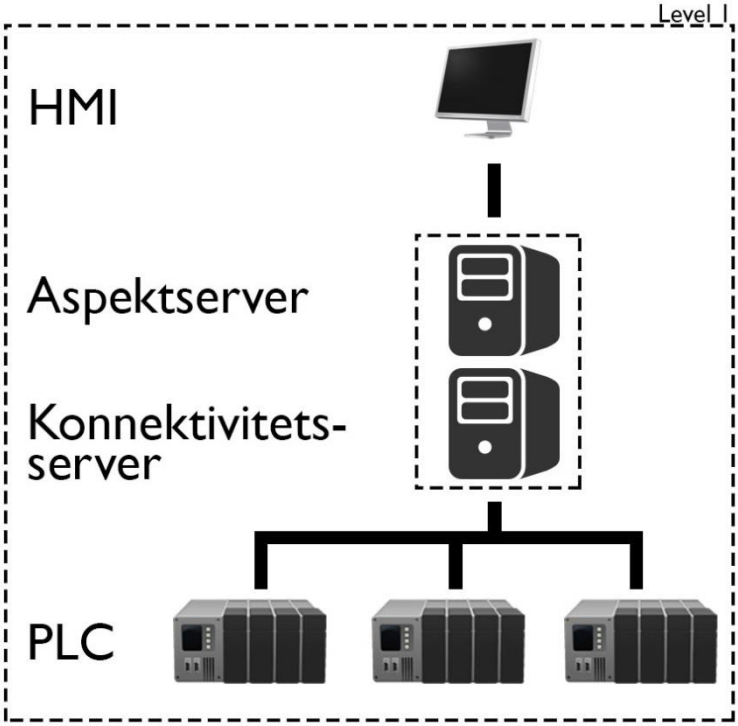
2.3 800xA

800xA är en industriell lösning som är skapad av ABB. xA står för Extended Automation och syftar på hur systemet är designat för att integrera flera delsystem till ett större helhetssystem. En industri består oftast av flera oberoende mindre stationer som tillsammans tillverkar en produkt. Ett exempel på en sådan industri är stålindustrin som har flera stationer som produkten måste gå igenom innan produkten får de egenskaperna som kunderna förväntar sig. Två stationer som kan finnas i en stålindustri är omvärmningsugn och stålvals. Först ska ämnet värmas till rätt temperatur för att sen valsas till rätt storlek och tills ämnet har fått de egenskaper som en kund har beställt.

Traditionellt styrs båda stationerna oberoende av varandra och det kan leda till slöseri av resurser om tillexempel maskinoperatören på omvärmningsugnen börjar värma ämnet så att det får rätt temperatur innan valsen är klar med sitt program. Då förbrukar det onödig mycket energi eftersom ämnet måste sitta i ugnen för att behålla rätt temperatur tills valsoperatören är redo att börja valsa nästa ämne. Med 800xA är tanken att flera stationer ska integreras och operatörerna får en helhetssyn över produktionen och därför kunna planera körningarna för att minimera flaskhalsar och resursslöseri i produktionen (ABB 2013).

2.3.1 800xA struktur och kommunikation

800xA är ett level 1-system som erbjuder en integrerad helhetslösning för industrier. Detta innebär att industrierna helt kan skötas med hjälp av produkter från ABB, men för mer avancerad funktionalitet, exempelvis optimering, krävs annan kompletterande teknik. 800xA kan då istället integreras med produkter från andra tillverkare genom användandet av standardiserade protokoll. Systemet 800xA kan delas upp i fyra olika delsystem, operatorsystemet som visar informationen för operatörerna, aspektservern som lagrar bilderna som visas i arbetsstationerna, konnektivitetsservern som har hand om informationsflödet från PLC:er och själva PLC:erna som styr de olika stationerna i industrin. En överskådlig bild av systemet syns i Figur 1. I figuren syns bara en inkopplad HMI men i 800xA kan man koppla in fler efter tycke. Även aspekt- och konnektivitetsservrar kan vara fler än en för att skapa redundans i systemet (Uhlander 2014).



Figur 1: 800xA:s struktur.

2.3.1.1 Operatorsystem

Operatorsystemet är delen av systemet som har hand om Human-Machine Interface (HMI), det vill säga grafiken som operatörerna använder för att övervaka och utföra ändringar i systemet. 800xA har flera olika typer av HMI och den viktigaste ur operatörernas synvinkel är ”extended operator workplace” (EXO) som är en station som är designad för att ha total översikt över vissa delar eller hela produktionen beroende på industrins storlek. EXO har ofta upp till nio skärmar av olika storlekar för att operatörerna hela tiden ska kunna ha översikt på de olika stationerna i produktionen. Detta istället för att ha en skärm och behöva byta vy beroende på vilken del man vill övervaka. I Figur 2 nedan visas hur en EXO-arbetsstation kan se ut. Man kan även ha mindre HMI som ett komplement utanför kontrollrummet eller om man vill ha en arbetsstation mer processnära (Uhlander 2014).

Figur 2: 800xA:s arbetsstation EXO.

2.3.1.2 Aspektserver

För att visa informationen på arbetsstationerna används bilder som automatiskt uppdaterar värden som den får från en konnektivitetsserver. Bilderna som visas i operatorsystemet är sparade som objekt och är skapade och lagrade i aspektservern. Det går även att ställa in hur ofta bilderna ska uppdatera enskilda värden. Alla bilderna är designade för den specifika industrin där systemet är installerat och efter en designmall för att underlätta arbetet för operatörerna (Uhlander 2014).

2.3.1.3 Konnektivitetsserver

Informationen som visas i bilderna får sina värden från konnektivitetsservern som sparar ner och omvandlar inkommande signaler från kontrollers till användbar data. Konnektivitetsservern har också hand om att omvandla signaler från operatorsystemet till kontrollern vid till exempel temperatursjustering (Uhlander 2014).
Konnektivitetsservern blev ett krav för att kunna integrera processtyrningar 1996 då standarden ”object linking and embedding for process control” (OPC) etablerades. OPC är en standard för kommunikation som kontrollers använder sig av (OPC Foundation 2014b [www]). Detta betyder att man inte längre behövde använda sig av samma leverantör av kontrollers som arbetsstation vilket fallet ofta tidigare var.
Konnektivitetsservern är egentligen ABB:s variant på en OPC-server men med lite mer egenskaper, den tar till exempel hand om all kommunikation från alla olika kontrollers som ingår i systemet och strukturerar upp dem, vilket inte alla OPC-servrar gör. Strukturen är en trädstruktur så att man lättare ska kunna särskilja olika kontrollers från varandra, vilket underlättar i industrier där man till exempel har flera likadana produktionslinjer parallellt (Uhlander 2014).

2.3.1.4 Kontroller

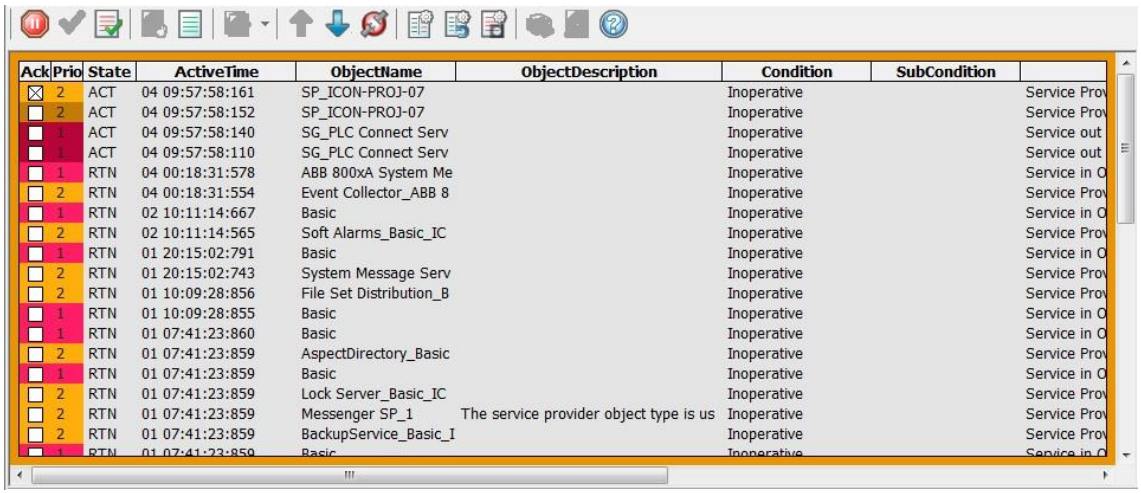
För att kunna skicka och ta emot signaler från maskiner i produktionslinjen behöver man en kontroller. En kontroller är en processnära PLC som är designad och programmerad för att snabbt kunna hantera till exempel börvärden och reglerloopar men har nackdelen att inte kunna hantera stora mängder data. En av ABB:s kontrollers som kan användas ihop med 800xA heter AC 800M och är en avancerad variant av en sådan PLC. Men även kontrollers tillverkade av konkurrerande företag kan användas efter införandet av OPC som standard. ABB tillhandahåller även mer avancerade kontrollers till sina kunder, en sådan var ABB:s FOCS (mer om FOCS i nästa kapitel). ABB har i dagsläget avslutat sin forskning med ABB FOCS. Detta innebär att kunder som vill effektivisera denna del av sin produktion måste vända sig till en annan leverantör. En annan anledning varför systemet är begränsat är att det endast hanterar funktionalitet på level 1-nivå och därför har begränsat minne vilket leder till begränsade möjligheter för optimering. Eftersom effektivisering är en stor del av dagens industrier finns det dock fortfarande en efterfrågan på effektiva ugnsregleringssystem (Uhlander 2014).

2.3.2 Larmhantering i 800xA

Det blir allt färre operatörer som ska hantera allt större processer i dagens industrier vilket gör det viktigt att kunna hantera larm som inträffar på ett effektivt och säkert sätt. ABB:s styrsystem 800xA är designat för att underlätta för operatörer genom att styra operatörens fokus till händelser som kräver djupare analys och ge den information som behövs för att operatören ska kunna hantera situationen. Det finns en rad olika funktioner som ska underlätta för operatören. Några viktiga sådana är (ABB 2012):

- Färdiga analysverktyg som exempelvis hjälper operatörer att se de vanligaste larmen, de alarmen som är aktiva under längst period och hur lång tid olika larm tar att kvittera över tid.
- Möjlighet att göra larmlistor som är specifika för den del av en industri som en viss operatör är intresserad av så att operatören får relevant information.
- Möjlighet för operatörer att kommentera larm och visa aktuella värden så operatören lätt kan se allvarligheten av ett larm. Exempelvis kan larmet gälla att nivån i en tank har blivit för låg och då kan operatören se om den är bara lite för låg eller mycket för låg.
- Möjlighet att göra larm som är väntade osynliga vid exempelvis reparationer eller kontrollerade avbrott i produktionen.
- Möjlighet att lägga ett larm ”på hyllan” då andra, mer prioriterade, larm finns. Larmet läggs då i en annan larmlista som operatören kan ta tag i när de mest kritiska larmen är undanröjda.
- Möjlighet att gruppera larm som hör ihop och kräver liknande respons av operatören. Syftet är att reducera antalet larm samtidigt som de blir mer informativa med förhoppningen att operatörens arbete blir mer effektivt och säkrare.

I Figur 3 visas ett exempel på hur en larmlista kan se ut i 800xA.



Ack	Prio	State	ActiveTime	ObjectName	ObjectDescription	Condition	SubCondition	
☒	2	ACT	04 09:57:58:161	SP_ICON-PROJ-07		Inoperative		Service Prov
☐	2	ACT	04 09:57:58:152	SP_ICON-PROJ-07		Inoperative		Service Prov
☐	1	ACT	04 09:57:58:140	SG_PLC Connect Serv		Inoperative		Service out
☐	1	ACT	04 09:57:58:110	SG_PLC Connect Serv		Inoperative		Service out
☐	1	RTN	04 00:18:31:578	ABB 800xA System Me		Inoperative		Service in O
☐	2	RTN	04 00:18:31:554	Event Collector_ABB 8		Inoperative		Service Prov
☐	1	RTN	02 10:11:14:667	Basic		Inoperative		Service in O
☐	2	RTN	02 10:11:14:565	Soft Alarms_Basic_IC		Inoperative		Service Prov
☐	1	RTN	01 20:15:02:791	Basic		Inoperative		Service in O
☐	2	RTN	01 20:15:02:743	System Message Serv		Inoperative		Service Prov
☐	2	RTN	01 10:09:28:856	File Set Distribution_B		Inoperative		Service Prov
☐	1	RTN	01 10:09:28:855	Basic		Inoperative		Service in O
☐	1	RTN	01 07:41:23:860	Basic		Inoperative		Service in O
☐	2	RTN	01 07:41:23:859	AspectDirectory_Basic		Inoperative		Service Prov
☐	1	RTN	01 07:41:23:859	Basic		Inoperative		Service in O
☐	2	RTN	01 07:41:23:859	Lock Server_Basic_IC		Inoperative		Service Prov
☐	2	RTN	01 07:41:23:859	Messenger SP_1	The service provider object type is us	Inoperative		Service Prov
☐	2	RTN	01 07:41:23:859	BackupService_Basic_I		Inoperative		Service Prov
☐	1	RTN	01 07:41:23:859	Basic		Inoperative		Service in O

Figur 3: 800xA:s larmlista.

Larmen som inkommer till 800xA kommer från en eller flera OPC A&E-servrar. Till exempel har ABB:s PLC AC 800M en OPC A&E-server som kan ställas in att skicka larm. Det finns även möjlighet att koppla upp en tredjeparts OPC A&E-server till systemet.

På Automation Scandinavia, en automationsmässa anordnad av ABB, demonstrerades hur larmhanteringen kan se ut i ett stort 800xA-system med EXO-arbetsstationer. När produktionen fungerar normalt sitter operatören vid sina skärmar och ett grönt ljus lyser upp rummet för att indikera att allting är normalt. När ett larm inkommer blir rummet rött och operatören hör en varningssignal genom en ljuddusch riktad så att enbart operatören hör den. Bordet som operatören sitter vid höjs och skärmarna vinklas så att operatören tvingas till en mer alert stående position tills larmet är kvitterat. Operatören kan även, via systemet, få hjälp direkt av en specialist som sitter på annan plats. Det finns även möjlighet att via video få en överblick av larmets ursprung så operatören kan se om något har inträffat på plats där komponenten som avgett larmet finns. Hur det ser ut i en specifik industri beror på vilken konfiguration av 800xA som finns installerad, i en normalstor industri behövs dock vanligtvis inte all denna funktionalitet (ABB 2014b).

2.4 Prevas Furnace Optimization Control System

Prevas är ett Västeråsbaserat företag som har en industri IT-avdelning som utvecklar sitt eget reglersystem Prevas FOCS. Industrier sparar vanligtvis 5–20% energi på att uppgradera från ett gammalt till ett nytt reglersystem, detta leder också ofta till ökad produktivitet och i sin tur till bättre resultat för företagen (Prevas 2014 [www]).

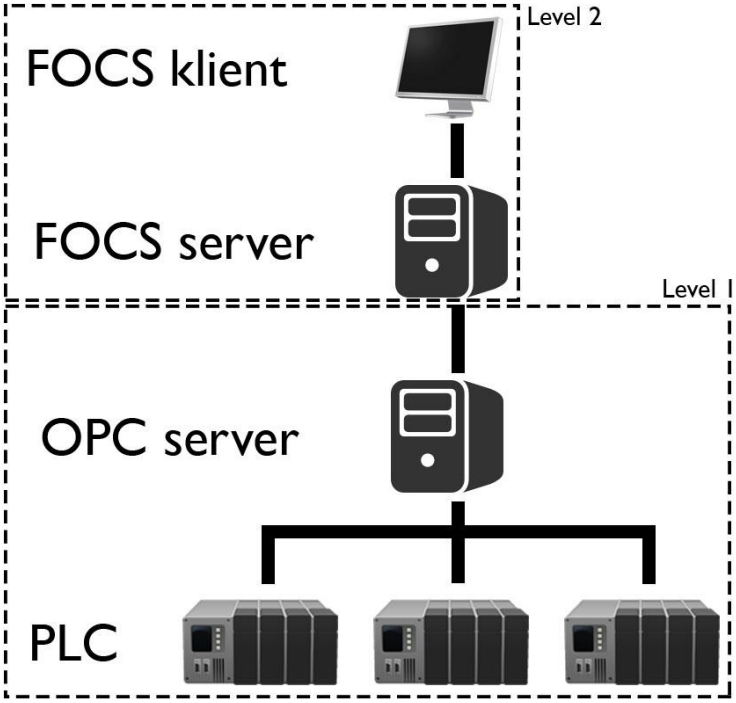
Furnace Optimization Control System (FOCS) bygger på ett resultat och utveckling från ett forskningsprojekt som Swerea MEFOS har utfört. Forskningsresultatet är en matematisk modell för hur olika typer av ämnen reagerar på temperatur. Med hjälp av denna modell kan man designa ett reglersystem för att värma upp ämnet mer effektivt till temperaturer som tillåter efterarbete (Söderlund 2014).

En omvärmningsugn är en ugn som ska värma upp stål, ofta i form av slabs, billets eller blooms (gjutna rätblock av olika form som har samlingsnamnet ämne) till rätt temperatur för att kunna valsas ut till en ny form, till exempel från ämne till plåt. För att en vals ska kunna hantera ämnen krävs det typiskt en temperatur mellan 1150 °C och

1270 °C och beroende på hur man värmer upp ämnet får de olika egenskaper för att passa olika tillämpningar. Det är viktigt att man inte värmer upp ett ämne med för hög värme, detta för att man då riskerar antingen att bränna utsidan vilket drar ner på kvalitén eller att kärnan inte får rätt temperatur. För att förhindra att detta händer använder man sig av reglersystem som bygger på just Swerea MEFOS matematiska modell och beroende på hur bra reglersystemet är optimerat kan man spara mycket energi i användningen av ugnen (Söderlund 2014).

2.4.1 Prevas FOCS struktur och kommunikation

Prevas FOCS är ett level 2-system som använder en server–klient-struktur baserad på Windows Communication Foundation (WCF). Eftersom systemet är level 2 kan avancerad funktionalitet, i form av exempelvis optimering av processer och beräkningar, utföras av systemet. Systemet kopplas till industrins befintliga OPC-server som i sin tur är kopplad till PLC:er. På så sätt fås informationen som systemet sedan kan reglera och optimera (Söderlund 2014). Strukturen åskådliggörs i Figur 4 nedan.



Figur 4: Prevas FOCS struktur.

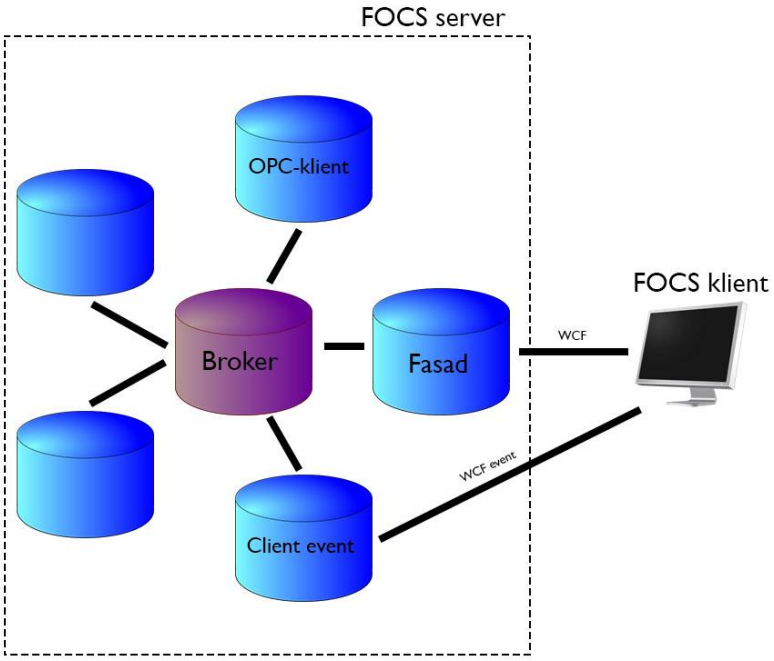
2.4.1.1 Prevas FOCS server

Prevas FOCS server hanterar all funktionalitet såsom styrningen av ugnarna, beräkningar och kommunikation med andra system. Servern har även en OPC-klient som kan läsa av processinformation från en OPC-server. På så sätt får Prevas FOCS server data som regleras och kan skicka data till Prevas FOCS klient. Den senaste versionen av Prevas FOCS är uppbyggd runt en medlare (kallad messagebroker) som all information går genom (Söderlund 2014).

Genom att använda en broker-integrationsarkitektur kan man integrera två eller fler system utan att förändra mycket i deras grundutförande. Arkitekturen går ut på att ha ett brokersystem som en central knutpunkt för att binda ihop flertalet system som annars

inte skulle kunna kommunicera med varandra utan integration, se Figur 5. Detta betyder att systemen aldrig kommer att kommunicera direkt med varandra, utan kommer att använda sig av brokersystemet för att förmedla information. För att det ska fungera måste brokern vara designad på ett sådant sätt att den kan kommunicera med båda systemen oberoende vilket språk eller arkitektur de använder. När brokern har mottagit en förfrågan eller information från ett system kommer den att agera som en tolk. Det vill säga att den tar emot information från ett system och analyserar vad det är för typ av information och vilken destination som informationen är ämnad för. När allt det är klart skickar brokern vidare informationen till sina slutmål. Ett brokersystem fungerar bra när det är information som ska föras mellan två system. Desto djupare integration som krävs av systemet ju sämre kommer ett brokersystem fungera (Isazadeh 2004:60).

I detta fall används brokersystemet för att skicka information mellan olika delar inom Prevas FOCS. Medlaren kan alltså skicka information mellan olika moduler till rätt del av programmet. I och med denna struktur finns det möjlighet att både skicka och ta emot OPC-signaler på ett relativt enkelt sätt (Söderlund 2014). Medlarstrukturen kan ses i Figur 5.



Figur 5: Strukturen kring en medlare i Prevas FOCS server.

Som kan ses i Figur 5 är varje modul (de blå cylindrarna) kopplade till en medlare som översätter och skickar meddelanden till andra moduler eller system. OPC-klientmodulen kan till exempel kopplas till en extern OPC-server för mottagande av information från exempelvis PLC:er. Via medlaren kan den informationen skickas vidare till fasaden som genom WCF skickar meddelanden till klienten som då visar informationen grafiskt (Söderlund 2014).

