



CHALMERS



Spårning i tillverkningsprocesser

Utveckling av ett spårningssystem för industri 4.0

Kandidatarbete inom civilingenjörsprogrammet

Pontus Andersson

Filip Eriksson

Jesper Hansson

Mårten Skogh

Vincent Szolnoky

John Wedin

Förord

Den här rapporten är en del av kandidatarbetet *Robotautomation - Intelligent styrning och övervakning för industri 4.0*, SSYX02-17-13, som genomfördes vid Chalmers Tekniska Högskola under vårterminen 2017. Vi vill rikta ett stort tack till följande personer:

Kristofer Bengtsson för handledning och stöd under projektets gång.

Göran Stigler för hjälp med 3D-printing.

Per Nyqvist för vägledning i PSL samt robotutbildning.

Hans Sjöberg för hjälp i PSL.

Niklas Lagergren på Teamster som tog emot oss på intervju.

Projektkoden finns på GitHub (<https://github.com/vinszent/TrackX>).

Sammanfattning

Världen står inför den fjärde industriella revolutionen, som innebär mer digitala, modulerbara och smartare industriella produktionssystem. En av de största fördelarna är ökad flexibilitet vilket möjliggörs av snabbare omställningar i produktionen, men i och med att uppgifterna allt mer överläts till robotar och andra automatiserade system, växer även behovet av god insyn i tillverkningsprocesserna för att kunna förebygga och upptäcka fel. I den här rapporten utreds olika alternativ för att spåra detaljer i ett produktionssystem, speciellt utvecklas ett spårningssystem som gör just detta i en testmiljö. Fokus ligger på att få fram så mycket användbar data om processen som möjligt samt att visualisera datan för operatören på ett begripligt och överskådligt sätt.

Spårningssystemet som utvecklats består av ett användargränssnitt som innehåller ett Gantt-schema, en karta av produktionssystemet samt tabeller med historik av händelser och nyckeltal. Alla dessa delar visualiserar datan på ett begripligt sätt för operatören. För att få fram denna data har QR-koder fästs på detaljerna. Koderna har sedan identifierats med hjälp av kameror utplacerade i produktionssystemet. Även en koppling mot en PLC har upprättats där information om systemet kan inhämtas.

Abstract

The world is on the verge of a fourth industrial revolution, which entails more digital, modular and smarter industrial production systems. One of the main benefits is improved flexibility, which is enabled by quicker changeovers in the scheduled production. However, because more and more of the tasks are left to robots, there is a need to be able to monitor the processes for the sake of error detection and prevention. This report investigates different alternatives for tracking components in an industrial environment, and in particular a monitoring system is developed that is capable of doing this in a test environment. The main focus is gathering as much, for the operator, useful data about the production process and its components as possible and visualizing that data for the operator in a way that is intuitive and provides a good overview.

The tracking system that was developed contains a user interface with a Gantt chart, a map of the production system and tables containing the history of system events and key performance indicators. All these parts visualize the data for the operator in an intelligible way. The data was obtained by attaching QR-codes to the parts and tracking them with cameras in the production system. As well as a connection to the PLC unit was established in order to retrieve the production system data.

Beteckningar

Beteckning	Beskrivning
Buss	Kommunikationsnätverk där man kan ta emot och lägga upp information
C	Systemnära programmeringsspråk
Canvas	Programmeringsgränssnitt för grafisk ritning
Cell	Samling av kompletta automatiserade system som jobbar gemensamt i en produktionsmiljö
Detalj	Någonting som bearbetas eller hanteras i en tillverkningsprocess
GUI	<i>Graphical User Interface</i> (Grafiskt användargränssnitt), det som användaren av mjukvaran ser och interagerar med
HTML5	En ny standard av märkspråket <i>HyperText Markup Language</i>
HTTP	<i>Hyper Text Transfer Protocol</i> , kommunikationsprotokoll som används för kommunikation med webbservrar
Industri 4.0	Används synonymt med den fjärde industriella revolutionen
Java	Objektorienterat programmeringsspråk
Kafka	Distribuerad buss för att skicka meddelande
KPI	<i>Key Performance Indicator</i> , nyckeltal som används för att utvärdera ett system eller process
NFC	<i>Near Field Communication</i> , trådlös kommunikationsteknik som kan användas över korta avstånd
PLC	<i>Programmable Logic Controller</i> , industriadator som styr industriella tillämpningar
PSL	<i>Production System Laboratory</i> , laboratoriet på Chalmers med cellen och robotarna
QR-kod	<i>Quick Response Code</i> , avancerad form av streckkod
Raspberry Pi	En typ av enkortsdator
RFID	<i>Radio Frequency Identification</i> , en typ av NFC
Socket	Programmeringsgränssnitt för nätverkskommunikation
SP	<i>Sequence Planner</i> , mjukvara för att styra och övervaka olika typer av fysiska processer
SQL	<i>Structured Query Language</i> , ett programspråk som används för att skapa och använda databaser