2.4.1.2 Prevas FOCS klient

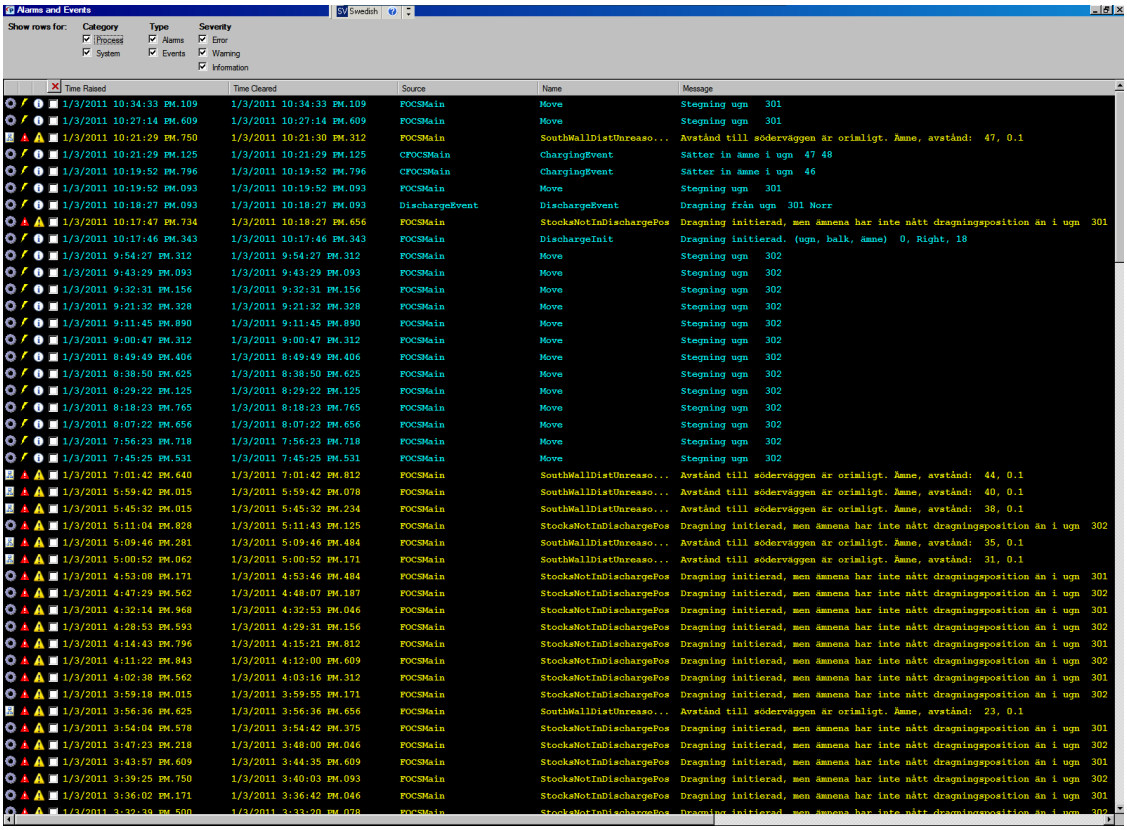
Prevas FOCS klient hanterar den grafiska biten av programmet och har en rad olika funktioner inbyggda i mjukvaran. Funktionerna kan delas upp i olika områden och kan användas beroende på vilken typ av användare som är inloggad på systemet. Det finns fyra olika användare som är förprogrammerade i Prevas FOCS, dessa är gäst,

processingenjör, operatör och administratör. Detta examensarbete fokuserar i första hand på operatörens funktioner då det är de bilderna som visas i kontrollrummet. Operatörerna behöver inte ha tillgång till alla olika funktioner som finns i Prevas FOCS, de kan till exempel inte ändra inställningar som inte berör dem och de kan inte heller mata in nya materiallistor. Operatörernas funktion är att överse operationen och för detta är de två viktigaste bilderna ugnsbilden och taktstyrningen (Söderlund 2014).

Klienten förser operatörerna med bilder som uppdateras genom anrop till Prevas FOCS server. Det är dessa bilder som gör det möjligt för operatören att styra operationerna, som att till exempel ändra temperatur i ugnen. Klienten kan också ta emot data genom att Prevas FOCS server skickar ett event till Prevas FOCS klient, som då hämtar relevant data från servern. Till skillnad från annan data skickas dock larm direkt med eventet från servern (Söderlund 2014).

2.4.2 Larmhantering i Prevas FOCS

I Prevas FOCS hanteras larm internt, det vill säga utan koppling till OPC A&E. Detta möjliggörs av att systemet bygger på Microsofts .NET Framework där mer komplexa strukturer kan byggas och sedan skickas till klienten (Söderlund 2014). Operatörerna får här, likt 800xA, en lista över aktuella larm som operatören måste reagera på. Operatören kan ta upp en enklare version av larmlistan eller den fullständiga listan genom att klicka på en knapp längst ner till vänster i alla Prevas FOCS klientfönster. Listan kan sorteras på kategori (process eller system), typ (alarm eller händelse) och allvarlighetsgrad (fel, varning eller information). Dessa visas även grafiskt, genom olika symboler, så att operatören lätt ser vad som har hänt. Vidare kan ses om larmet är kvitterat eller inte och kortfattad information om larmet visas. Om operatören dubbelklickar på ett larm fås ytterligare information om det. Alla alarm och händelser sparas även i databas och kan sökas på i efterhand i en alarm- och händelserapport (Prevas 2013). FOCS larmlista visas nedan i Figur 6.



Alarms and Events

Show rows for: Category: Process, System; Type: Alarms, Events; Severity: Error, Warning, Information

Time Raised	Time Cleared	Source	Name	Message
1/3/2011 10:34:33 PM.109	1/3/2011 10:34:33 PM.109	FOCSMain	Move	Stegning ugn 301
1/3/2011 10:27:14 PM.609	1/3/2011 10:27:14 PM.609	FOCSMain	Move	Stegning ugn 301
1/3/2011 10:21:29 PM.750	1/3/2011 10:21:30 PM.312	FOCSMain	SouthWallDistUnreaso...	Avstånd till söderväggen är orimligt. Ämne, avstånd: 47, 0.1
1/3/2011 10:21:29 PM.125	1/3/2011 10:21:29 PM.125	CFOCSMain	ChargingEvent	Sätter in ämne i ugn 47 48
1/3/2011 10:19:52 PM.796	1/3/2011 10:19:52 PM.796	CFOCSMain	ChargingEvent	Sätter in ämne i ugn 46
1/3/2011 10:19:52 PM.093	1/3/2011 10:19:52 PM.093	FOCSMain	Move	Stegning ugn 301
1/3/2011 10:18:27 PM.093	1/3/2011 10:18:27 PM.093	DischargeEvent	DischargeEvent	Dragning från ugn 301 Norr
1/3/2011 10:17:47 PM.734	1/3/2011 10:18:27 PM.656	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 301
1/3/2011 10:17:46 PM.343	1/3/2011 10:17:46 PM.343	FOCSMain	DischargeInit	Dragning initierad. (ugn, balk, ämne) 0, Right, 18
1/3/2011 9:54:27 PM.312	1/3/2011 9:54:27 PM.312	FOCSMain	Move	Stegning ugn 302
1/3/2011 9:43:29 PM.093	1/3/2011 9:43:29 PM.093	FOCSMain	Move	Stegning ugn 302
1/3/2011 9:32:31 PM.156	1/3/2011 9:32:31 PM.156	FOCSMain	Move	Stegning ugn 302
1/3/2011 9:21:32 PM.328	1/3/2011 9:21:32 PM.328	FOCSMain	Move	Stegning ugn 302
1/3/2011 9:11:45 PM.890	1/3/2011 9:11:45 PM.890	FOCSMain	Move	Stegning ugn 302
1/3/2011 9:00:47 PM.312	1/3/2011 9:00:47 PM.312	FOCSMain	Move	Stegning ugn 302
1/3/2011 8:49:49 PM.406	1/3/2011 8:49:49 PM.406	FOCSMain	Move	Stegning ugn 302
1/3/2011 8:38:50 PM.625	1/3/2011 8:38:50 PM.625	FOCSMain	Move	Stegning ugn 302
1/3/2011 8:29:22 PM.125	1/3/2011 8:29:22 PM.125	FOCSMain	Move	Stegning ugn 302
1/3/2011 8:18:23 PM.765	1/3/2011 8:18:23 PM.765	FOCSMain	Move	Stegning ugn 302
1/3/2011 8:07:22 PM.656	1/3/2011 8:07:22 PM.656	FOCSMain	Move	Stegning ugn 302
1/3/2011 7:56:23 PM.718	1/3/2011 7:56:23 PM.718	FOCSMain	Move	Stegning ugn 302
1/3/2011 7:45:25 PM.531	1/3/2011 7:45:25 PM.531	FOCSMain	Move	Stegning ugn 302
1/3/2011 7:01:42 PM.640	1/3/2011 7:01:42 PM.812	FOCSMain	SouthWallDistUnreaso...	Avstånd till söderväggen är orimligt. Ämne, avstånd: 44, 0.1
1/3/2011 5:59:42 PM.015	1/3/2011 5:59:42 PM.078	FOCSMain	SouthWallDistUnreaso...	Avstånd till söderväggen är orimligt. Ämne, avstånd: 40, 0.1
1/3/2011 5:45:32 PM.015	1/3/2011 5:45:32 PM.234	FOCSMain	SouthWallDistUnreaso...	Avstånd till söderväggen är orimligt. Ämne, avstånd: 38, 0.1
1/3/2011 5:11:04 PM.828	1/3/2011 5:11:43 PM.125	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 302
1/3/2011 5:09:46 PM.281	1/3/2011 5:09:46 PM.484	FOCSMain	SouthWallDistUnreaso...	Avstånd till söderväggen är orimligt. Ämne, avstånd: 35, 0.1
1/3/2011 5:00:52 PM.062	1/3/2011 5:00:52 PM.171	FOCSMain	SouthWallDistUnreaso...	Avstånd till söderväggen är orimligt. Ämne, avstånd: 31, 0.1
1/3/2011 4:53:08 PM.171	1/3/2011 4:53:46 PM.484	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 301
1/3/2011 4:47:29 PM.562	1/3/2011 4:48:07 PM.187	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 302
1/3/2011 4:32:14 PM.968	1/3/2011 4:32:53 PM.046	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 301
1/3/2011 4:28:53 PM.593	1/3/2011 4:29:31 PM.156	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 302
1/3/2011 4:14:43 PM.796	1/3/2011 4:15:21 PM.812	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 301
1/3/2011 4:11:22 PM.843	1/3/2011 4:12:00 PM.609	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 302
1/3/2011 4:02:38 PM.562	1/3/2011 4:03:16 PM.312	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 301
1/3/2011 3:59:18 PM.015	1/3/2011 3:59:55 PM.171	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 302
1/3/2011 3:56:36 PM.625	1/3/2011 3:56:36 PM.656	FOCSMain	SouthWallDistUnreaso...	Avstånd till söderväggen är orimligt. Ämne, avstånd: 23, 0.1
1/3/2011 3:54:04 PM.578	1/3/2011 3:54:42 PM.375	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 301
1/3/2011 3:47:23 PM.218	1/3/2011 3:48:00 PM.046	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 302
1/3/2011 3:43:57 PM.609	1/3/2011 3:44:35 PM.609	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 301
1/3/2011 3:39:25 PM.750	1/3/2011 3:40:03 PM.093	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 302
1/3/2011 3:36:02 PM.171	1/3/2011 3:36:42 PM.046	FOCSMain	StocksNotInDischargePos	Dragning initierad, men ämnena har inte nått dragningsposition än i ugn 301

Figur 6: Prevas FOCS larmlista.

3 Systemintegrering

I det här kapitlet definieras ett ramverk för hur man kan integrera två system och vad man bör tänka på när man gör det. Dessutom kommer MDI-perspektiv att beskrivas som kommer att utgöra grunden för designande av bilderna i 800xA:s HMI för prototypbygget av det integrerade systemet.

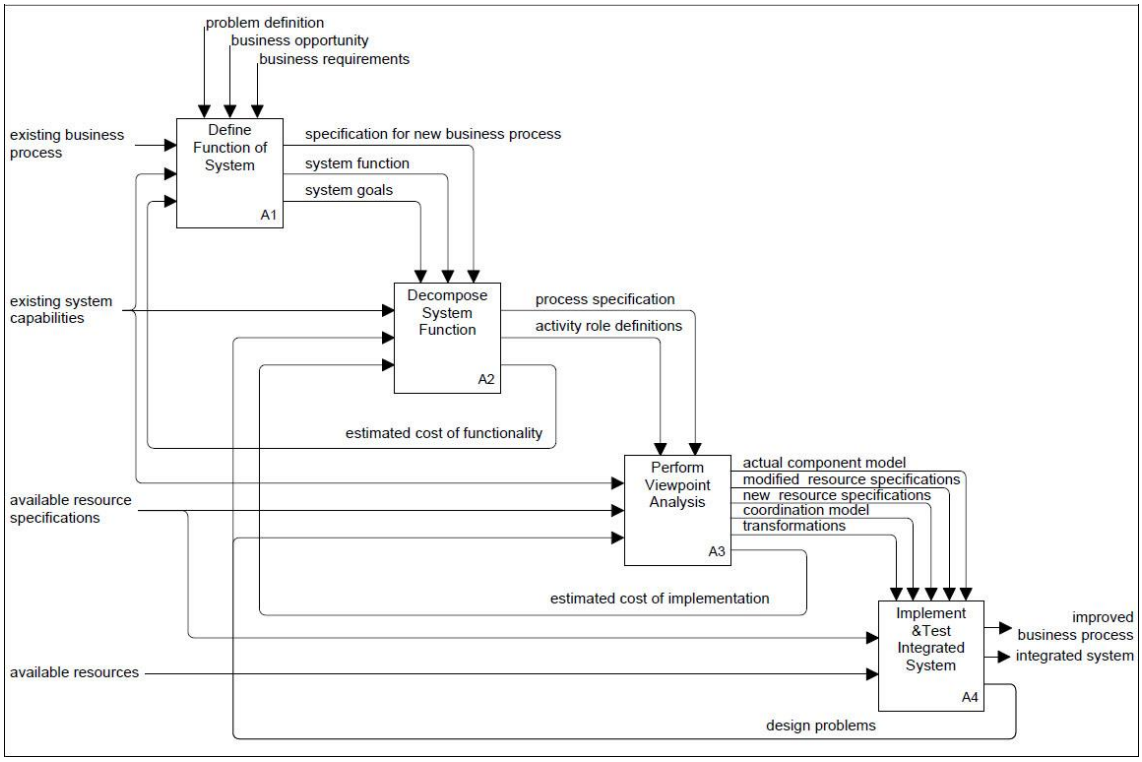
3.1 Integrationsprocessen

Syftet med systemintegrering är att få två separat designade resurser att fungera tillsammans för att uppfylla ett mål. Det finns två sätt att formulera en integrationsprocess: top-down (funktionsdriven) eller bottom-up (datadriven). Ingenjörsmässigt krävs ofta en kombination av dessa tankesätt för att lyckas. I top-down börjar man med att formulera vilken affärsprocess man har eller vill ha. Sedan utvärderar man vilka användbara resurser man har att tillgå, vad man kan använda direkt och vad man måste modifiera för att fylla luckorna som integreringen medför. I bottom-up bestämmer man att man vill använda de olika systemen till att fylla vissa funktioner i affärsprocessen. Problemet blir då istället att flytta information (data) från ett system till ett annat för att kunna förbättra processen. Dessa tillvägagångssätt är dock ofta flexibla och designen av det nya systemet sker ofta cykliskt där hänsyn måste tas till vad som är rimligt ur en kostnadssynpunkt (Barkmeyer et al. 2003:7–9).

Systemintegration i industriella system sker ofta bottom-up eftersom de resurser som är inblandade ofta redan är färdiga system med väldefinierade roller. Det huvudsakliga integrationsproblemet blir då att skapa gränssnitt och hitta sätt att medla mellan

systemen så att nödvändig data finns tillgänglig och att en specifik resurs kan utföra nödvändiga funktioner (Barkmeyer et al. 2003:16).

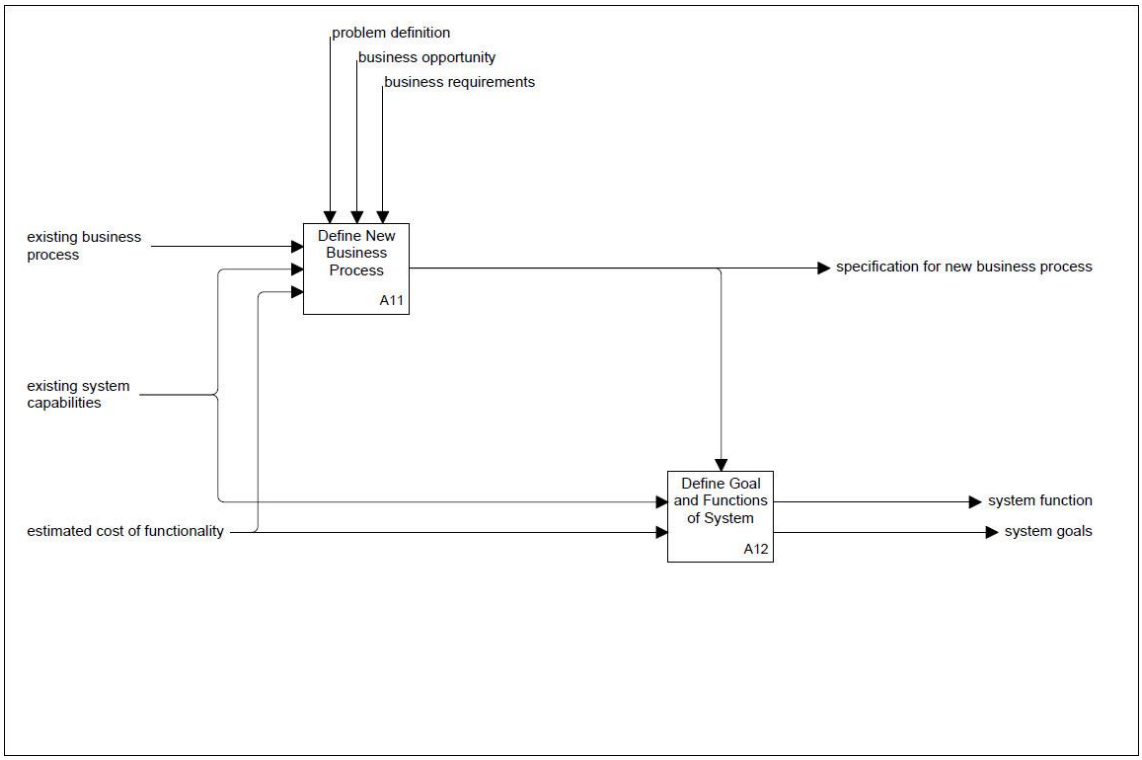
Det finns inga formella riktlinjer för hur en integreringsprocess ska se ut, däremot finns det vissa steg som är bra att följa. Barkmeyer et al. (2003:9–23) definierar ett övergripande tillvägagångssätt för systemintegrering som här kompletteras med Sommervilles (2011:83–96) beskrivning av framtagning av kravspecifikationer inom programvaruteknik. Barkmeyer et al. (2003:7) beskriver fyra övergripande steg som bör genomföras. De två första stegen i processen är mer av top-downkaraktär, medan en bottom-upintegration fokuserar mer på steg tre. De övergripande stegen för hela processen visas i Figur 7 och presenteras steg för steg härnäst.



Figur 7: De fyra övergripande stegen i systemintegrering enligt Barkmeyer et al. (2003:7).

3.1.1 Steg 1: Definiera systemets funktion och ta fram en kravspecifikation

Det första steget är att definiera problemet som ska lösas. Ibland ingår det ur en ingenjörssynpunkt att utföra en studie i huruvida den nya, integrerade, processen är kostnadseffektiv. Ofta är dock det första steget för ingenjören att definiera affärsprocessen som systemet ska stödja och definiera mål och funktioner hos systemet som stöder den processen. Efter detta steg har man en specifikation av vad systemet ska göra. Detta steg åskådliggörs i Figur 8.



Figur 8: Det första steget är enligt Barkmeyer et al. (2003:11) att ta fram en systembeskrivning.

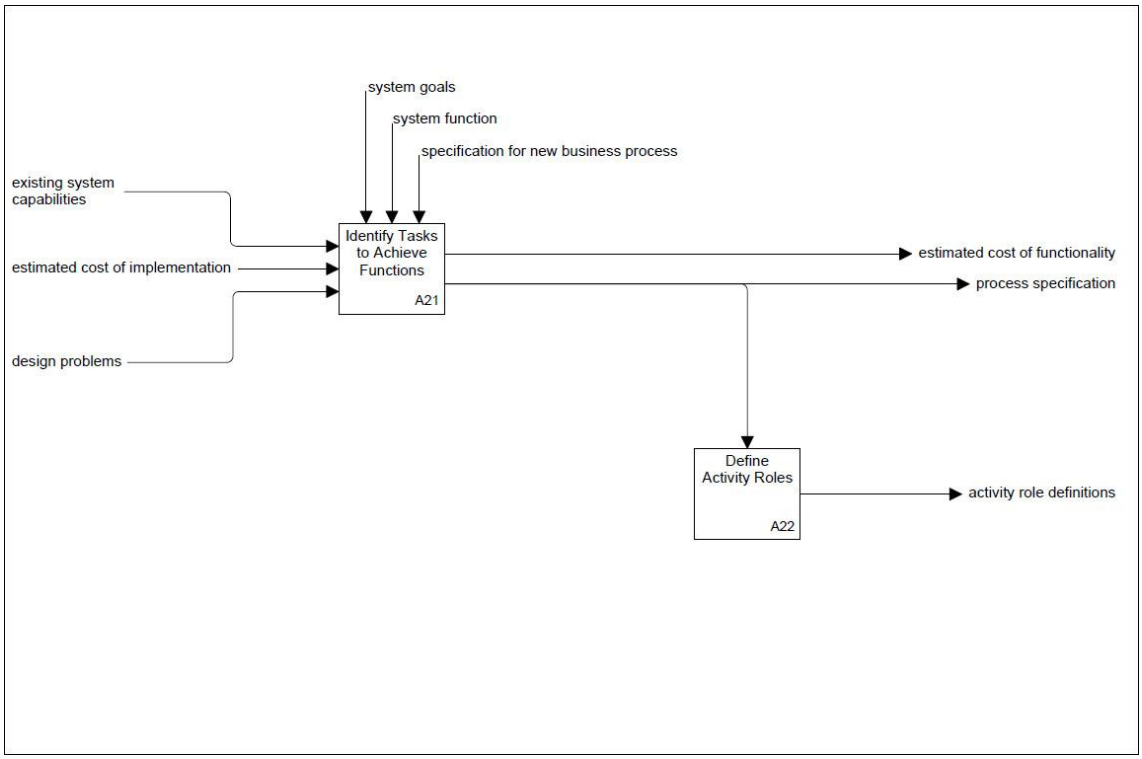
Sommerville (2011:83–96) beskriver hur en sådan kravspecifikation kan tas fram. En kravspecifikation tas fram för att beskriva vad systemet ska göra, vilka tjänster det erbjuder och vilka begränsningar som finns på det. Kraven reflekterar kundens behov av systemet. Kraven kan vara formulerade på många olika sätt och på olika detaljnivå. Ofta produceras olika kravspecifikationer för olika läsare, exempelvis behöver slutkunder inte samma information som programmerare av systemet (Sommerville 2011:83).

Krav på mjukvarusystem klassificeras ofta som funktionella eller icke-funktionella. Med funktionella krav menas beskrivningar av vad systemet ska göra. Beroende på vem de är skrivna för kan de variera i komplexitet. Användarkrav är ofta skrivna på en abstrakt nivå medan mer specifika funktionella krav kan beskriva systemets funktioner, inputs och outputs, undantag, med mera, på en mer detaljerad nivå. Det är viktigt att skriva dessa krav på en tillräckligt detaljerad nivå så att de inte kan misstolkas av systemutvecklare. Oklarheter i kravspecifikationen leder ofta till problem inom mjukvaruutveckling. Med icke-funktionella krav menas krav som inte direkt är kopplade till tjänsterna som systemet erbjuder till användarna. Det kan vara egenskaper som tillförlitlighet eller responstid men kan också vara restriktioner på systemet såsom datarepresentationer mellan gränssnitt i olika system. Dessa krav gäller generellt hela systemet och är mer kritiska än funktionella krav eftersom systemet ofta inte fungerar om kraven inte möts. Det kan vara till exempel att säkerhetskrav inte bemöts eller att systemet inte möter sina prestandakrav och därmed inte fungerar. Icke-funktionella krav uppstår på grund av användarkrav, budgetbegränsningar, organisatoriska policys, behovet av att samverka med andra system (både mjukvara och hårdvara) eller externa faktorer såsom regler och lagar. De icke-funktionella kraven bör formuleras så att de kan testas objektivt och mätas så att systemet uppfyller kraven. Ofta krockar de icke-funktionella kraven med funktionella krav eller andra icke-funktionella krav, vilket leder till att kompromisser måste göras (Sommerville 2011:84–90).

Kravspecifikationsprocessen innebär att skriva ner användar- och systemkrav i ett dokument. Det är önskvärt att kraven är otvetydiga, lätta att förstå, kompletta och konsekventa vilket i praktiken är svårt att åstadkomma eftersom de ofta går att tolka på olika sätt och kan vara beroende av varandra (Sommerville 2011:94). Det finns många olika metoder, med varierande komplexitet, för att strukturera en kravspecifikation. I många fall används naturligt språk för att skriva krav, ofta i kombination med någon grafisk modell, exempelvis UML (Sommerville 2011:95). Fördelen med att använda naturligt språk när man skriver kravspecifikationen är att det är intuitivt och lätt att förstå. Nackdelen är att det ofta är ganska vagt och betydelsen kan bero på läsarens bakgrund. Det är därför bra att på något sätt försöka strukturera upp formatet man skriver kraven på. Detta kan göras genom att exempelvis skriva varje krav som en mening och skriva källan till kravet så att man vet vem som ska informeras om något i kravet måste ändras. Man kan också skilja på vad systemet måste hantera och vad som är önskvärt att systemet ska hantera. Det kan också vara önskvärt att försöka undvika tekniska termer och förkortningar som lätt kan misstolkas (Sommerville 2011:96).

3.1.2 Steg 2: Identifiera vilka gemensamma åtgärder och resurser som behövs

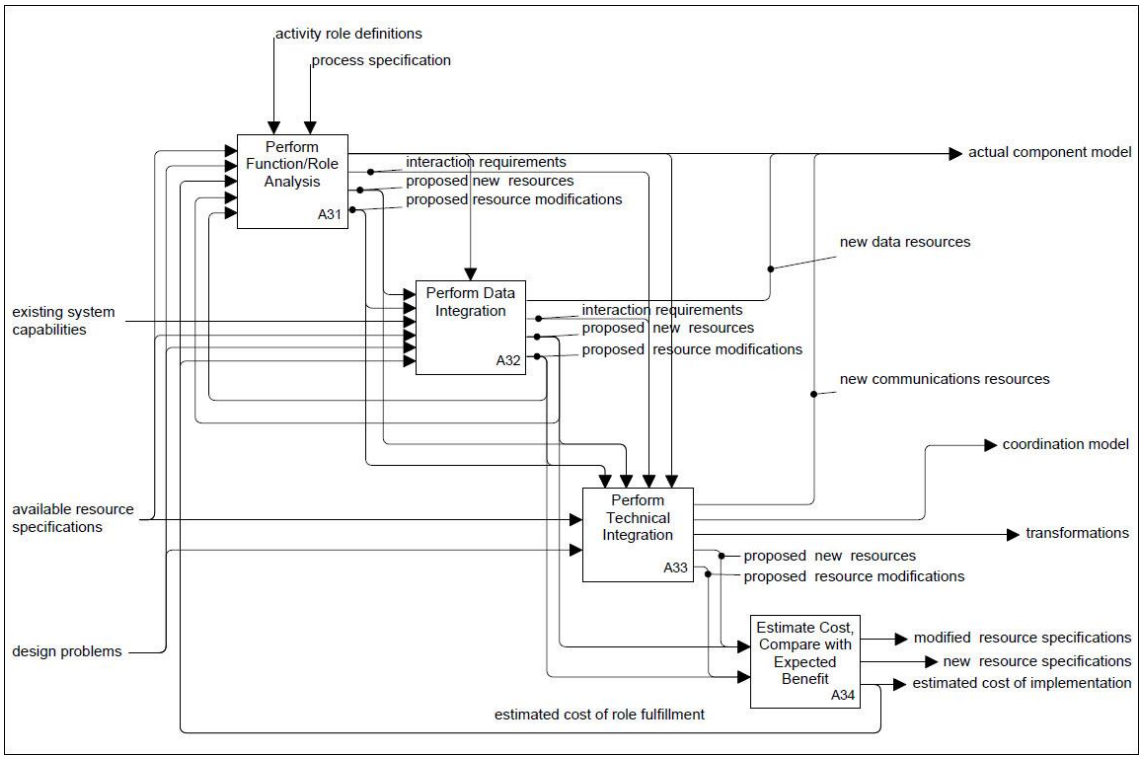
Nästa steg är att för varje funktion eller mål definiera processen för att uppnå funktionen i en graf av uppgifter som ska utföras. För varje sådan uppgift som måste utföras för att funktionen ska uppfyllas definierar man användandet av resurser i specifika roller. Roller karaktäriseras av de nödvändiga beteendena hos resursen och med egenskaper som antyder möjligheten för resursen att uppvisa dessa beteenden. När resurser som har två eller fler olika roller interagerar måste specifika behov för dess samverkan tas fram. Särskilt krav på hur de ska kommunicera är primär fokus för de tekniska aspekterna av integrationen. Generellt finns det många olika sätt att lösa dessa problem på, beroende på tillgängliga resurser. Detta steg visas i Figur 9.



Figur 9: Det andra steget går ut på att identifiera roller för resurser och definiera hur de ska samverka (Barkmeyer et al. 2003:12).

3.1.3 Steg 3: Tilldela systemroller till resurser

I detta steg identifierar man beteenden eller behov för alla roller av en resurs. Dessa bildar en övergripande roll för resursen i systemets övergripande funktion (en systemroll). Genom att undersöka karaktäristiken hos de tillgängliga resurserna tilldelar man systemroller till dem och identifierar roller som inte kan tilldelas någon resurs. För dessa roller måste någon resurs modifieras eller byggas så att dessa roller fylls. Här finns ofta också många olika tänkbara lösningar som löser problemet på ett godtagbart sätt. Det gäller att ta hänsyn till kostnader för lösningen mot förväntad nytta av lösningen. Ofta när systemen redan är färdiga och fungerar separat är resurser redan fixerade, det gäller då att hitta en lösning så att de kan kommunicera rent tekniskt. För varje rollpar som måste kommunicera med varandra kan deras resurser antingen använda samma teknologier, protokoll och datarepresentationer eller så kan de kommunicera huvudsyftet, men inte med samma teknologier, protokoll och datarepresentationer. Steg 3 visas i Figur 10.



Figur 10: Steg 3 innefattar systemrollsdefinitioner, dataintegration, teknisk integration och kostnadsanalys (Barkmeyer et al. 2003:9).

3.1.3.1 Dataintegration

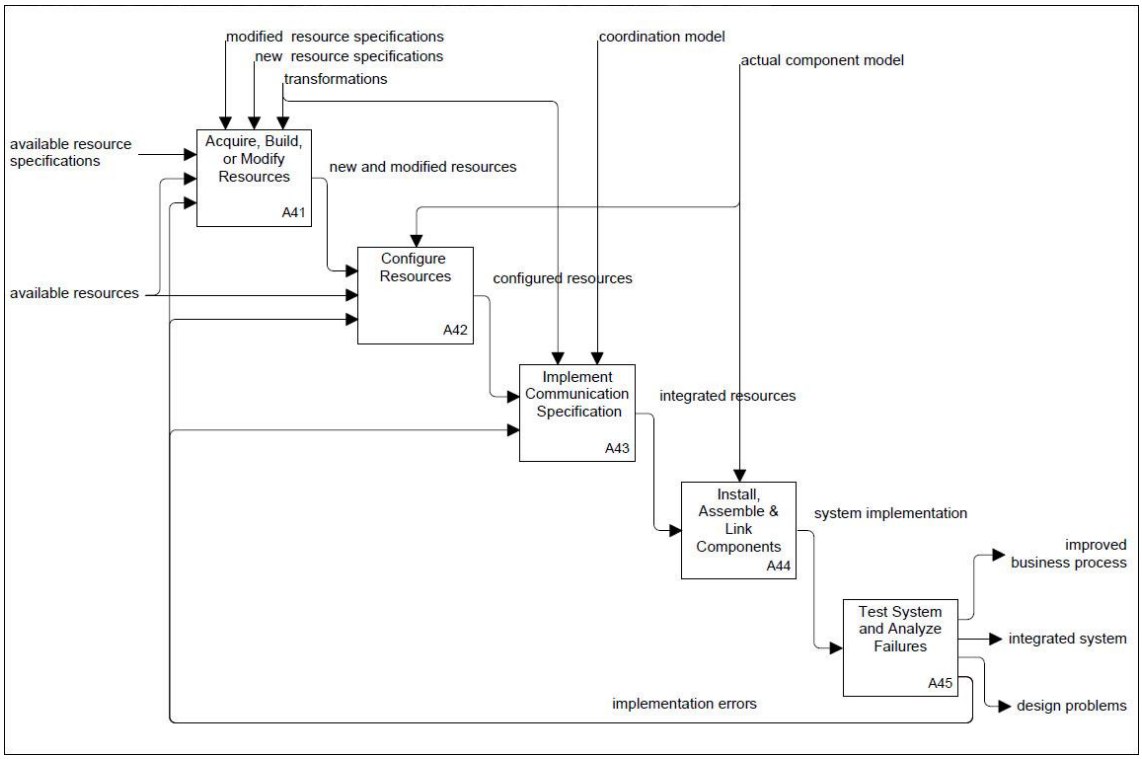
Processen för dataintegration börjar med en rolldefinition och resurstilldelning. Här försöker man ta reda på informationsbehovet hos varje uppgift ett delsystem har och hitta vägar för data som tillgodoser dessa behov i de andra resurserna som det integrerade systemet har tillgång till. Där det saknas data eller data kan tas från olika resurser specificerar man nya behov och roller för nya resurser som kan tillhandahålla den saknade informationen.

3.1.3.2 Teknisk integration

Teknisk integration innebär att se till att komponenter kan integreras genom att säkerställa att de kan kommunicera likvärdigt med andra komponenter. Transformationer som kan behövas för informationsutbyte mellan nya eller modifierade komponenter specificeras. Vilka gränssnitt och protokoll som behövs för kommunikationen definieras och de tekniska transformationer som behövs av teknologier, protokoll, datastrukturer och datarepresentationer för att komponenterna ska kommunicera bestäms. Ofta kan någon typ av broker (medlare) som omvandlar dessa behövas.

3.1.4 Steg 4: Implementera det integrerade systemet

I detta steg implementeras det integrerade systemet genom att bygga de tilläggsfunktioner eller dataresurser som behövs. Resurser modifieras eller byggs för att passa de roller som identifierats och de tekniska specifikationerna för kommunikationer implementeras. Mjukvara byggs där mer komplexa transformationer behövs och de tekniska komponenterna byggs, installeras och länkas. Detta steg visas i Figur 11.



Figur 11: Det sista steget i en integrationsprocess är enligt Barkmeyer et al. (2003:21) att implementera och testa systemet.

Den sista delen av implementationen är att testa och validera systemet för att se hur väl det tjänar den tänkta funktionen. Detta sker genom två processer: validering och verifikation. Validering innebär att bevisa att komponenterna är integrerbara i sin specifika roll och att de gemensamma åtgärder som produceras är de som ämnades. Verifikation innebär att bevisa att komponenterna kan kommunicera med varandra och att beteendet överensstämmer med specifikationerna för interoperabilitet. Hur validerings- och verifieringsprocessen gått till i detta arbete beskrivs i nästa delkapitel.

3.2 Test, validering och verifikation

Sommerville (2011:207) menar att målet med en validerings- och verifikationsprocess är att se till att systemet fungerar ändamålsenligt. Med det menas att systemet måste vara bra nog för sitt tänkta användningsområde. Ett system med hög komplexitet som styr kritiska processer måste fungera tillförlitligt och därför testas mer grundligt. En prototyp med syfte att demonstrera funktionalitet och idéer behöver dock inte testas lika noggrant. Detta beror bland annat på att användarna av systemet inte förväntar sig ett komplett, fungerande system. Det beror även på att systemet inte är tänkt att användas i produktionsmiljön utan enbart i en begränsad testmiljö. En krasch i prototypsystemet får därmed inga konsekvenser för produktionen.

Det finns många olika metoder för att testa och validera ett system. Många av dessa går ut på att ett team grundligt går igenom krav och kod enligt vissa kriterier med syftet att se till att det följer standarder, att vanliga fel undviks och att det följer specifikationerna (Adrion, Branstad och Cherniavsky 1982:166-167). Detta är dock inget vi kommer att göra, dels eftersom vi inte har resurser för det, dels för att systemet är på prototypstadiet och syftet inte är att producera ett färdigt system.

Ett annat sätt är att kontinuerligt testa systemets olika delar när ny funktionalitet läggs till. Syftet är då att ifrågasätta lösningen och utvärdera om vissa delar hade kunnat göras annorlunda (Adrion, Branstad och Cherniavsky 1982:166-167). Det är även vanligt att testa system genom simulering. Detta görs ofta i realtidssystem där kopplingen till hårdvara är central i designen. Det är då viktigt att den simulerade modellen överensstämmer med det verkliga systemet (Adrion, Branstad och Cherniavsky 1982:168–169).

3.3 Designprinciper vid skapande av ett användargränssnitt

Molich och Nielsen (1990) genomförde 1990 en fältstudie på ett användargränssnitt till ett system som används av ett telefonbolag för att få översikt på sina kunder. Systemet var ett gammalt kommandopromptsystem och använde sig därför bara av två färger och ett tangentbord, varken mus eller grafik fanns att tillgå. Fokus på studien var: hur borde gränssnittet designas för att hjälpa användarna att arbeta på ett mer effektivt sätt. Detta gjordes genom att använda sig av dels sin egen expertis i området och dels en enkätundersökning med 154 medverkande respondenter med olika akademiska bakgrunder. Respondenterna fick självständigt utvärdera systemet utifrån sin unika erfarenhet för att sedan lämna in resultatet till författarna. Molich och Nielsen (1990) sammanställde sedan de olika problem, brister och lösningar som respondenterna hade identifierat till kategorier. Sammanlagt gav fältstudien upphov till nio olika kategorier som sedan skrevs om till principer. Tanken var att om man följer dessa principer vid designande av ett system ökar man chansen att få ett användarvänligt system.

Simple and Natural Dialogue
Alla informationsrutor ska innehålla relevant information, oberoende om det är text eller bilder som informationen utgörs av. All onödig eller irrelevant information konkurrerar med den viktiga informationen, vilket betyder att översiktligheten minskar. Den relevanta informationen ska presenteras på ett naturligt och logiskt sätt för att öka överskådligheten för användarna.

Speak the User's Language
Informationen som finns tillgänglig för användarna ska vara skriven på användarens språk istället för med tekniska och systemorienterande termer. Om det är möjligt att skriva informationen på användarens modersmål ska även detta göras. Rör det sig om avancerade användare som använder sig av ett eget fackspråk (till exempel läkare och tandläkare) ska även informationen i systemet vara skrivet på detta fackspråk.

Minimize the User's Memory Load
Människor har ett begränsat närminne som är beroende av olika faktorer, till exempel stress och trötthet. Därför är det viktigt att förse användarna med hjälpmedel så att de ska slippa att memorera i största möjliga mån. Tydliga och lätta instruktioner ska finnas så att användaren snabbt kan kontrollera dessa vid tveksamheter. Komplicerade instruktioner ska brytas ner och skrivas om simplare.

Be Consistent
En användare ska aldrig uppleva oklarheter kring definitioner i systemet. Om möjligt ska användarna alltid behöva göra samma steg för en funktion oberoende var i systemet det är. Ett exempel på det är hur användarna gör för att acceptera en förändring i systemet. Detta skulle kunna ske på följande sätt: steg1: tryck ändra, steg2: välj nytt

värde och slutligen steg3: acceptera. Samma gäller med till exempel avbrytknappar där det måste vara konsistent om det ska betyda att backa ett steg eller avbryta till början.

Provide Feedback
Systemet ska alltid förse användarna med feedback på vad som pågår i systemet. Speciellt viktigt är detta när en handling tar tid att utföra. Vid stressiga situationer kan annars användaren göra samma handlingar flera gånger i tron att systemet inte uppmärksammade de första knapptryckningarna.

Provide Clearly Marked Exits
Ett system ska aldrig låsa en användare till sina val. Eftersom användare ofta råkar klicka på fel knapp eller ångrar sig måste det finnas alternativ för att ta sig tillbaka utan att utföra några ändringar.

Provide Shortcuts
En egenskap som gör ett system lätt att lära sig är oftast dåligt och icke önskvärt för avancerade användare. Smarta genvägar kan då skapa ett system som är önskvärt för både nya och avancerade användare.

Provide Good Error Messages
Ett bra felmeddelande ska vara definitivt, noggrant och konstruktivt. Defensiva felmeddelande skyller det inträffade felet på brister i systemet istället för att kritisera användaren för att ha använt systemet på fel sätt. För att klassificera ett felmeddelande som noggrant ska meddelandet ge användaren tillräckligt med information berörande vad som har gått fel och varför. Konstruktiva felmeddelanden ger användaren meningsfulla förslag på hur problemet kan lösas.

Error Prevention
Vad som är ännu bättre än att ha bra felmeddelanden är att designa systemet på ett sådant sätt att felen inte kan inträffa från första början.

Sedan den första artikeln kom ut har de ursprungliga nio principerna skrivits om mer generellt för att kunna passa in på flera system och även utökats med ytterligare en princip: ”help and documentation”:

Help and Documentation
Även om man strävar efter att designa system så naturligt som möjligt kommer det alltid finnas fall där användningsområdet är tillräckligt avancerat att hjälpfiler är ett måste. Då måste det finnas bra skrivna hjälpfiler som användarna kan enkelt söka igenom för att hitta den information som de behöver (Nielsen Norman Group 1995 [www]).

De nya principerna bygger som sagt på de ursprungliga nio principerna och lyder i sin helhet (ibid):

- Visibility of system status
- Match between system and the real world
- User control and freedom
- Consistency and standards
- Error prevention
- Recognition rather than recall
- Flexibility and efficiency of use

- Aesthetic and minimalist design
- Help users recognize, diagnose, and recover from errors
- Help and documentation

4 Framtagning av integrationsalternativ

I detta kapitel utreds integrationsmöjligheter för att kunna besvara studiens första mål. Kapitlet inleds med en definition av systemets funktion och framtagning av en kravspecifikation för integreringen av de två styrsystemen. Kravspecifikationen och de olika systemens egenskaper ligger här till grund för de olika möjliga integrationsmetoderna som presenteras. Detta kapitel behandlar steg 1 till och med steg 3 i integrationsprocessen som beskrevs i föregående kapitel. Den valda metoden har använts vid skapandet av prototypen som behandlas i nästkommande kapitel.

4.1 Systemdefinition och framtagning av kravspecifikation

ABB och Prevas har ingått ett samarbete för att marknadsföra och sälja optimeringssystem för styrning av ugnar, FOCS (Prevas 2010 [www]). Samarbetet har inletts eftersom en integrerad lösning av Prevas styrsystem för ugnar FOCS och ABB:s styrsystem för industrier 800xA skulle innebära möjligheter att hitta nya marknader och stärka positionerna på befintliga marknader. Ytterligare en möjlighet med samarbetet är att kunder som använder det integrerade systemet kommer att spara energi, vilket även är ett krav för industrier som enligt miljöbalken måste använda teknik som hushåller med energi (Naturvårdsverket 2014 [www]). Om en industri vill använda sig av de båda systemen idag så går det, men systemen har olika gränssnitt och de olika funktionerna som styr liknande processer, som exempelvis larm, hanteras på olika sätt i de båda systemen. Det är därför önskvärt att kunna integrera dessa funktioner. Ur en operatörssynpunkt är det dessutom fördelaktigt att systemen fungerar och ser ut som ett enhetligt system.

För att kunna definiera det nya systemet beskrivs först de existerande systemens affärsprocesser samt vad systemen idag klarar av enligt modellen för systemintegrering.

ABB:s styrsystem 800xA är ett level 1-system för att integrera flera olika funktioner inom industrier till ett enhetligt system. I 800xA kan bilder skapas genom aspektservern som via en eller flera OPC-servrar (i ABB:s fall konnektivitetsserver) prenumererar på värden som gör information tillgänglig till operatörerna. Operatörerna kan i sin tur styra industriprocesser genom HMI:et som skickar signaler till en kontroller (i ABB:s fall AC 800M) att ändra något i styrningen av processen.

Prevas styrsystem för ugnar Prevas FOCS är ett level 2-system som är designat för att optimera energiförbrukningen för processer som använder sig av omvärmningsugnar. Systemet består av en server och en klient. Servern har en OPC-klient som kan kopplas upp mot en tredjeparts OPC-server. Servern kan på det viset prenumerera på OPC-signaler och styra ugnar. Serven får också tillgång till ett annat level 2-system som har hand om ämneslistor. Informationen ur dessa två system visas i Prevas FOCS klient där operatören kan övervaka och styra produktionen.

Syftet med projektet är att dessa två styrsystem ska integreras ur ett operatörsperspektiv. Med detta menas att operatörer ska kunna styra en ugnsprocess och få tillgång till relevant information för att göra detta. Informationen ska göras tillgänglig via 800xA,

eftersom att det är detta system som kommer användas för styrningsprocessen. För att kunna ta fram mål och funktioner som systemet måste hantera har en kravspecifikation tagits fram. Eftersom att båda systemen redan fungerar separat uppfyller de redan många av de krav som ofta tas fram i en kravspecifikation. Därför kommer vi endast att ta hänsyn till de krav som är relevanta för att integrera de båda systemen. Det gäller dock att se till så att systemet, efter integrationen, fortfarande uppfyller viktiga krav som ställs på systemen separat i form av bland annat tillförlitlighet och säkerhet.

Utförandet av kravspecifikationen har följt processen beskriven i systemintegreringskapitlet, dock utan UML. För att få bättre överblick av kraven har de istället strukturerats i olika kategorier utifrån vad som påverkas av kravet. De olika kategorierna som valdes är: systemkrav, designkrav, kommunikationskrav och dokumentationskrav.

Kraven har skrivits med naturligt språk och eftersom de båda systemen redan är fullt fungerande är de flesta kraven av icke-funktionell karaktär. Dessa krav är svåra att testa och tenderar att vara ganska övergripande, men kommer att finnas med i val av systemdesign och vid val av systemarkitektur. Detta innebär att för dessa krav blir det författarnas tolkning av desamma, med motivering, som används vid validering av systemet. Vissa krav, som rör funktionalitet av det integrerade systemet, kommer att tas upp i samband med byggandet av prototypen i kapitel 5. Vidare har kraven specificerats i vad systemet ska hantera och vad systemet bör hantera. De har även strukturerats utifrån om de är funktionella eller icke-funktionella och om det är ABB eller Prevas som har ställt kravet. ABB och Prevas kommer således att ha kundrollen, då inte slutkunder av systemet står för kraven. På grund av detta har vi valt att använda vissa tekniska termer i kraven, eftersom att läsarna av kraven redan är insatta i tekniken och missförstånd därför är mindre troliga. Kraven har tagits fram vid intervjuer med Uhlander (2014) och Söderlund (2014) och presenteras nedan i Tabell 1.

Tabell 1: Kravlista på integreringen av Prevas FOCS och 800xA.

Systemkrav	**Typ**	**Ska/Bör**	**Kravägare**
Systemen ska vara så åtskilda som möjligt	Icke-funktionell	Ska	ABB
Minimalt ingrepp i 800xA	Icke-funktionell	Ska	ABB
All funktionalitet som förs över från Prevas FOCS ska fungera likvärdigt i 800xA	Icke-funktionell	Ska	Prevas
Integrationen ska vara billig i underhåll och resultatet ska kunna återanvändas	Icke-funktionell	Ska	Prevas
Integrationen ska bara gälla funktionalitet relevant för operatörer	Icke-funktionell	Ska	ABB
Designkrav	**Typ**	**Ska/Bör**	**Kravägare**
Bilder som skapas ska följa stilguiden för metals design	Icke-funktionell	Ska	ABB
Bilderna ska vara generella för alla industrier	Icke-funktionell	Ska	ABB
Ugnskarta och taktstyrning (pacing) ska finnas tillgängligt för operatörer	Funktionell	Ska	Prevas

Kommunikationskrav	Typ	Ska/Bör	Kravägare
Kritisk data får inte passera genom level 2	Icke-funktionell	Ska	ABB
Bilderna ska få information via OPC	Funktionell	Ska	ABB
Prevas FOCS bör kommunicera med endast en OPC-server	Funktionell	Bör	Prevas
Dokumentationskrav	**Typ**	**Ska/Bör**	**Kravägare**
All kod som behöver skrivas ska vara väldokumenterad	Icke-funktionell	Ska	ABB
All kod ska vara på engelska	Icke-funktionell	Ska	ABB
Det ska finnas gemensam dokumentation i form av signallistor mellan ABB och Prevas	Icke-funktionell	Ska	ABB

De framtagna kraven har givetvis en stor påverkan på hur lösningen kan komma att se ut. Därför kommer kraven att presenteras mer specifikt utifrån de fyra funna kategorierna.

Systemkrav
Systemen ska vara så åtskilda som möjligt, vilket har en del implikationer på lösningen. Det är önskvärt att systemen är åtskilda av olika orsaker. Både Prevas och ABB vill kunna sälja sina system separat, vilket underlättas av att de hålls så åtskilda som möjligt. Det blir även enklare att undvika potentiella fel och brister som en integrering kan medföra på systemen om lösningen sker på en åtskild nivå. Att hålla systemen åtskilda ökar således chansen att de fungerar på samma sätt som innan även efter integreringen. För annan funktionalitet än den som operatörerna behöver ska en annan dator som kör FOCS kunna användas.

ABB har även som krav att minimalt ingrepp ska ske i 800xA. Med detta menas att den större engineeringen bör ske i FOCS, det vill säga att större delen av kodning och så vidare ska ske i Prevas system. Detta är även rimligt eftersom att kod enklare kan modifieras i FOCS, då den i 800xA ligger dold längre ner i strukturen. 800xA är designat för att vara enkelt att använda, men det innebär även att det är svårare att modifiera förprogrammerade strukturer och designer.

Vidare finns krav på att all funktionalitet som förs över från FOCS ska fungera likvärdigt i 800xA. Systemet ska kunna styras i 800xA på samma sätt som i FOCS klient (undantaget viss funktionalitet som beskrivs senare) utan buggar och med bibehållen precision. En mer detaljerad kravspecifikation för vilken funktionalitet som ska integreras presenteras i kapitel 5 och finns även utförligare beskriven i Bilaga 1: Prevas FOCS funktioner.

Integrationen ska även vara så billig som möjligt och resultatet ska kunna återanvändas. Med att lösningen ska vara billig menas att underhållskostnaden ska vara så låg som möjligt. Initialt kan projektet kosta en del, men att uppdatera och hålla systemet gående ska vara enkelt och därmed billigt. Att resultatet ska kunna återanvändas betyder i detta fall att det ska vara enkelt att flytta lösningen mellan olika företag med liknande behov av systemen. Det ska alltså inte behöva göras stora ändringar mellan liknande industrier.

Dessutom ska lösningen helst kunna appliceras för integreringar mellan andra system än just de här två om sådana behov skulle uppstå i andra projekt. För det här projektets omfång ska integrationen enbart gälla funktionalitet som är relevant för operatörer.

Designkrav
Det finns även krav på hur lösningen ska vara designad. Alla bilder som skapas i 800xA ska följa ABB:s stilguide för metals design. Denna stilguide består av interna dokument som beskriver vilka färger och fonter som ska användas i olika situationer. Den beskriver även generell design så som hur knappar ska se ut. Den är framtagen för att underlätta för operatören att se relevant information och innehåller mycket dämpade färger eftersom att rummen operatörerna befinner sig i ofta är ganska mörka. Att då behöva se på skrikiga färger och samtidigt vara koncentrerad under en lång tid blir påfrestande för operatörerna.

Bilderna ska även vara generella för alla industrier, vilket betyder att bilderna ska vara skapade så att de fungerar dynamiskt och går att använda för exempelvis olika storlekar på ugnar eller ugnar som har olika antal ugnszoner. Detta är viktigt för att minimera behovet av, potentiellt kostsamma, specialanpassade lösningar så långt som möjligt. Att bilderna är dynamiska är viktigt för att de ska gå att placera på godtycklig plats i HMI:et. Det ska även finnas möjlighet att ändra bildernas storlek beroende på exempelvis antal skärmar som finns tillgängliga för operatören, vilket kan variera mellan industrier. Det var även önskvärt att all information ska gå att göra tillgänglig på endast en skärm, i alla fall för standardutförandet med bara en ugn och begränsat antal zoner.

För den här integrationen var kravet att funktionalitet som rör operatörer ska integreras. En mer detaljerad beskrivning av dessa funktioner fås i Bilaga 1: Prevas FOCS funktioner. De funktioner som operatörer främst behöver är ugnskartan, med tillhörande funktionalitet så som att byta värmningsmod i ugnens olika zoner och taktstyrningsinformation med möjlighet att byta taktstyrningsmod. Även funktionen att stega och dra ämnen, samt funktionen att ändra mintider (minimal tid som måste passera mellan dragna ämnen för långt och kort ämne) ska vara med vid integreringen. Övrig funktionalitet som idag finns tillgänglig för operatörer i Prevas FOCS rör planerade stopp och luckor i produktionen. Denna funktionalitet kommer troligtvis att försvinna i framtida versioner av Prevas FOCS och kommer därför inte att integreras. Dessa funktioner beskrivs även utförligare i kapitel 5.1.3 där designen av användargränssnittet för det integrerade systemet presenteras.

Kommunikationskrav
Det uppkom även krav som specifikt rör kommunikationen mellan systemen. Information som bedöms kritisk för att systemet ska fungera korrekt får inte passera genom level 2. PLC:er är designade för att alltid vara operativa och förväntas att klara av att köras i ungefär 10 år utan avbrott. Level 2-system å andra sidan är baserade på vanliga datorer som kör Windows. Om information som är kritisk för systemet, exempelvis sådan information som hanteras av PLC:er, skulle behöva passera en vanlig dator blir systemet känsligt för datorkraschar. Sådana processer ska fortsättningsvis hanteras på samma sätt som innan integreringen.

Bilderna som skapas i 800xA får sin information via OPC och ett krav är att de bilder vi skapar också får sin information via OPC. Det finns ingen möjlighet att exempelvis skicka information i datastrukturer som Prevas FOCS använder sig av med hjälp av WCF eftersom att 800xA inte kan hantera den informationen. Detta krav begränsar valet

av integrationsmetod, eftersom möjligheten att använda andra alternativ än just OPC blir svårt (mer om detta senare).

Prevas har även ett bör-krav att FOCS helst bara ska kommunicera med en OPC-server, då detta är standard i dagsläget.

Dokumentationskrav
Det framkom även en del krav på dokumentation. Dessa krav är inte direkt relevanta för integreringsprocessen, utan mer viktiga för Prevas och ABB för att kunna hantera informationen som utbyts mellan systemen. Signallistor har tagits fram som beskriver informationen som förs över från Prevas FOCS till 800xA. Dessa har varit till hjälp för att ta reda på vilken information som behövs och för att ta reda på hur den kan överföras. Det är även bra att veta vilken enhet som används för information som flödar mellan systemen och i de flesta fall har SI-enheter använts vid prototypbygget. Det uppkom även krav på själva kodandet, nämligen att det ska vara väldokumenterat och på engelska.

4.2 Resurser och roller

Utifrån kravspecifikationen och de befintliga resurserna som finns att tillgå i de båda systemen kan roller för det nya systemet tilldelas resurserna. I vissa fall finns olika sätt att lösa kommunikationen mellan systemen. Syftet med detta steg i integrationsprocessen är att identifiera vilka resurser som kan ges vilka roller. Därför presenteras inledningsvis vilka resurser som finns att tillgå i de båda systemen för att komma fram till vilka roller som behövs efter integrationen.

4.2.1 800xA:s resurser

De, för integreringen, mest intressanta resurser som 800xA har är aspektservern som har hand om bilderna och konnektivitetsservern som sköter kommunikationen, framförallt i form av OPC.

4.2.2 Prevas FOCS resurser

Prevas FOCS har sin server med möjlighet att kommunicera både via WCF och OPC. Dock finns idag endast stöd för OPC DA eftersom att larmhanteringen sker internt. Informationen fås från kontrollers via OPC och uträkningar och optimering görs för att sedan skickas till klienten med WCF för visning.

4.2.3 Roller för det integrerade systemet

För att överföra den funktionalitet som är relevant för operatörer, det vill säga ugnskartan och taktstyrningen, sätter främst kommunikationskraven begränsningar på vilka roller som kan tilldelas de olika resurserna.

Bilder ska skapas i 800xA som tillgodoser operatörer med information. ABB:s aspektserver och bildbyggarprogram har som roll att tillgodose denna funktionalitet. För att bilderna ska få information måste informationen kommuniceras till dem via OPC enligt kravspecifikationen. Idag har ABB:s konnektivitetsserver som uppgift att tillgodose detta behov, men nu måste även hänsyn tas till information som finns i Prevas FOCS. Rollen att förse bilderna med information kan därmed tillgodoses från flera håll, men kommer att ske via OPC. Rollen att förse bilderna med information via OPC måste tillgodoses av någon resurs. Här kan konnektivitetsservern användas då den har en

OPC-server och Prevas FOCS server har en OPC-klient. Dessa resurser har potential att samarbeta. Här måste dock hänsyn också tas till kravet från ABB att så lite som möjligt ska modifieras i 800xA och bör-kravet från Prevas att FOCS helst bara ska kommunicera med en OPC-server. Om konnektivitetsservern ska ha som roll att föra information mellan systemen krävs att en större mängd OPC-taggar definieras i den, vilket får anses innebära ett ganska stort ingrepp i systemet (mer om detta senare).

Prevas FOCS har som uppgift att optimera ugnsstyrningen och tar information från en ämnesserver och från kontrollers. Denna information bearbetas i Prevas FOCS server för att optimera processen och kommuniceras till Prevas FOCS klient via WCF och nu måste även delar av denna information skickas till 800xA.

Prevas FOCS server har även möjlighet att direkt kommunicera via WCF genom att skicka samlade datastrukturer med information. Men eftersom konnektivitesservern inte själv har en resurs för att tillgodogöra denna information kommer det behövas ett hjälpmedel för detta om denna kommunikationsväg ska användas.

Prevas FOCS klient kommer efter integrationen inte behövas ur operatörssynpunkt. Däremot kommer den ha som roll att täcka de behov som finns för anda användare som behöver använda systemet och funktionerna är såklart viktiga att ha kvar så att systemet kan säljas separat.

4.3 Systemroller och saknad funktionalitet

ABB:s aspektserver kan användas för att skapa alla för operatören relevanta bilder och får därför systemrollen att skapa bilderna. Här finns all funktionalitet redan tillgänglig och allt som behöver göras är att skapa bilder som överensstämmer med bilder som just nu skapas i Prevas FOCS. Ett krav är att bilderna ska följa ABB:s designmall för metals design och är generella för alla industrier för att kunna återanvändas, exempelvis att ugnskartan är stor nog att tillgodose alla ugnstyper.

Prevas FOCS server kommer i det nya systemet att vara ansvarig för ugnsstyrning och all funktionalitet som har med styrningen att göra, exempelvis beräkningar. Den kommer därför att ha detta som sin systemroll. Ugnsstyrningen kommer att kunna skötas från två olika håll, dels från Prevas FOCS klient som administrativa användare och processingenjörer kommer att använda, dels från 800xA där operatörer kommer interagera med produktionen. Prevas FOCS server kommer också ha hand om rollen att förse 800xA med den information som behövs för att inte operatörerna ska tappa någon funktionalitet vid borttagandet av Prevas FOCS klient.

För att få kommunikationen mellan systemen att fungera behövs det introduceras en ny roll då detta inte existerar i något av systemen idag. Den information som operatörerna använder sig av för att sköta den dagliga driften finns bara att tillgå som WCF vilket 800xA inte stöder.

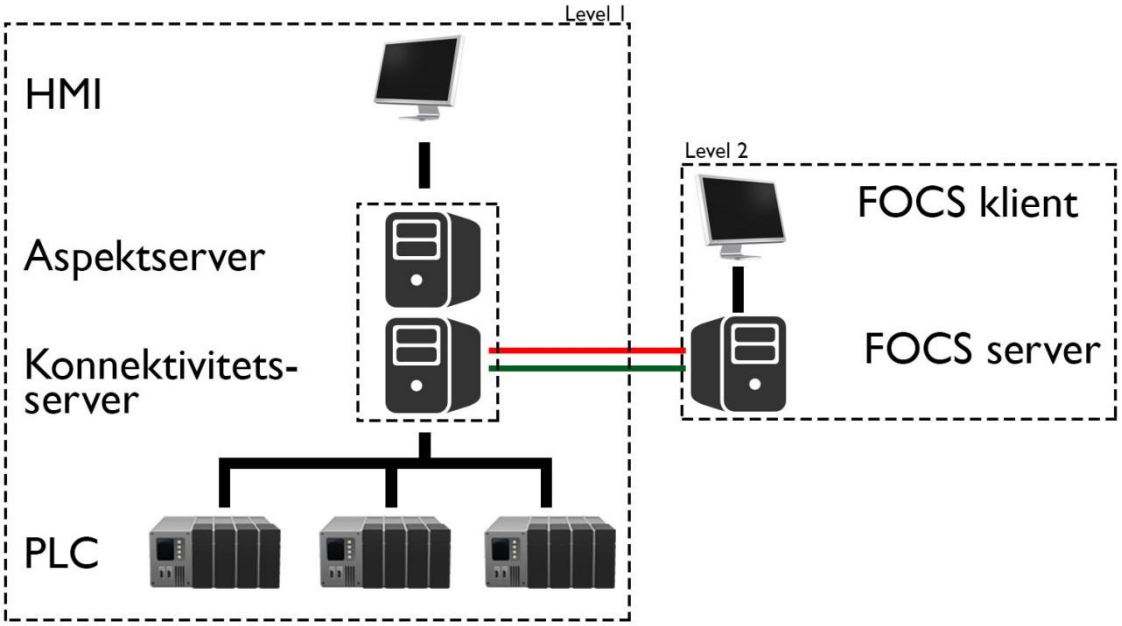
Det saknas även en roll för att integrera larmen eftersom att dessa i Prevas FOCS hanteras med hjälp av WCF och i 800xA med hjälp av OPC A&E. På grund av detta måste larmhanteringen ses över i Prevas FOCS eftersom att rollen att förse 800xA med larm som rör Prevas FOCS funktioner kommer att förses av Prevas FOCS server.

4.3.1 Dataintegration och teknisk integration av 800xA och Prevas FOCS

Antalet sätt att integrera 800xA och Prevas FOCS med varandra är redan reducerade genom kravspecifikationen tidigare i kapitel 4. Till exempel så omöjliggör ”Systemen ska vara så åtskilda som möjligt” och ”Minimalt ingrepp i 800xA” att Prevas FOCS styrfunktioner integreras in som funktioner i 800xA. Däremot finns det en rad olika möjligheter som fortfarande är aktuella.

Integrationslösning 1

Genom att använda den redan existerande konnektivitsservern som finns i 800xA och OPC-klienten som finns inbyggd i Prevas FOCS kan ett informationsutbyte ske relativt enkelt. Om också denna OPC-server tillhandahåller den information som Prevas FOCS behöver för sina modeller kommer även kravet ”Prevas FOCS bör kommunicera med endast en OPC-server” uppfyllas. Utmaningen med att använda sig av denna integrationsmöjlighet är att Prevas FOCS server kommer behöva programmeras om så att den inte enbart använder sig av WCF (för HMI-data). Detta innebär att flertalet OPC-taggar måste definieras i konnektivitetsservern. Det beror på att WCF är dynamiskt, vilket gör att det är enkelt att tillföra nya variabler, medan OPC är statiskt. Till exempel kan ett nytt ämne enkelt läggas till i den färdiga datastrukturen som är gjord i WCF, däremot krävs att lika många OPC-taggar definieras som antalet variabler som finns för ett ämne (runt 20 stycken). På grund av antalet ämnen blir därför antalet OPC-taggar snabbt stort. Lösningen visas i Figur 12, där gröna streck visar att Prevas FOCS får information medan röda streck visar att Prevas FOCS skickar information.



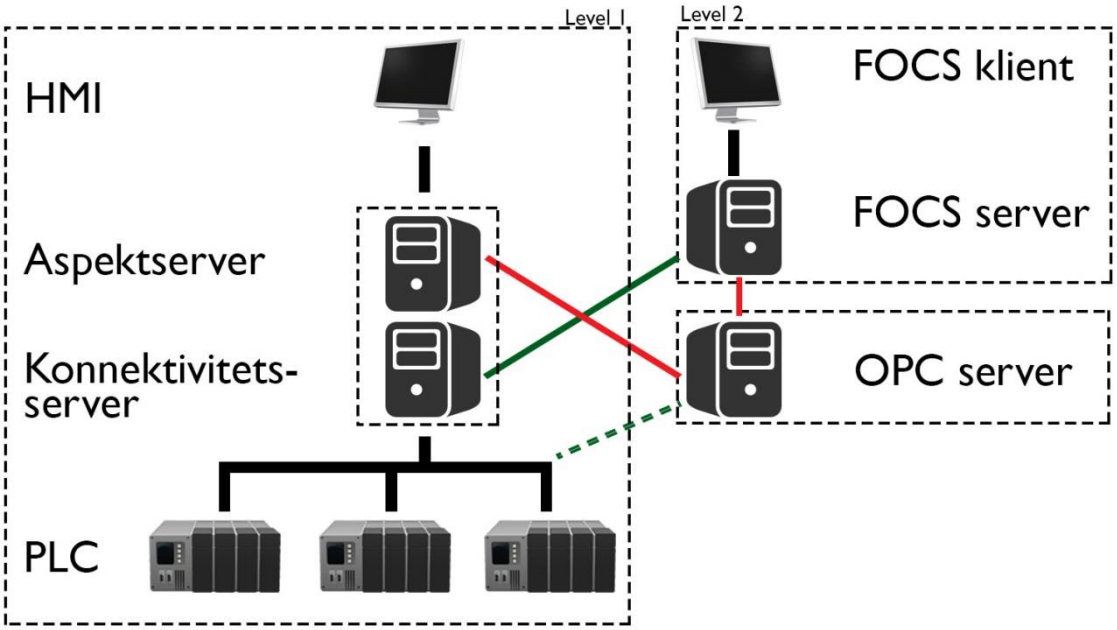
Figur 12: Integrationslösning med Prevas FOCS server kopplad till 800xA:s konnektivitetsserver.

Positivt:
Kommunikationskanalen finns redan.
Generell integration som går att återanvända.

Negativt:
Ett stort antal nya OPC-taggar behöver definieras i konnektivitetsservern.
Signaler som 800xA behöver måste konverteras till OPC i Prevas FOCS. Om Prevas

FOCS klient även behövs efter integrationen måste samma information skickas både genom WCF och OPC.

Integrationslösning 2
Istället för att använda den redan existerande konnektivitetsservern kan man installera en ny tredjeparts OPC-server. Det skulle innebära att de nya utökade taggarna inte kommer att beröra det redan existerande systemet. Däremot kommer antingen Prevas FOCS behöva kontakt med två olika OPC-servrar för att få tillgång till all nödvändig data eller så kommer informationen att finnas på två ställen. Dels i den nya tredjeparts OPC-server och dels i den redan existerande konnektivitetsservern. Detta kan leda till viss problematik för larmlistorna eftersom att tidstämplarna kan variera från den ena servern till den andra. Integrationslösningen kan ses nedan i Figur 13. Grön streckad linje betyder att vägen är ett alternativ eller en komplettering till den heldragna gröna linjen.



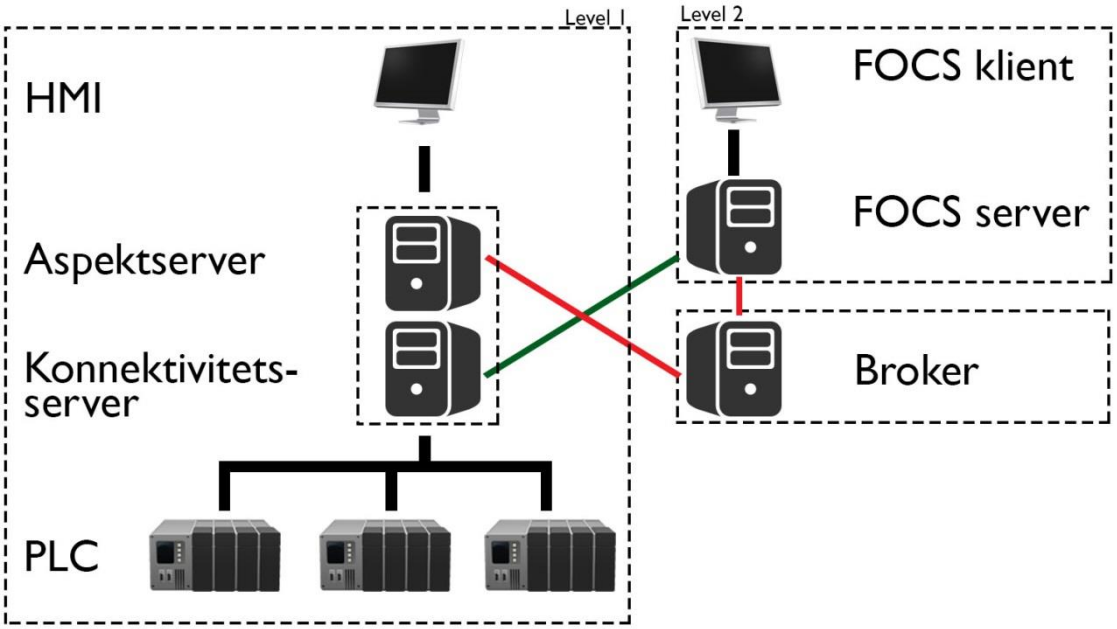
Figur 13: Extern OPC-server kopplad till systemen.

Positivt:
Kommunikationskanalen finns redan.
Generell integration som går att återanvända.

Negativt:
Prevas FOCS kommer att behöva använda sig av dubbla kommunikationer.
Larmlistan kan komma att vara osynkroniserad.
Prevas FOCS kan behöva hämta information från två olika OPC-servrar.

Integrationslösning 3
En broker som är inkopplad direkt till Prevas FOCS server skulle möjliggöra användningen av WCF. Fördelen med detta är att informationen som ska överföras från Prevas FOCS inte behöver översättas till OPC direkt i Prevas FOCS. Detta innebär att data utan stor modifikation kan skickas på samma sätt som den gör till Prevas FOCS klient idag. Ett problem med denna lösning är dock att konnektivitetsservern inte stöder WCF. Därför kommer brokern behöva översätta WCF till OPC i alla fall, och samma

problem med OPC-taggarna uppkommer. Dessutom försvinner fördelen med att Prevas FOCS redan har inbyggda funktioner för att lätt konvertera till OPC. Lösningen gör dock informationsutbytet mer åtskilt eftersom att varken Prevas FOCS eller ABB 800xA behöver modifieras nämnvärt. Lösningen visas i Figur 14 nedan.

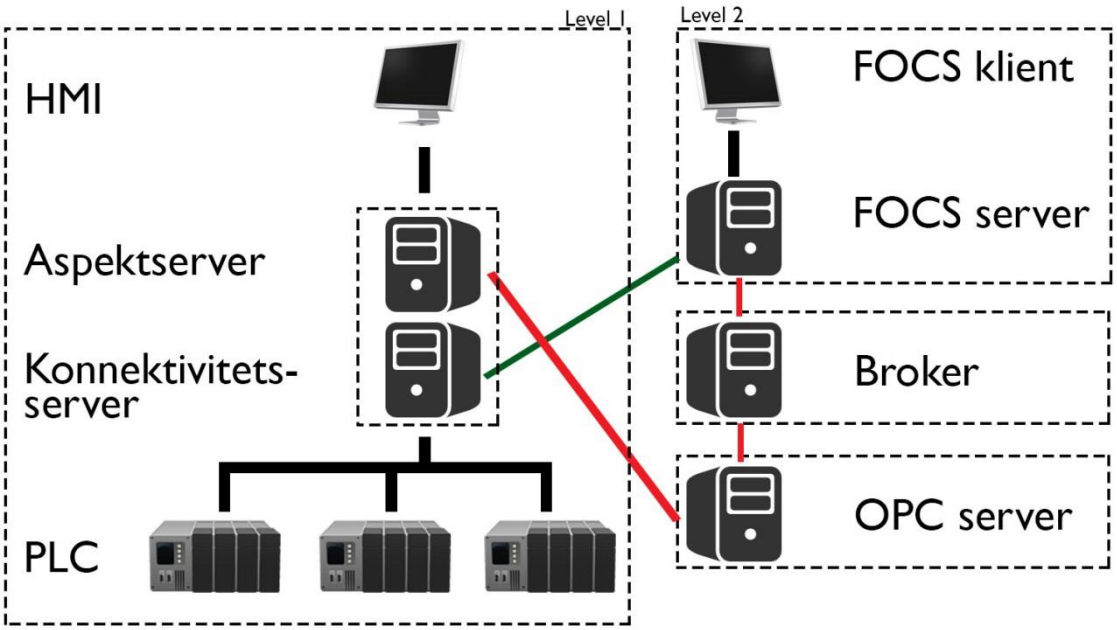


Figur 14: Lösning med broker som översätter WCF till OPC.

Positivt:
Prevas FOCS behöver inte använda dubbla kommunikationer.
Minimal påverkan på de båda systemen eftersom det mesta sker i brokern.
Generell integration som går att återanvända.

Negativt:
Använder inte Prevas FOCS färdiga system för att konvertera till OPC.
En ny mjukvara behöver skapas och underhållas.
Ett stort antal nya OPC-taggar behöver definieras.

Integrationslösning 4
Integrationslösning 4 är en kombination av integrationslösning 2 och integrationslösning 3, det vill säga att koppla brokern till en tredjeparts OPC-server. Även här blir för och nackdelarna liknande. Prevas FOCS kan komma att behöva ha kontakt med två OPC-servrar som enligt kravspecifikationen inte var önskvärt och det kommer behöva skapas en ny mjukvara i form av en databroker. Lösningen åskådliggörs i Figur 15.

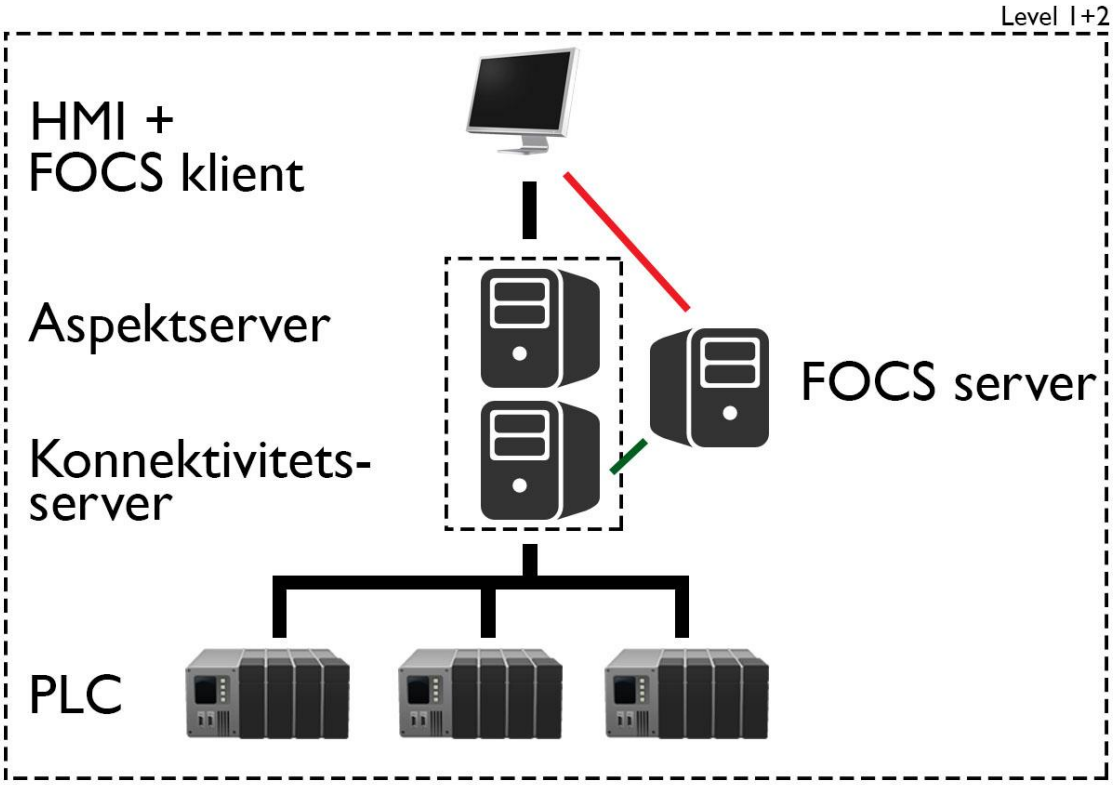


Figur 15: Lösning med broker och OPC-server.

Positivt:
Prevas FOCS behöver inte använda dubbla kommunikationer.
Generell integration som går att återanvända.

Negativt:
Använder inte Prevas FOCS färdiga system för att konvertera till OPC.
En ny mjukvara behöver skapas och underhållas.
Larmlistan kan komma att vara osynkroniserad.
Prevas FOCS kan behöva hämta information från två olika OPC-servrar.

Integrationslösning 5
Den enklaste lösningen rent tekniskt är att installera Prevas FOCS som en applikation inuti 800xA:s HMI. Det innebär att operatörerna får full funktionalitet genom att de använder det redan beprövade gränssnittet som kommer med Prevas FOCS. Nackdelarna med detta blir däremot stora eftersom det kommer att fungera som ett gränssnitt inuti ett annat gränssnitt med helt olika design. Det kommer också finnas två helt olika larmlistor, en från 800xA och en från Prevas FOCS. Detta resulterar i att operatörerna själva måste ha uppsikt över vilket system som larmet inträffat i, vilket innebär att under stressfulla situationer ökar risken för att operatörerna missar viktig information. Lösningen visas i Figur 16.



Figur 16: FOCS klient körs som externt program från 800xA:s HMI.

Positivt:
Enkelt, billigt, beprövat.

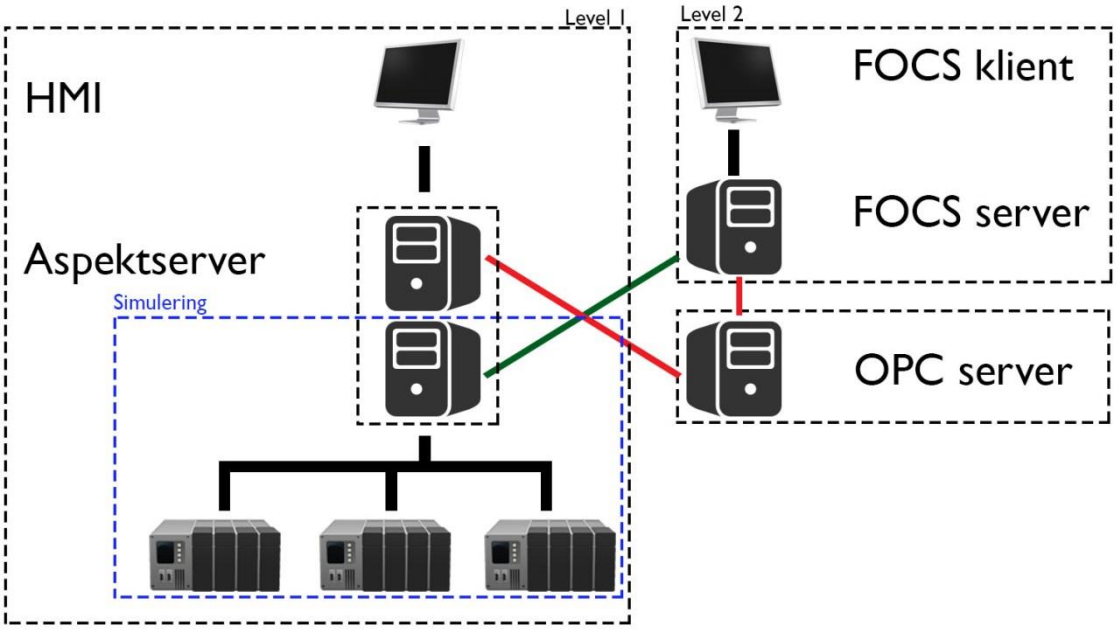
Negativt:
Bilderna kommer inte att följa stilguiden.
Ökade krav för operatörerna.

4.3.2 Val av integrationsmetod

De olika alternativen som har presenterats innebär olika för- och nackdelar. Vid val av lösning har hänsyn tagits till kraven som ställdes av både ABB och Prevas. Vi har också fått göra en avvägning mellan ekonomiska faktorer och användbarhetsfaktorer. Till exempel är alternativ 5, att helt enkelt bara köra Prevas FOCS klient som externt program i 800xA, klart både det enklaste och billigaste alternativet. Däremot är det det klart sämsta ur operatörernas synvinkel, eftersom de kommer att behöva använda olika program med olika gränssnitt för att utföra sina arbetsuppgifter. Som tidigare nämnts i inledningen av denna uppsats kan detta leda till allvarliga följder i form av hälsorisker såsom stress och utbrändhet. Av den anledningen har detta alternativ inte bedömts som tillräckligt bra. Både alternativ 3 och 4 kräver byggandet av en broker som översätter WCF till OPC för att systemen ska kunna kommunicera. Fördelen med dessa lösningar är att systemen blir mer åtskilda, eftersom att den mesta kommunikationsmodifikationen kommer att ske i ett separat program på en extern server. Lösningarna blir dock troligtvis dyrare, eftersom att ett helt nytt system måste byggas och underhållas. Det blir även ett mer komplext system eftersom att flera extra servrar måste läggas till (kanske behövs mer än en broker för att skapa redundans så att systemet inte ska sluta fungera om servern av någon anledning kraschar).

Ur en ekonomisk synpunkt är det helt klart bäst att så långt som möjligt använda sig av befintliga resurser. Integrationslösning 1 använder sig av ABB:s konnektivitetsserver som OPC-server och i Prevas FOCS server byggs tilläggsfunktioner för att föra över informationen som behövs. Denna lösning innebär att så många funktioner som möjligt återanvänds samtidigt som HMI:et integreras för att öka användbarheten för operatörerna. Av de funna lösningarna är det denna som vi anser har bäst förutsättningar med avseende på krav, ekonomi och användbarhet. Det är därför denna integrationslösning vi rekommenderar ABB och Prevas att använda i detta fall.

För byggandet av en prototyp har vi däremot valt lösning 2 eftersom prototypbygget har utgått från simulerad data och för teständamål har det varit praktiskt att istället för att använda konnektivitetsservern sätta upp en egen OPC-server. Data simuleras i Prevas FOCS server och ersätter den information som systemet vanligtvis får från PLC:er i industrimiljön. Att denna lösning använts för prototypbygget påverkar inte slutresultatet, utan ger Prevas och ABB ytterligare en möjlighet till integration om exempelvis en extern OPC-server skulle vara önskvärd. Lösningen som använts utgår som sagt från integrationsalternativ 2 och presenteras nedan i Figur 17, där den simulerade delen av systemet är markerad.



Figur 17: Integrationslösning 2 som används för prototypbygget där den simulerade delen av systemet är markerad.

5 Resultat och prototypbygge

Efter val av integreringsmetod påbörjades bygget av en prototyplösning på integrationen av systemen. Detta kapitel beskriver processen med att ta fram denna prototyp och uppfyller därmed även det andra och tredje målet med denna uppsats. Utgångspunkten för detta kapitel är steg 4 i integrationsprocessen som presenterades i kapitel 3 som just behandlar implementering, test och validering.

5.1 Implementering av det integrerade systemet

För att implementera det integrerade systemet måste de funktioner som saknas byggas och de resurser som finns att tillgå modifieras för att kommunikationen mellan systemen ska fungera som tänkt. Det gäller att se till att skapa och länka ihop dessa tilläggsfunktioner för att det integrerade systemet ska fungera som en helhet.

Utifrån den integrationslösningen som valdes för prototypbygget behövdes vissa resurser modifieras och vissa läggas till. Dessa var:

- En OPC-server måste implementeras och konfigureras.
- Prevas FOCS måste modifieras så att relevant information även skickas med OPC till den nya OPC-servern och så att Prevas FOCS kan ta emot de värden som skickas via OPC från 800xA.
- Bilder måste designas i 800xA så att funktionaliteten i Prevas FOCS klient kan styras helt via 800xA:s HMI.

För att kunna utföra dessa modifikationer på systemen fick vi tillgång till både Prevas FOCS och 800xA via virtuella maskiner (VMware) med programmen installerade. På så sätt kunde vi koda, bygga bilder och sätta upp servrar som vi ville.

5.1.1 Implementering av OPC-server

För att sätta upp en OPC-server och få kommunikationen att fungera mellan systemen krävdes initialt mycket arbete med brandväggs- och DCOM-inställningar. Det blev snabbt uppenbart att nackdelen med DCOM, att det är svårkonfigurerat, stämde bra in på verkligheten. För att kunna koppla upp mot en OPC-server som inte finns på samma maskin krävs att DCOM-rättigheter ställs in så att användare får koppla upp via ”remote access”. Detta måste dock göras på en hel del olika ställen och är svårt att felsöka, vilket gjorde att det tog en hel del tid att få att fungera. Till sist fick vi dock kommunikationen att fungera en väg; Prevas FOCS kunde nås från 800xA, men inte tvärtom. Därför valde vi att sätta upp OPC-servern på Prevas FOCS virtuella maskin.

Det finns många tillverkare av OPC-serverlösningar och de flesta erbjuder testversioner som får köras kostnadsfritt, men har en begränsning på körtid. Detta för att kunna konfigurera och testa systemet. Vill man sedan använda lösningen kommersiellt, med obegränsad körtid, får man köpa en licens. Valet av OPC-server blev i vårt fall Kepwares KEPServerEX 5. Den valdes för att Prevas FOCS tidigare använt sig av denna server, men då OPC är en standard spelar valet av tillverkare ingen större roll. Utan licens har KEPServerEX 5 en begränsad körtid på två timmar åt gången, sedan måste den startas om för att fungera igen.

Konfigurationen av OPC-servern gick till så att taggar skapades för varje signal som kommer att behövas. Signaler som behövs fick vi från att i Prevas FOCS klient studera vilken information som kommer att behöva visas i 800xA. Varje siffra som behöver visas eller ändras behöver en egen OPC-signal och måste därmed finnas som OPC-tagg. För varje tagg ställer man in ett unikt namn, exempelvis FOCS.Stock.CustomerId1 (visar kundnumret för ämne 1). Dessutom definierar man vilken datatyp som taggen har, exempelvis float för ett flyttal. Det finns möjlighet att i KEPServerEX 5 generera flera taggar på en gång, vilket gjorde processen ganska smidig då lösningen skulle vara generell och då exempelvis innehålla möjlighet för användande av åtminstone 80 olika ämnen.

Larmhantering i KEPServerEX 5 sker genom ett plug in för alarm & events. Genom detta finns möjlighet att ställa in så att ett larm avges när en viss händelse inträffar. Detta ställs in genom att lyssna på en OPC-tagg och välja vid vilket värde man vill larma, vilket kan vara om värdet är under eller över en viss nivå, eller om värdet byter status (till exempel från 0 till 1). Dessutom ställer man in allvarlighetsgrad mellan 1 och 1000, för att indikera hur allvarlig händelsen är.

5.1.2 Modifikation av Prevas FOCS

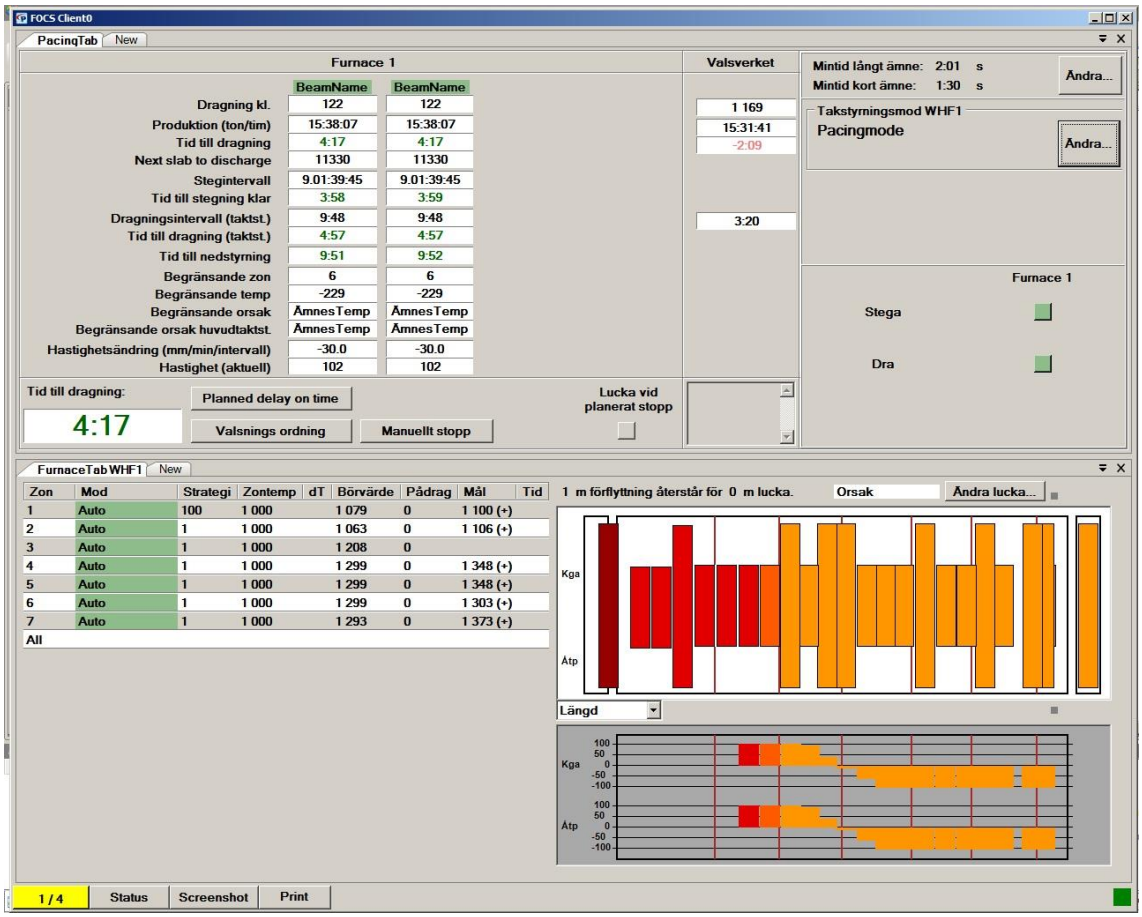
Som tidigare nämnts skickas idag information från Prevas FOCS server till Prevas FOCS klient via WCF. Prevas FOCS har dock programmerats så att det finns möjlighet att ta emot och skicka information via OPC, vilket är ett måste för att kunna kommunicera med PLC:er. Däremot finns inga färdiga funktioner för att skicka den information som operatörerna behöver via OPC. Dessa funktioner har vi fått lägga till genom C#-programmering. Prevas FOCS använder sig även av Microsoft SQL Server 2005 för databashanteringen och för OPC-signaler ska varje signal (OPC-tagg) även finnas i databasen. Ett problem var att lösa hur dessa rader skulle läggas till i databasen på ett effektivt sätt (bara ämnesinformationen som krävs för 80 ämnen resulterar i 1440 rader i databasen). För att underlätta denna uppdatering av databasen skrevs ett script som automatiskt genererar alla databasrader.

Eftersom att Prevas FOCS inte stöder OPC A&E finns ingen given integrationsmöjlighet för detta. För att larmen ska gå att integrera i vår prototyp måste de antingen konverteras till A&E-standarden helt och hållet eller konverteras till OPC-taggar där larm kan ställas in att avges när taggarna byter status. Båda alternativen kräver mycket jobb i och med att larmen idag finns på olika platser i Prevas FOCS.

5.1.3 Design av bilder i 800xA

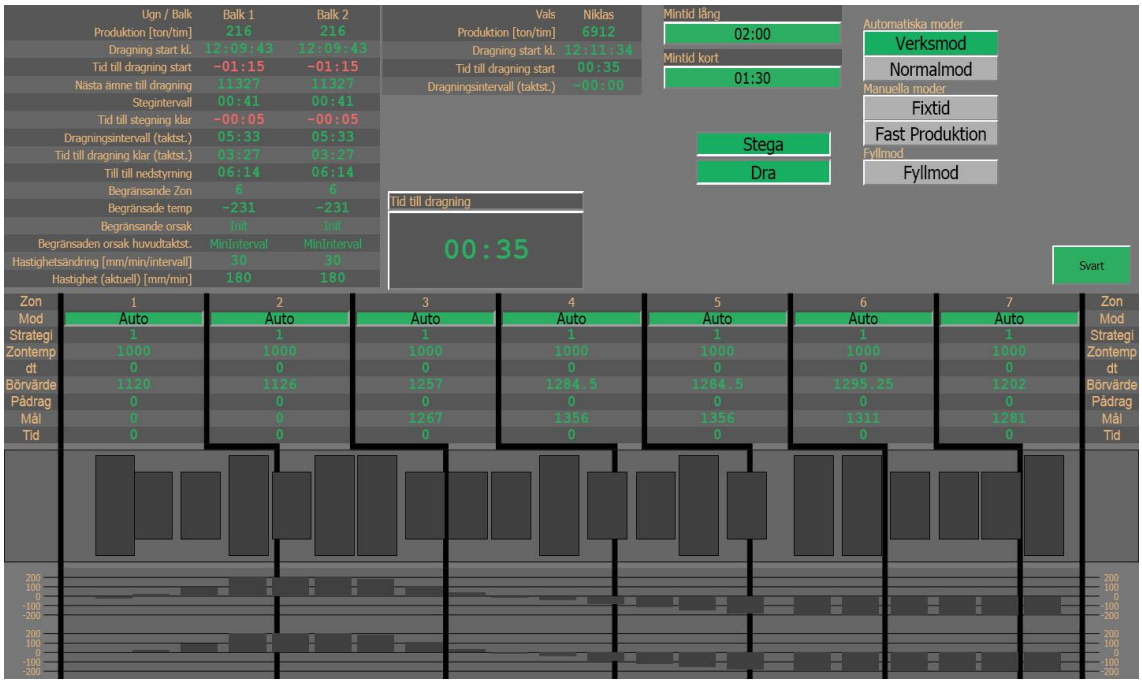
Vid design av bilder i 800xA har hänsyn tagits till både designkraven som ställdes från Prevas och ABB och Molich och Nielsens (1990) designprinciper vid framtagning av ett användargränssnitt. En mer detaljerad kravspecifikation av vilka funktioner som måste tas med vid integreringen finns i Bilaga 1: Prevas FOCS funktioner.

Den första bilden för operatörerna är ugnsöversikten, här kan operatören ha en helhetsbild över vad som pågår i ugnen och utföra justeringar som att ställa in temperaturen eller att ställa in så att Prevas FOCS själv automatiskt ställer in temperaturen. Översiktsbilden kan ses i Figur 18 som är ett exempel på hur det kan se ut för en operatör för en omvärmningsugn som värmer stål innan det ska valsas.



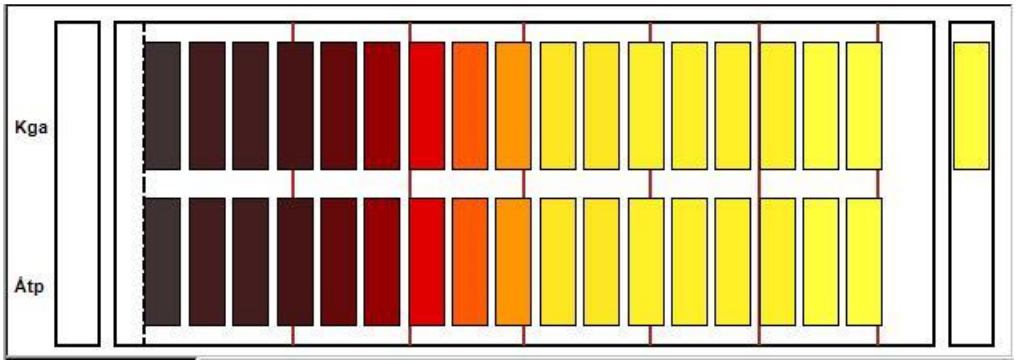
Figur 18: Översiktsbild för operatörer i Prevas FOCS med taktstyrning överst och ugnsstyrningsbilden underst.

Designen för prototypbygget bygger ursprungligen på en uppdaterad version av Prevas FOCS design. Den nya designen är utvecklad utifrån både ABB:s metals design och Molich och Nielsens (1990) principer för användargränssnitt. ABB:s metals design är en designmall som tar upp vilka färger som är optimalt att använda för operatörer till omvärmningsugnar. Deras miljö skiljer sig mot de flesta operatörers eftersom de sitter i ett nedsläckt rum med en glasruta till vägg så de kan se in i lokalen bredvid där ugnen är placerad. Rummet är nedsläckt för att det inte ska finnas några reflektioner på glasrutan. Genom att dova ner alla färger i användargränssnittet blir inte påfrestningen på operatörernas ögon lika högt, vilket leder till att de kan koncentrera sig längre. Den nya designen i sin helhet syns i Figur 19.



Figur 19: Nya designen på FOCS HMI i 800xA.

Längst ner till höger i Figur 18 finns en ugnskarta som visar hur många ämnen som finns i ugnen och vilken temperatur de har. Figur 20 är en närbild på just en sådan karta. Det finns sex stycken lodräta linjer som går från toppen till botten på bilden, de visar de olika zonerna i ugnen som operatörerna kan justera temperaturen i. Det finns två kvadrater längst upp och längst ner till höger i ugnsbilden som indikerar om det finns något ämne i ugnen som ligger i dragläge (detta syns bara i Figur 18). Att ämnet ligger i dragläge innebär att det ligger på en position så att uttagsmaskinen kan ta ut det. Går man tillbaka till Figur 18 ser man att dessa kvadrater är gråa, vilket indikerar att inget ämne är i dragläge. Är dessa kvadrater istället gröna innebär det att ett ämne ligger i dragläge. Beroende på hur ugnen fysiskt är placerad på de olika industrierna, eller preferenser från operatörerna kan man välja om ämnena ska gå från höger till vänster eller från vänster till höger.



Figur 20: Ugnskartan i Prevas FOCS.

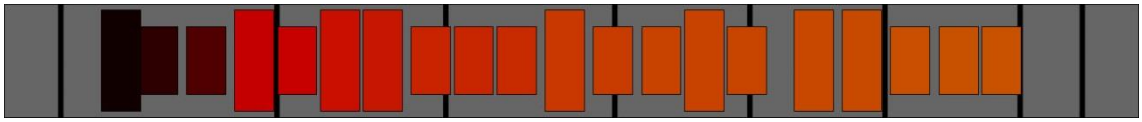
Om man för muspekaren över ett av ämnena på ugnskartan kommer det upp en informationsruta som innehåller all data som finns om ämnet. Denna information finns förklarad i Bilaga 1: Prevas FOCS funktioner. Designmässigt ser informationen likadan ut på både det gamla och det nya systemet. Informationsrutan ses i Figur 21.

Figur 21: Ämnesinformation i Prevas FOCS. Informationen ser i princip likadan ut i den nya designen för 800xA.

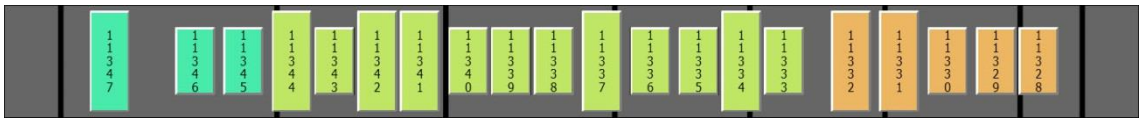
Som kan ses i Figur 19 ser ugnskartan, förutom det nya färgschemat, likadant ut som i Prevas FOCS version av systemet. Linjerna som visar var de olika ugnszonerna finns kvar och ämnena rör sig i realtid från vänster till höger. En ny funktion är däremot att operatörerna enkelt kan byta färg på ämnena beroende på egen preferens. De har valen att välja följande:

- Svarta
- Efter temperatur
- Efter produktorder

Svart betyder helt enkelt att alla ämnena är så mörka som ABB:s metals design tillåter, detta för att minimera påfrestningen på operatörernas ögon. Om operatören vill ha bättre koll på hur varmt varje ämne är, precis som i Prevas FOCS, används färgvalet ”efter temperatur”. ”Efter produktorder” betyder att varje ämne som tillhör samma produktklass får en egen färg. Detta är en vanlig form av visning som operatörer idag använder sig av eftersom det är i produktionsskiftena som operatörerna vanligtvis behöver bäst översikt. För att öka friheten hos operatörerna kan även både ugnsbildens storlek och visningsplats förändras beroende på operatörens preferenser. Dessa olika färgalternativ syns i Figur 22 och Figur 23.



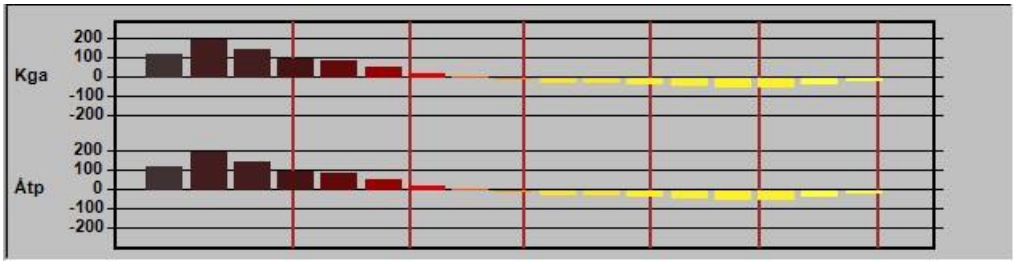
Figur 22: Nya ugnsbilden inställd på temperatur.



Figur 23: Nya ugnsbilden inställd på order.

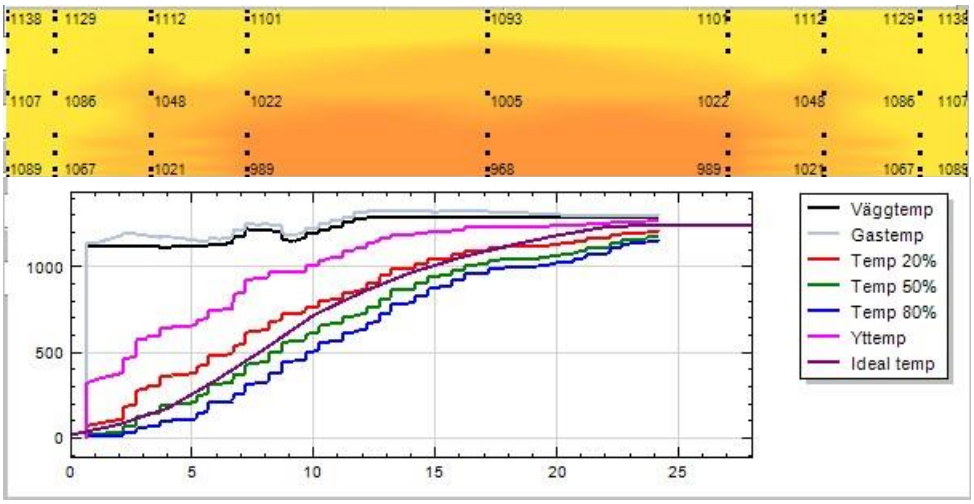
Under ugnskartan i översiktsbilden finns visningskurvor för ämnestemperatur. Det är en graf över hur mycket varians det är från optimal temperatur till faktisk temperatur i varje ämne. Figur 24 visar hur de första ämnena ligger 100-200 grader över sin

optimala temperatur och ämnena på slutet ligger lite under sin optimala temperatur fram till dragningspositionen då det bara skiljer ett tiotal grader.



Figur 24: Ämnens temperaturavvikelse i Prevas FOCS.

Om operatören håller muspekaren över något av ämnena på visningskurvorna visas ytterligare två bilder med information. Den översta bilden i Figur 25 visar temperaturen på 81 olika punkter inuti ämnena. Denna figur är också färgkodad beroende på temperatur. Den andra bilden i Figur 25 är en graf över olika temperaturer olika långt in i ugnen. Det finns sju olika temperaturberäkningar, nämligen; väggtemperaturen inuti ugnen, gastemperaturen som värmer upp ugnen, fyra olika temperaturer på fyra olika djup i ämnet och ämnets optimala temperatur.



Figur 25: Ämnestemperatur i Prevas FOCS.

Temperaturavvikelsegrafen i prototypen som syns längst ner i Figur 19 är också designad som Figur 24, det vill säga mycket har återanvänts från Prevas FOCS design. Samma nya funktioner som i ugnsbilden finns även här. Nu är också bilderna uppdelade så att operatören själv kan bestämma om han vill se båda bilderna eller bara en. Det finns också en ny bild som operatörerna kan välja att lägga till mitt i ugnsbilden och temperaturavvikelsebilden som länkar samman linjerna för zongränserna för en bättre översikt. Denna bild återfinns också mellan temperaturstyrningen och ugnsbilden.

Eftersom 800xA inte stödjer att ha grafer i popup-fönster som i Prevas FOCS har författarna valt att inte flytta över denna funktionalitet till prototypen. Den funktionaliteten bedömdes även mindre viktig för operatörerna, i alla fall för prototypen av systemet.

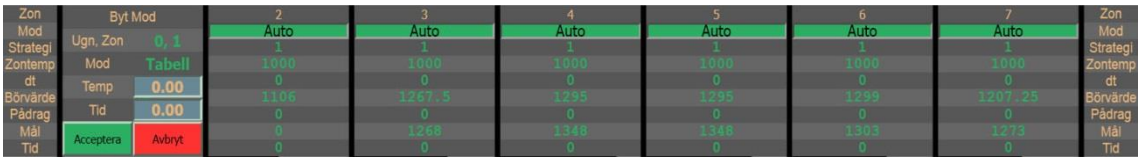
Den sista bilden i översiktsfönstret är temperaturregleringen. Där övervakas och styrs temperaturen i de olika områdena i ugnen. Temperaturregleringen har åtta olika

kolumner med information som syns i Figur 26. Antal rader i temperaturregleringen varierar beroende på vilken typ av ugn det är, i detta fall finns det 16 olika zoner inuti ugnen som operatören individuellt kan byta värmemod i. En mer detaljerad beskrivning av temperaturstyrningens funktioner finns i Bilaga 1: Prevas FOCS funktioner.

Zon	Mod	Strategi	Zontemp	dT	Börvärde	Pådrag	Mål	Tid
1	Auto	1	1 125	0	1 125	85		
2	Auto	1	1 125	0	1 125	85		
3	Auto	1	1 176	0	1 176	85		
4	Auto	1	1 126	0	1 126	85		
5	Auto	1	1 126	0	1 126	85		
6	Auto	1	1 201	0	1 201	85		
7	Auto	1	1 300	-148	1 300	85	1 347 (+)	
8	Auto	1	1 152	*	1 152	85		
9	Auto	1	1 300	101	1 300	85	1 352 (+)	
10	Auto	1	1 199	*	1 198	85		
11	Auto	1	1 300	0	1 300	85	1 360 (+)	
12	Auto	1	1 300	*	1 300	85	1 347 (+)	
13	Auto	1	1 300	0	1 300	85	1 372 (+)	
14	Auto	1	1 300	*	1 300	85	1 372 (+)	
15	Auto	1	1 300	0	1 300	85	1 324 (+)	
16	Auto	1	1 300	*	1 300	85	1 372 (+)	
All								

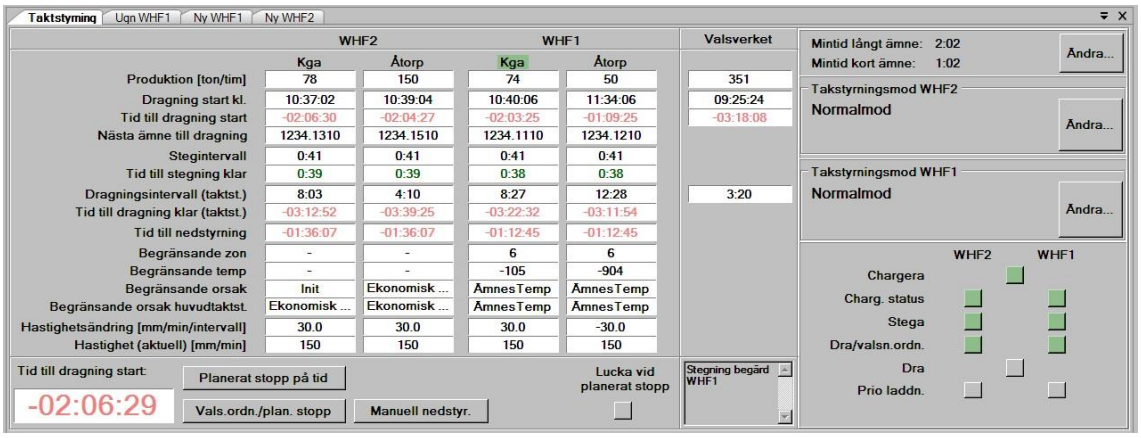
Figur 26: Temperaturstyrning i Prevas FOCS.

Temperaturstyrningen har helt designats om som kan ses i Figur 27. Istället för att ha de olika zonerna som rader har de valts att visas som kolumner. Detta för att på ett bra sätt kunna expandera zonbegränsningslinjerna till att även visuellt visa vilken temperaturstyrning som sköter vilken zon. Inuti zonbegränsningen sker nu också alla temperaturinställningar. Detta för att underlätta användarens minneshantering och översikt. Om operatören råkar trycka på fel zon eller blir störd och förlorar fokus får operatören direkt en visuell påminnelse vilken zon vars inställningar han håller på att förändra. Det finns nu även en stor ”Avbryt” knapp som operatören kan trycka på för att nollställa alla ändringar om fel zon valts. I Figur 27 har operatören valt att byta mod i zon 1 till Tabell.



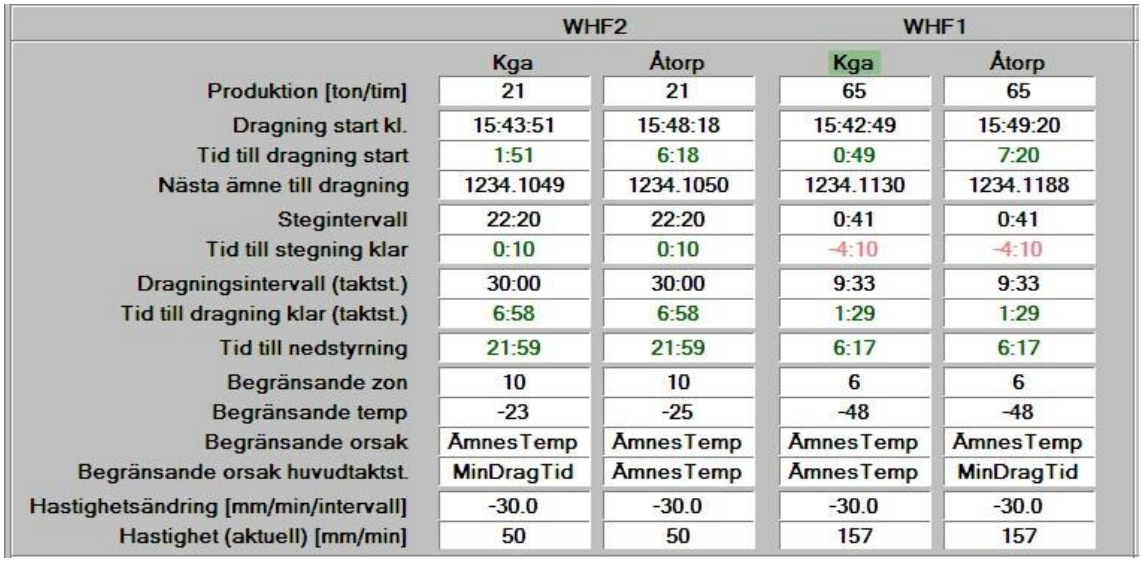
Figur 27: Nya temperaturstyrningsbilden.

Den andra bilden som operatörerna använder sig av är taktstyrningen. Denna bild använder operatörerna för att övervaka processinformation samt att byta vilken modell som ugnen styrs av om temperaturstyrningen är inställd på auto som i Figur 26. En översikt av taktstyrningen syns i Figur 28.



Figur 28: Taktstyrning i Prevas FOCS.

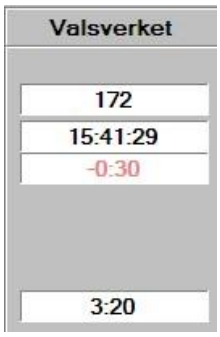
Taktstyrningsbilden visar 15 rader med olika informationsrutor, beroende på vad det är för typ av ugn finns det olika antal kolumner. Dessa beskrivs utförligare i Bilaga 1: Prevas FOCS funktioner. Figur 29 visar en taktstyrning över två olika ugnar, WHF2 och WHF1. Båda ugnarna har två rader med ämnen, Kga och Åtorp som båda har sin egen taktstyrning. Hur många ämnen som får plats i en ugn varierar.



	WHF2		WHF1	
	Kga	Åtorp	Kga	Åtorp
Produktion [ton/tim]	21	21	65	65
Dragning start kl.	15:43:51	15:48:18	15:42:49	15:49:20
Tid till dragning start	1:51	6:18	0:49	7:20
Nästa ämne till dragning	1234.1049	1234.1050	1234.1130	1234.1188
Stegintervall	22:20	22:20	0:41	0:41
Tid till stegning klar	0:10	0:10	-4:10	-4:10
Dragningsintervall (taktst.)	30:00	30:00	9:33	9:33
Tid till dragning klar (taktst.)	6:58	6:58	1:29	1:29
Tid till nedstyrning	21:59	21:59	6:17	6:17
Begränsande zon	10	10	6	6
Begränsande temp	-23	-25	-48	-48
Begränsande orsak	ÄmnesTemp	ÄmnesTemp	ÄmnesTemp	ÄmnesTemp
Begränsande orsak huvudtaktst.	MinDragTid	ÄmnesTemp	ÄmnesTemp	MinDragTid
Hastighetsändring [mm/min/intervall]	-30.0	-30.0	-30.0	-30.0
Hastighet (aktuell) [mm/min]	50	50	157	157

Figur 29: Taktsyrningsinformation i Prevas FOCS.

Gröna siffror här betyder att det är en nedräkning tills något ska hända, till exempel när nästa dragning kommer att ske. Om en nedräkning har skett utan att händelsen inträffat byts färgen till rött och tiden börjar räkna hur längesedan det var händelsen skulle ha inträffat. Svarta värden är statiska värden som till exempel med vilket intervall balkarna kommer stega fram ämnena inuti ugnen.



Figur 30: Information om valsverkets produktion i Prevas FOCS.

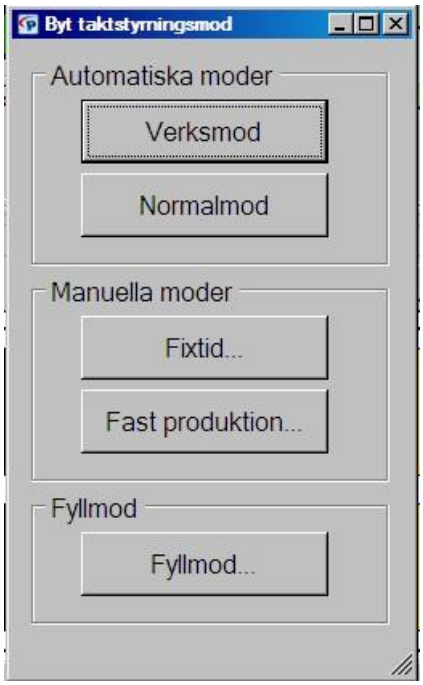
Valsverket är stationen som ligger direkt efter omvärmningsugnen, därför har man valt att även ha med information om hur lång tid det är kvar tills valsen är ledig för ett nytt ämne från ugnen. Eftersom valsen inte är lika intressant ur operatörernas synvinkel finns det bara med fyra olika informationsrutor i denna kolumn som ses i Figur 30

Taktstyrningsbilderna har i den nya designen förlorat viss funktionalitet som finns i Prevas FOCS. Dessa funktioner bedömdes mindre viktiga för operatörerna, och skulle förmodligen försvinna från framtida versioner av Prevas FOCS (Söderlund 2014). Resterande funktioner har delats upp i flertalet nya bilder för att öka anpassningsmöjligheterna precis som på ugnsöversiktsbilderna.

Figur 31: Nya bilden för taktstyrningsinformation.

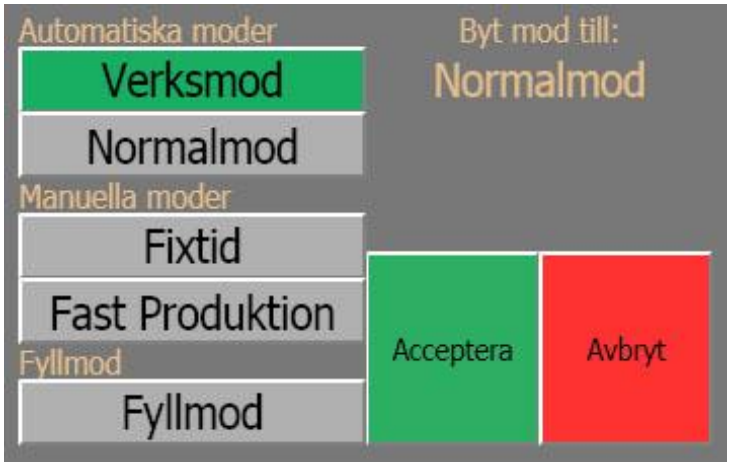
Figur 31 är identisk med Prevas FOCS version förutom färgschemat, detta eftersom detta är rader med nödvändig information utan några inställningsmöjligheter. Därav finns det inget utrymme för förändring.

På den högra sidan av taktstyrningen finns två olika knappar för att välja vilka takter som Prevas FOCS styr efter. I Figur 28 syns dessa på höger sida tillsammans med en knapp för att ändra mintider mellan dragning av ämnen. Om operatören trycker på en av knapparna för byte av taktstyrningsmod kommer ytterligare ett fönster upp där taktmoder kan väljas. De olika taktmoderna syns i Figur 32 och beskrivs utförligare i Bilaga 1: Prevas FOCS funktioner.



Figur 32: Bild för byte av taktstyrningsmoder i Prevas FOCS.

Byte av taktstyrningsmoder sker nu på ett sätt som mer liknar byte av temperaturstyrning. Istället för att man får en popup utvidgas menyn för att ge en bättre överblick som syns i Figur 33. Även här finns det en avbrytknapp om operatören skulle ha råkat klicka på fel mod eller ångrar sig.



Figur 33: Nya bilden för byte av styrningsmod.

Figur 34: Ändring av mintider i Prevas FOCS.

Mintider ger operatören en möjlighet att ställa in en minimum tid som måste passera mellan ett ämne har tagits ur ugnen. Det finns fyra olika rutor, minuter och sekunder för både långa och korta ämnen. Hur detta ser ut i Prevas FOCS syns i Figur 34.

Den popup som ändring av mintider i Prevas FOCS använder sig av är också borttagen för ett mer konsekvent användargränssnitt. Den nya bilden kan ses i Figur 35. Man behöver dock fortfarande ställa in både minuter och sekunder för både långa och korta ämnen även om man bara velat ändra en av dem. Detta på grund av hur funktionen för detta i Prevas FOCS server är designad.

Figur 35: Nya bilden för mintidsändring.

Längst ner till höger i Figur 28 kan operatören styra och övervaka ugnens steg och dragfunktion, det vill säga huruvida ugnen får stega fram ämnen genom ugnen och om ämnen får lämna ugnen. Om knapparna är gröna betyder det att funktionen är aktiverad, om funktionen inte är aktiverad är den grå. Stegfunktionen kan också bli röd om funktionen är stoppad av antingen ett meddelande från PLC, att ämnet är positionerat i en vänteposition men inte redo att dras, att verket inte är redo för att ta emot ett nytt ämne eller att det finns ett ämne som har prioritering och kommer att gå före i kön. För att ett ämne ska kunna gå före ett annat i kön krävs det att det finns ämnen som ligger parallellt i ugnen. Knapparna ses i Figur 36.

Figur 36: Stegningsbild i Prevas FOCS.

Bilden för att tillåta ugnen att dra eller stega de ämnen som finns inuti ugnen är uppdaterad så att texten är inuti bilden men funktionen är densamma. Även här är färgschemat enligt ABB:s metals guide. Detta kan ses i Figur 37.

Figur 37: Nya bilden för tillåtande av stegning och dragning.

Resterande funktioner som finns längst ner till vänster i Figur 28 kommer att försvinna i nästa version av Prevas FOCS och kommer därför inte att förklaras.

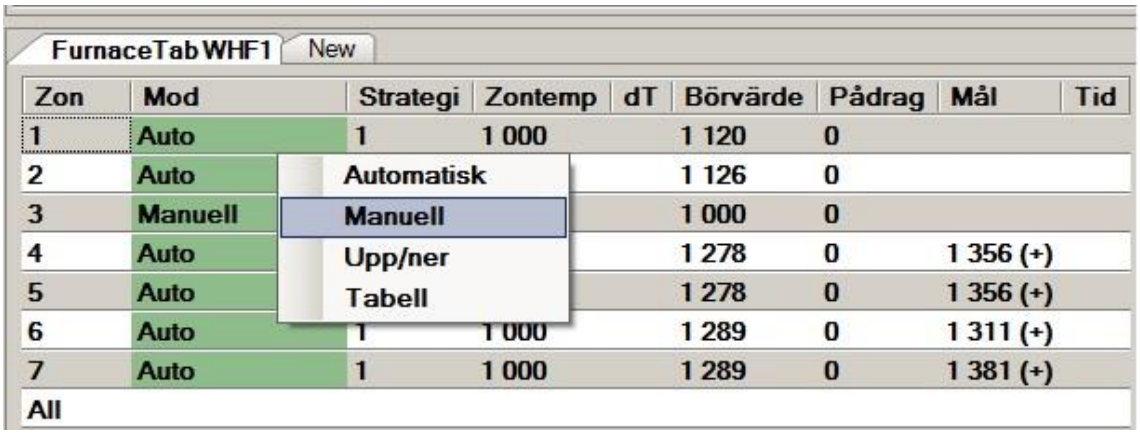
5.2 Test och utvärdering av systemet

För att testa och utvärdera systemet har dess olika delar kontinuerligt testats när ny funktionalitet lagts till. Syftet med detta är att se till att varje ny funktionalitet fungerar tillfredsställande, det vill säga att systemet går att styra som i Prevas FOCS klient fast via 800xA:s HMI. Dessutom har olika lösningsalternativ diskuterats med handledarna på både Prevas och ABB för att utreda om någonting kan göras annorlunda. Ett exempel på detta är att simuleringen av systemet endast använder sig av enkla zoner, medan vissa ugnar använder dubbla zoner där ämnet kan värmas både ovan- och underifrån. Därför är systemet designat så att installatören enkelt kan ställa in denna funktionalitet. Det finns dock ytterligare ugnsdesigner som gör det möjligt att ställa in zonerna för att snedställa uppvärmningen av ämnena. Denna möjlighet finns dock inte implementerad i prototypen, men skulle utan problem kunna implementeras som en extra funktion utan att ändra för mycket i designen. Det finns även fall då problem uppstår när både Prevas FOCS klient och den nya designen i 800xA används samtidigt för att styra samma funktion. Detta beror på den valda designen av OPC-signaler där exempelvis en signal styr alla ugnszoner. Om denna signal ändras för samma zon i båda systemen ungefär samtidigt kommer systemet att ignorera den första inställningen. Detta kommer dock troligtvis inte innebära några problem eftersom tanken är att endast det nya HMI:et kommer att användas och att det oftast är endast en operatör som styr ugnsfunktionerna. Det finns dock även möjlighet att göra designen säkrare genom att införa nya OPC-signaler som styr varje zon separat, vilket gör att risken för ”signalkrockar” minskar. Detta innebär dock att komplexiteten på lösningen också ökar något och det blir en

fråga om prioriteringar huruvida det bedöms värt att göra i en framtida kommersiell version av systemet.

Eftersom många av kraven som framkom på systemintegreringen var av icke-funktionell karaktär blir valideringen av den framtagna integrationslösningen till viss del subjektiv. Detta eftersom kraven i vissa fall inte varit särskilt specifika. Till exempel är det svårt att objektivt testa att ”systemen är så åtskilda som möjligt”. Sådana krav har istället fungerat som en bakomliggande princip vid systemdesignen, där alternativ som av oss ansetts uppfylla kravet bäst har använts. I några fall har dessa val dock fått avvägas mellan exempelvis enkelhet av lösning och ekonomiska faktorer. Att bilderna ska vara generella för alla industrier är i praktiken svårt att uppnå eftersom variationen mellan industrier skiljer sig markant från fall till fall. Lösningen som används i prototypen är standardiserad för ungefär 50 ämnen, men kan enkelt utvidgas. Det är även förberett för 10 olika ugnszoner med möjlighet för dubbla zoner, det vill säga att man har värmeelement både under och över ämnet. Omvärmningsugnar kan dock skilja sig i storlek och konfiguration då ingen standard finns på dessa. Detta innebär för lösningen att det exempelvis för design av ugnsbilden kan vara fördelaktigt att anpassa den efter hur ugnen ser ut i den specifika industrin där lösningen används. Att kunna använda exempelvis 100 ämnen när det bara behövs 20 kan då istället göra att systemet blir onödigt belastat och att det blir svårt att få en bra överblick vid installation av systemet. Därför rekommenderas viss individanpassning för att lösningen ska vara lätt att hantera vid installation och även passa operatören bättre.

Designen av systemet har även skett utifrån designprinciper framtagna för att underlätta för mänsklig interaktion med IT-systemet. Detta har inneburit att vissa bilder har designats annorlunda än hur originalet sett ut i Prevas FOCS för att ta hänsyn till dessa principer. Detta illustreras med två exempel: temperaturstyrningsbilden och taktstyrningsbilden. I Figur 38 ses Prevas FOCS temperaturstyrningsbild där operatören ska byta värmningsmod från automatisk till manuell.

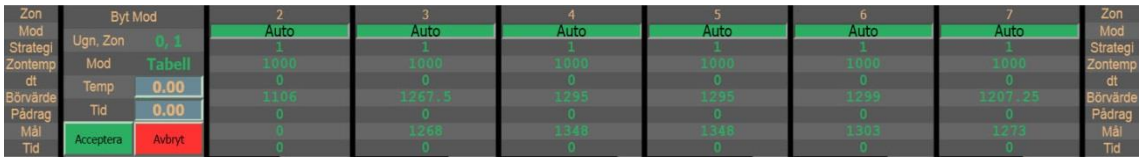


Figur 38: Byte av värmningsmod i Prevas FOCS.

I Figur 38 vill operatören byta värmningsmod i zon 1 från automatisk till manuell uppvärmning. Det finns några faktorer i denna bild som kan göra denna uppgift problematisk för operatören. För det första är det inte självklart vilken zon som operatören i detta fall har markerat. I en stressfull situation skulle lätt förväxlingar kunna ske där operatören byter mod i fel zon. Ett annat problem är att popup-fönstret täcker information som kan vara relevant för operatören. I detta fall kanske

zontemperaturen ska uppdateras till samma temperatur som i zon 3. Det går dock inte, på grund av det täckande popup-fönstret, att se temperaturen i denna zon.

I den nya temperaturstyrningsbilden, som kan ses i Figur 39, har Molich och Nielsens (1990) designprinciper legat till grund för designen.



Figur 39: Den nya temperaturstyrningsbilden med designprinciper i åtanke.

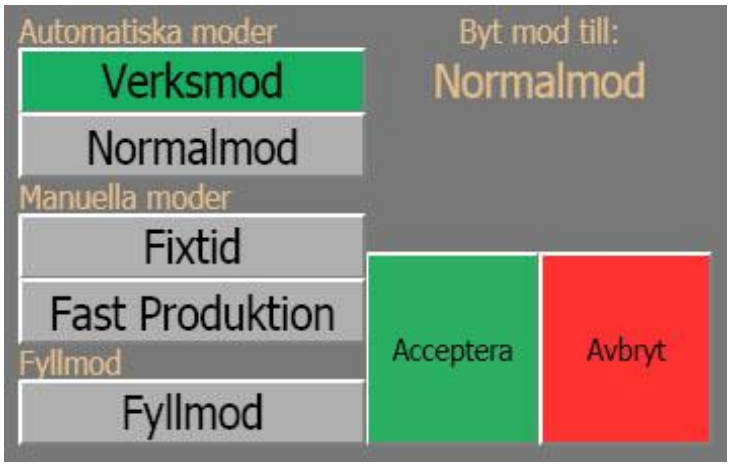
Som ses i Figur 39 blir det tydligare vilken zon som har valts i och med att popup-fönstret endast täcker just den zon som operatören vill modifiera. Detta enligt designprincipen ”visibility of system status” som innebär att operatören alltid ska ha kontroll och få feedback på vad som händer i systemet. En annan princip som följs är ”user control and freedom” som säger att användaren inte ska låsas till ett val utan enkelt kan förstå var den befinner sig och bli uppmärksammad på om fel har begåtts och hur man i sådana fall kommer tillbaka. Ytterligare en princip som följs är ”match between system and the real world” i och med att zonerna nu är placerade i kolumner istället för rader. Detta följer ugnens design och gör det lättare att se hur zonerna är kopplade till ugnen precis som i ugnsbilden (jämför med Figur 19).

I det andra exemplet vill operatören byta taktstyrningsmod. Detta visas i Figur 40.

Figur 40: Byte av takstyrningsmod till fix tid i Prevas FOCS.

Som kan ses i Figur 40 består denna bild också av flera popup-fönster på varandra som täcker bakomliggande information. Dessa fönster är dessutom inte konsekventa i sin design. Till exempel finns en ”avbrytknapp” på det översta popup-fönstret som gör det enkelt för operatören att avbryta om den har gjort fel. Däremot finns ingen sådan knapp på det undre popup-fönstret, utan där måste operatören använda krysset för att stänga fönstret. Har operatören valt fel mod från det undre popup-fönstret och trycker ”avbryt” stängs dock båda popup-fönstrena och operatören måste återigen trycka på en knapp för att få upp rutan för byte av taktstyrningsmod igen.

Den nya designen, som kan ses i Figur 41, är även den designad med de olika användarvänlighetsprinciperna i åtanke.



Figur 41: Den nya bilden för byte av taktstyrningsmod.

Som ses i Figur 41 är även här bilden designad utan popup-fönster som blockerar information. Dessutom behöver operatören inte trycka på någon ytterligare knapp om fel mod har valts och alla modval kan ses och användas utan att trycka på någon knapp. Som ses i Figur 39 och Figur 41 följer nu båda bilderna samma design, vilket följer designprincipen ”consistency and standards” som säger att användaren inte ska behöva använda flera olika arbetsmetoder för att lösa liknande uppgifter. Dessutom är det med den här designen lättare att se vad som kommer att hända när ”Avbryt” trycks. Det följer också designprincipen ”be consistent”.

Generellt har designprinciperna använts som en grundfilosofi vid bildbyggandet, samtidigt som ABB:s stilguide metals design använts. En viktig aspekt har varit att alla bilder ska gå att anpassa dynamiskt både i storlek och placering. Detta ger frihet för operatörerna att själva bestämma på vilket sätt HMI:et ska vara uppbyggt för att passa individuella preferenser. Detta följer designprincipen ”flexibility and efficiency of use” som säger att avancerade användare hjälps av möjligheter att kunna modifiera vanliga inställningar. För alla bilder som kräver operatörens interaktion för att ändra inställningar i systemet har även designprinciperna ”visibility of system status” och ”provide feedback” använts. Dessa principer går ut på att användaren av systemet snabbt ska uppfatta att gjorda ändringar har registrerats av systemet.

Vid testning av dessa funktioner framkom det att 800xA har som grundinställning att OPC-signaler ska uppdateras med 3000 millisekunders intervall. För funktioner som bara visar information är denna inställning tillräcklig, då dessa inte kräver att operatören omedelbart får uppdaterade värden. Vid funktionerna som kräver interaktion blir denna tid dock alldeles för lång. För dessa signaler ändrades istället uppdateringsfrekvensen till 500 millisekunder. Denna tid användes som en avvägning mellan prestanda (som påverkas negativt av ökad uppdateringsfrekvens) och användarvänlighet. Uppdateringsfrekvensen går dock att ändras godtyckligt efter behov.

En viktig designprincip som har diskuterats vid designen är ”aesthetic and minimalist design”, som bygger på principen ”simple and natural dialogue” och säger att all information som visas ska vara relevant för användaren. Viss information som visas för operatören i taktstyrningsinformationen kan behöva ses över för att avgöra om den är viktig att alltid visa eller om den kan läggas på något annat ställe som operatören kan nå vid behov. Denna information kan också behöva anpassas till den specifika anläggningen som använder sig av systemet, då skillnader i informationsbehov kan

förekomma mellan dessa. Informationsbehoven hos operatörer skulle behöva utredas vidare för att kunna avgöra vilken information som behövs och vilken som kan utelämnas.

6 Diskussion

Resultatet av studien och byggandet av prototypen påvisar vissa områden som är värda att diskutera. Först och främst anser vi att integrationsmetoden som presenterats och använts i denna studie passar väl in när två IT-system ska integreras. Den är utförlig och därmed ganska avancerad, vilket gör att den med modifikation förmodligen passar för flera olika integrationsprojekt av varierande komplexitet. För alla integrationer behövs dock inte alla moment som modellen beskriver utan den bör anpassas efter vad som eftersträvas. I vårt fall, med redan existerande system, ansågs steg 1 och 2 som mindre viktiga eftersom arbetet skett mest bottom-up. De olika systemens karaktärer har givetvis påverkat utfallet av denna studie. De viktigaste förutsättningarna för att integrera två redan fungerande system har visat sig vara deras befintliga möjligheter att kommunicera. Generellt sett kan konstateras att följandet av kommunikationsstandarder är viktigt för att smidigt kunna integrera två system. Från Prevas sida hade det givetvis varit smidigast om 800xA hade stöd för WCF då detta hade gjort att knappt något måste modifieras för att integrera systemen. Marknadsmässigt bedömer vi det dock inte rimligt att implementera detta, i alla fall så länge WCF inte är en kommunikationsstandard i branschen. Det är därför bra att OPC har tagits fram som en standard för kommunikation i industriella nätverk.

Prevas använder sig däremot inte av OPC A&E, vilket försvårar för integrationen av larmen. För att integrera larmen i 800xA måste OPC A&E användas, eftersom detta är det enda som stöds i 800xA. Larmhanteringen är också en så pass viktig del för operatörernas arbete och arbetsplatsens säkerhet vilket gör det viktigt att larmen visas på ett enhetligt sätt. Vi anser därför att Prevas bör anpassa FOCS till OPC A&E-standarden. Det går visserligen, som tidigare nämnt, att göra om larmen till OPC-taggar för att på så sätt kunna använda OPC A&E. Detta alternativ kommer dock att kräva större underhållsarbete, medan en full övergång till OPC A&E är mer framtidssäkert. Larmhanteringen ser dessutom ur användarens perspektiv rent grafiskt nästan identisk ut som om den vore gjord med hjälp av OPC A&E. Detta borde göra övergången enklare samtidigt som systemet också fortsättningsvis kan säljas separat med minimal påverkan på befintligt HMI.

Ytterligare en viktig lärdom dragen av prototypbygget är att systemens egenskaper bidrar till hur lösningen bör utformas. ABB:s styrsystem är som level 1-system bra i miljöer där ingenjören vet vilka funktioner som krävs av systemet och det är lätt att bygga grundläggande funktionalitet. Däremot finns begränsade möjligheter att modifiera förinstallerade strukturer. Prevas FOCS, som level 2-system, kan istället modifieras relativt enkelt och det är lätt att lägga till helt ny funktionalitet. Vid prototypbygget strävade vi efter att informationen så mycket som möjligt skulle föras över i SI-enheter. Detta eftersom vi ansåg att det skulle vara enklast att omvandla dessa till för operatören önskad enhet. På grund av tidigare nämnda systemegenskaper visade sig detta dock vara en dålig idé på grund av begränsningarna i 800xA. Olika datastrukturer är betydligt enklare att modifiera och strukturera i Prevas FOCS. Detta gör det dock än viktigare att hålla uppdaterade signallistor mellan systemen så att klarhet görs i med vilka enheter värdena visas.

Det finns ofta många faktorer att ta hänsyn till vid framtagning av integrationslösningar. Dessa står även ofta i konflikt med varandra. Exempelvis måste hänsyn tas till ekonomiska faktorer, användarvänlighet och komplexitet av integreringen och ofta måste kompromisser göras mellan dessa. Vi kan dock, som tidigare nämnt, konstatera att det är fördelaktigt att återanvända så många redan existerande resurser som möjligt. I alla fall så länge det inte går ut för mycket över användarvänligheten. Här är det även väldigt viktigt att ha en så tydlig kravspecifikation som möjligt, vilket kan vara svårt att uppnå då det inte är säkert att kunden vet exakt vad de vill ha. I detta fall har även två företag behövt samarbeta vilket gör det ännu svårare att få en specifik och detaljerad kravspecifikation av integrationen. Vi är medvetna om att många av kraven som togs fram i denna studie kan anses vara för ospecifika. Detta har påverkat resultatet i och med att valideringen av systemet till viss del kan ses som subjektiv. Däremot anser vi att valen gjorda i dessa fall har motiverats. Dessutom kan otydligheten i kravspecifikationen i detta fall anses okej i och med att de båda systemen redan är fullt fungerande och de viktigaste kraven, i form av exempelvis säkerhet och tillförlitlighet, redan är uppfyllda. Hade det varit ett nytt system som skulle utvecklas hade dock en så pass generell kravspecifikation troligtvis varit katastrofal.

Vi anser även att MDI-perspektivet är en viktig del när ett nytt system ska designas. Valet att utgå från Molich och Nielsens (1990) designprinciper gjordes för att de är generella och ger en bra grund för att skapa ett användargränssnitt. Däremot hade de fördelaktigt kunnat komplementeras med en mer grundlig användarstudie för att få en bättre inblick i vad operatörer i valsverk faktiskt behöver. Tyvärr var operatörerna inte tillgängliga för oss i detta projekts genomförande och en större användbarhetsstudie ryms inte i studiens omfång. Vi är dock övertygade om att projektet hade dragit nytta av en mer omfattande användbarhetsstudie. Detta är något som troligtvis ofta inte prioriteras i IT-projekt, mycket på grund av att de tar tid att utföra och kostar för mycket. Detta anser vi är synd då det är svårt för ingenjörer att veta vad som är viktigt för användarna av systemet.

Vi vill även poängtera att den framtagna prototypen just är en prototyp och att den därmed inte har testats lika utförligt som ett färdigt system. Det är därför möjligt att vissa buggar förekommer. Säkerhetsaspekterna av systemen är heller inte implementerade i prototypen. Dessa innefattar möjligheten att begränsa befogenheter för olika användare av systemet. I prototypen har alla användare befogenhet att ändra all funktionalitet. Syftet med prototypen är att den ska användas i marknadsföringssyfte, där grundläggande funktionalitet ska kunna demonstreras för kunder av systemet. För detta fungerar prototypen enligt oss bra.

7 Slutsatser

Den funktionalitet som bedömdes vara mest relevant för en operatör har i detta projekt överförts från Prevas FOCS till 800xA där en fungerande prototyp har implementerats. Rekommendationer vad gäller implementeringen av larm bygger på att Prevas börjar använda OPC A&E som standard för larmhantering. Vi rekommenderar även att en större användbarhetsstudie utförs på operatörerna som ska använda systemet för att vidare utreda om bilderna fungerar som avsett. Dessutom bör utredas vilka funktioner och vilken information som är viktigast för operatörerna för att kunna avgöra vilken information som ska visas och var den bör visas. Att utföra användbarhetsstudier skulle inte bara gynna det integrerade systemet utan även de båda systemen separat.

Generellt för en integrering av två IT-system anser vi att redan existerande resurser bör användas så långt som möjligt och att kommunikationsstandarder bör följas. Dessa faktorer bidrar även till att hålla nere kostnaderna för integrationsprojektet.

8 Vidare forskning

Som fortsättning på den utförda studien finns vissa delar som kan utvecklas och viss funktionalitet hos prototypen som kan förbättras. Därför vill författarna avslutningsvis sammanfatta och lista några framtida möjligheter för integreringen av Prevas FOCS och 800xA.

- Den enligt oss kanske viktigaste faktorn för ett väl fungerande integrerat system är att utföra en utförlig användbarhetsstudie där operatörerna får säga sitt om design och funktionalitet.
- En annan viktig funktion i det integrerade systemet är larmhanteringen. För att underlätta denna i framtiden rekommenderar författarna att Prevas implementerar standarden OPC A&E i FOCS.
- Prototypen saknar idag viss funktionalitet som är viktig för en färdig produkt. Det saknas idag stöd för dubbla zoner och sidledszoner där zontemperaturen noggrannare kan ställas in. Det går heller inte att byta håll på hur ämnena i ugnen färdas. Detta sker i dag från vänster till höger. På grund av att simulatorn av Prevas FOCS som använts i arbetet inte stöder visning av nästa ämne för inmatning och senast dragna ämne finns denna information inte heller i prototypen. Denna information var enligt Uhlander (2014) viktig för operatörerna och författarna rekommenderar därför att den implementeras i prototypen i framtiden.
- Även om författarna tänkt på användbarhetsprinciper vid skapandet av bilder i 800xA finns rum för förbättring. Viss information som visas vid byte av moder hade kunnat döljas där den inte behövs. Exempelvis behöver operatören i temperaturstyrningsbilden inte se inmatningsrutorna för temperatur och tid när de vill byta till moderna auto eller tabell då de inte kan ställas in för dessa moder (jämför med Figur 27). Dessutom finns för vissa bilder ingen OPC-signal för befintligt värde. Detta medför att aktuellt värde i dessa bilder inte kan visas när vissa moder ska bytas. Fler liknande problemområden skulle dock troligtvis fångas upp vid en användarstudie.

9 Referenser

9.1 Böcker

House, JS. 1981, Work, stress and social support. London: Addison-Wesley.

Karasek, R., Theorell, T. 1990. Healthy work. Stress, productivity and the reconstruction of working life. New York: Basic Books.

Mahnke, Wolfgang., Leitner, Stefan-Helmut., Damm, Matthias. 2009. OPC Unified Architecture. Berlin: Springer.

Sommerville, Ian. 2011. Software engineering, 9th edition. Boston: Addison-Wesley.

Troelsen, Andrew. 2012. Pro C# 5.0 and the .NET 4.5 Framework. New York: Springer.

Vicente, Kim. 2006. The Human Factor. New York: Routledge.

9.2 Tidskriftsartiklar

Adrion, W. Richards., Branstad, Martha A., Cherniavsky, John C. 1982. Validation, Verification, and Testing of Computer Software. *Computing Surveys* 14 (2): 159–192.

Galloway, Brendan., Hancke, Gerard P. 2013. Introduction to Industrial Control Networks. *IEEE Communications Surveys & Tutorials* 15 (2): 860–880.

Isazadeh, Ayaz. 2004. Software Engineering: Integration. *Applied and Computational Mathematics* 3 (1): 56–66.

Molich, Rolf., Nielsen, Jakob. 1990. Improving a Human-Computer Dialogue. *Communications of the ACM* 33 (3): 338–348.

Vikström, Linda. 2013. Energieffektivisera på rätt sätt. *Kunden 2013 (1)*: 40–41.

9.3 Elektroniska källor

ABB. 2014. Industrial IT System 800xA Operations.
http://www.abb.se/product/seitp334/102b314866674ccdc12572bb0031ea42.aspx
(Hämtad 2014-07-29)

Control Global. 2010. What is OPC in 2010? It is all about the tag.
http://www.controlglobal.com/articles/2010/opc2010/
(Hämtad 2014-07-29)

Microsoft. 2014a. Overview of the .NET Framework.
http://msdn.microsoft.com/en-us/library/zw4w595w(v=vs.110).aspx
(Hämtad 2014-07-30)

Microsoft. 2014b. What Is Windows Communication Foundation.
http://msdn.microsoft.com/en-us/library/ms731082(v=vs.110).aspx
(Hämtad 2014-07-30)

Naturvårdsverket. 2014. Energieffektivisering i industrin.
http://www.naturvardsverket.se/Miljoarbete-i-samhallet/Miljoarbete-i-

Sverige/Uppdelat-efter-omrade/Energi/Energieffektivisering/Industrin/
(Hämtad 2014-07-29)

Nielsen Norman Group. 1995. 10 Usability Heuristics for User Interface Design.
http://www.nngroup.com/articles/ten-usability-heuristics/
(Hämtad 2014-07-30)

OPC Foundation. 2014a. What is OPC?
https://opcfoundation.org/about/what-is-opc/
(Hämtad 2014-07-29)

OPC Foundation. 2014b. History.
https://opcfoundation.org/about/opc-foundation/history/
(Hämtad 2014-07-30)

Prevas. 2010. Prevas och ABB inleder nytt samarbete.
http://www.prevas.se/nytt_samarbete.html
(Hämtad 2014-07-29)

Sandblad, Bengt. 2012. Arbetsmiljö vid IT-stött arbete. En kort introduktion.
http://www.it.uu.se/edu/course/homepage/hci/vt12/IT_AM.pdf
(Hämtad 2014-07-29)

9.4 Rapporter och övriga dokument

ABB. 2012. System 800xA Alarm Management.
Tillgänglig på:
http://www05.abb.com/global/scot/scot349.nsf/veritydisplay/8b9bda6f7631107cc12579e600402930/$file/3bse067466_a_en_system_800xa_alarm_management_overview.pdf

ABB. 2013. System 800xA System introduction.
Tillgänglig på:
http://www05.abb.com/global/scot/scot313.nsf/veritydisplay/28f6f90d00299d9bc1257c2700580481/$file/3BUS095072_L_E_en_System_800xA_System_Introduction.pdf

Barkmeyer, Edward J., Barnard Feeny, Allison., Denno, Peter., Flater, David W., Libes, Donald E., Potts Steves, Michelle., Wallace, Evan K. 2003. Concepts for Automating Systems Integration. National Institute of Standards and Technology. US Department of Commerce.

Prevas. 2013. FOCS användarmanual. Internt dokument.

9.5 Personlig kommunikation

ABB. 2014b. Automation Scandinavia. Automationsmässa anordnad av ABB.

Söderlund, Rickard; handledare Prevas. 2014. Intervjuer utförda under arbetets gång.

Uhlander, Lars-Göran; handledare ABB. 2014. Intervjuer utförda under arbetets gång.

Bilaga 1: Prevas FOCS funktioner

Relevant funktionalitet för operatörer som måste överföras från Prevas FOCS till det nya HMI:et i 800xA är ugnskartan med temperaturstyrning och taktstyrningsinformationen med möjlighet att byta taktstyrningsmod. I denna bilaga presenteras dessa funktioner utförligare.

Ugnsbild och temperaturstyrning
Om man för muspekaren över ett av ämnena på ugnskartan kommer det upp en informationsruta som innehåller all data som finns om ämnet. Följande information finns tillgängligt för operatörerna:

- **CustomerId**
  Detta är ett serienummer på ett ämne så man kan spåra det.
- **Dimensioner**
  Varje ämne i ugnen har olika mått, det visas som bredd x djup x längd.
- **Vikt**
  Vikten på ett ämne i Kg.
- **Position i ugn**
  Var någonstans i ugnen ämnet är, mätt från ämnets framkant till ugnsöppningen.
- **Tid i ugn**
  Hur länge ämnet varit i ugnen.
- **Planerat stillestånd**
  Om det finns ett planerat stillestånd under tiden som detta ämne är i ugnen kommer det stå en förväntad tid här, till exempel 120 min.
- **Produktnummer**
  En siffra som berättar vilka strategier som används för att ämnet ska få rätt egenskaper.
- **Önskad dragningstemperatur**
  Vilken temperatur ämnet ska ha innan det lämnar ugnen.
- **Medeltemperatur**
  Hela balkens temperatur.
- **Yttemperatur**
  Balkens yttre temperatur.
- **Yemperaturgardient i tjockleksriktningen**
  Temperaturskillnader inuti ämnet för att övervaka att inte ämnet ofrivilligt har värmts upp ojämnt.
- **Avvikelse från ideal temperatur**
  Hur många grader balkens medeltemperatur skiljer sig från optimal temperatur, både varmare och kallare.
- **Krav på utjämningstid**
  Hur lång tid ett ämne måste ligga på sin optimala maxtemperatur för att få rätt egenskaper.
- **Krav på utjämningstemperatur**
  Den temperatur som ämnet ska ligga på under utjämningstiden.
- **Tid över temperatur**
  Hur länge ämnet varit varmare än kravet på utjämningstemperaturen.

Temperaturstyrningsbilden där de olika ugnzonerna kan styras har följande funktionalitet:

- **Mod** ställer in olika program för att värma ämnena, alternativen är: auto, manuell, upp/ner och tabell.
    - I normal drift ska auto användas och då sköter Prevas FOCS själv inställningarna för temperaturbörvärden beroende på vald produktionstakt. För att vara så noggrann som möjligt används en återkoppling.
    - Vid manuell drift ställer operatören själv in börvärden.
    - Upp/ner är också en manuell inställning med skillnaden att temperaturen ökas eller minskas till det önskade värdet under en viss inställbar tid.
    - Tabell betyder att temperaturen ställs in efter en tabell beroende på produktionstakt och produktionsnummer. Här används ingen återkoppling.
- **Strategi** visar vilken strategi som används vid uppvärmandet av ämnena, det vill säga vilket produktnummer som ämnet som värms här har.
- **Zontemp** visar den uppmätta temperaturen i en viss zon. Denna temperatur används av temperaturregulatorn.
- **dT** står för värmeskillnaden mellan balkarna inuti ugnen.
- **Börvärde** är den önskade temperaturen i varje zon. Beroende på vad den aktuella temperaturen är visas ett (+) om temperaturen behöver ökas eller ett (-) om den ska minskas.
- **Pådrag** är en procentuell indikation av hur hög bränsleförbrukning är som används för att värma ugnen.
- **Mål** visar vilken temperatur Prevas FOCS siktar emot att värma upp ämnena till. Detta betyder dock inte att det är den aktuella börtemperaturen för tillfället.
- **Tid** visar hur lång tid det kommer att ta för Prevas FOCS att rampa upp temperaturen till börvärdet. Om ingen tid visas är det på grund av att strategin är inställd på att använda maximal ramptid för att spara på bränsle.

Taktstyrning
Den andra bilden som operatörerna använder sig av är taktstyrningen. I denna finns följande information:

- **Produktion** Visar produktionstakten uttryckt i ton per timme för varje balk. Vid beräkning utgås från att alla ämnen är av standardlängd och hänsyn tas varken till verklig längd, eller verklig vikt.
- **Dragning start kl** Visar tidpunkten för nästa dragning ur ugnens synvinkel.
- **Tid till dragning start** Visar hur lång tid det är kvar till nästa dragning.
- **Nästa ämne till dragning** Visar vilket ämne som står på tur för dragning på aktuell rad.
- **Stegintervall** Visar den stegtakt systemet rekommenderar ur ugnens synvinkel.
- **Tid till stegning klar** Visar hur lång tid det är kvar till nästa stegning.
- **Dragningsintervall (taktst.)** Visar den takt systemet rekommenderar ur ugnens (ämnenas) synvinkel.
- **Tid till dragning klar (taktst.)** Visar hur lång tid det är kvar till nästa dragning ur ugnens (ämnenas) synvinkel.
- **Tid till nedstyrning** Visar hur lång tid det är kvar innan ugnen går in i automatisk fördröjning.
- **Begränsande zon** Visar vilken zon i ugnen som just nu begränsar takten.
- **Begränsande temp** Visar hur stort temperaturfel som ger aktuell begränsning.

- **Begränsande orsak** kan variera mellan följande alternativ beroende på vad som är den begränsade orsaken.
    - **EjTaktstyr** Ingen taktstyrning beräknas.
    - **Init** FOCS-servern har nyligen startats och befinner sig fortfarande i initieringsfasen. Infaller även efter automatisk fördröjning.
    - **ÖkaTakt** Inga begränsningar finns.
    - **ÄmnesTemp**. Låg ämnestemperatur begränsar takten.
    - **AvgasTemp.** Avgastemperaturen har nått sitt gränsvärde.
    - **Valsverket** Valsverket begränsar takten.
    - **MinDragTid** Takten har nått kravet på lägsta dragningsintervall.
    - **MaxProd** Takten har nått kravet på högsta produktionstakt.
    - **PyroTemp** Takten begränsas av för låg valsningstemperatur.
    - **Hög yttemp.** Något ämnes yttemperatur begränsar takten.
    - **Skenmärken** Takten begränsas av risk för skenmärken.
    - **TempGrad.** Temperaturskillnaden i tjockleksriktningen begränsar takten.
    - **UpplösnTid** För korta upplösningstider begränsar takten.
    - **UtjämnTid** För korta utjämningstider begränsar takten.
    - **Låg hastighet vänta** Låg hastighet, vänta istället.
    - **Snabbast fylltid** För snabb fyllning i fyllmod.
    - **Begärd tid** Fyllning på begärd tid i fyllmod.
    - **Ekonomisk fyllningshastighet** Fyllningshastighet i ekonomisk fyllmod.
- **Begränsande orsak huvudtaktst.** Visar begränsande orsak i huvudtaktstyrningen.
- **Hastighetsändring** Visar hur mycket ämneshastigheten önskar ökas/minska uttryckt i millimeter per minut per intervall.
- **Hastighet (aktuell)** Visar den ämneshastighet i millimeter per minut som systemet rekommenderar ur ugnens synvinkel.

Vid byte av taktstyrningsmoder finns följande val:

- **Verksmod** betyder att det är valsverket som har högsta prioritet. Detta innebär att Prevas FOCS styr ugnen på ett sådant sätt att det kommer finnas ett ämne varmt nog att valsas direkt när valsen är ledig. Ugnens begränsningar sätts här av dess fysiska egenskaper och kraven på hur länge ämnen ska vara i en viss temperatur innan dragning frånses.
- **Normalmod** betyder att ugnen styrs efter andra egenskaper som till exempel optimal energiåtgång. Valsens produktion tas fortfarande in i beräkningarna men nu kommer ämnena istället inte bli uppvärmda så snabbt som möjligt även om valsen inte används.
- **Fix tid** betyder att operatören manuellt ställer in hur ofta det ska dras ett ämne från ugnen till valsverket oberoende storlek.
- **Fast produktion** liknar fast tid men går på ton istället för på tid. Använder man fast produktion tas det bättre hänsyn till att tunga ämnen behöver värmas längre än lättare ämnen.
- **Fyllmod** använder operatörerna sig av när det inte finns några ämnen i ugnen. Operatören ställer in en tid när det första ämnet ska vara i dragningsposition. När detta är utfört måste operatören byta mod för att komma vidare till en normal produktionstakt.