Innehållsförteckning

1	Introduktion	1
1.1	Bakgrund	1
1.1.1	Industriella revolutioner	1
1.1.2	Industri 4.0	2
1.1.3	Spårning	3
1.1.4	Dataframställning	4
1.2	Syfte	4
2	Problembeskrivning	5
2.1	Vilken data ska samlas in	5
2.2	Hur ska datan samlas in	5
2.3	Hur ska datan användas	5
3	Produktionssystemet i PSL	7
3.1	Robotcell	7
3.2	PLC	8
3.3	Sequence Planner	8
3.4	Operatören	9
3.5	Bussen	9
4	Utveckling av identifieringssystemet	10
4.1	Problem	10
4.2	Utvärdering av identifieringstekniker	11
4.2.1	RFID	11
4.2.2	Streckkoder	11
4.2.3	QR	12
4.2.4	Val av teknik	12
4.3	QR-avkodning	12
4.4	Kommunikation med PLC:n	14
5	Databehandling	16
5.1	Val av lagringsteknik	16
5.2	Insättning i databasen	16
5.3	Förfrågningar och heuristiker	17
6	Datavisualisering	18
6.1	Gantt-schema	18
6.2	Karta	19
6.3	Historik och nyckeltal	20
7	Resultat	22
7.1	Testning	22
7.1.1	Identifieringssystemet	23
7.1.2	Användargränssnittet	23

8	Diskussion	24
8.1	Skalbarhet	24
8.2	Modularitet	24
8.3	Spårningssystemets brukbarhet	25
8.4	Estetik	25
8.5	Möjliga förbättringar av existerande funktionalitet	25
8.5.1	Spårningssystemet	25
8.5.2	Användargränssnittet	25
8.6	Möjlig ytterligare funktionalitet	25
9	Slutsats	27
	Bilaga A Intervju med Teamster	32

1 Introduktion

Tillverkningsindustrin står idag inför ett paradigmskifte, där utvecklingen går mot allt mer automation och minskad mänsklig inblandning, detta främst i de delar där människor tvingas till monotont och farligt arbete.[1] Dagens tillverkningsindustrier använder sig i stor utsträckning av automatiserade processer i vilka industrirobotar programmeras för att utföra specifika uppgifter. Då uppgifterna som automatiseras blir allt mer komplexa blir det komplicerat att få en tydlig bild över alla inblandade resurser, som till exempel robotar, maskiner, detaljer eller transportband, i processen.[2] Det blir därför intressant att undersöka hur man på ett konkret sätt kan spåra och presentera information om delar i hela produktionssystemet.

1.1 Bakgrund

Under projektets gång utfördes en intervju med en projektledare från industrin (se bilaga A "Intervju med Teamster") som hade följande att säga om den potentiella nyttan av mer spårning inom industriella produktionssystem:

”Här finns en del att göra och när man pratar om industri 4.0 så är det en stor grej att man kan få bättre information direkt. (...) Så där finns det mycket att göra och att man direkt kan peka ut att det är den där komponenten som har fallerat och hjälpa operatören. Man skulle kunna använda sig bättre av tekniken så man snabbt kan avvärja sig felen.”

Med en ökad grad av spårning skulle industrin kunna spara in på bland annat utbildning av operatörer och den tid det tar för operatörer att lokalisera fel som uppstår. Det finns helt enkelt mycket att göra inom detta område då det enligt intervjun som utfördes ofta är svårt för operatören att veta vad det är som har hänt och därmed krävs det en hel del kunskap om utrustningen. En effektivisering av detta kan leda till mindre resursutnyttjande då produktionen flyter på bättre.

1.1.1 Industriella revolutioner

Det har funnits tre industriella revolutioner: den första var den mekaniska där vatten- och ångkraft ersatte mänskligt arbete; den andra var massproduktion och det löpande bandet; den tredje, och rådande, är där datorer och automation ersätter både fysiskt, och till viss del mentalt, arbete.[3]

Den första industriella revolutionen kom i och med att produktionsindustrin genomgick övergången från att manuellt producera varor till att använda maskiner. Revolutionen började i Storbritannien och det var även den brittiska industrin som stod för majoriteten av innovationerna som gav oss den första industriella revolutionen. Den brittiska textilindustrin var den mest dominanta i termer av sysselsättning och investerat kapital. Det var även den brittiska textilindustrin som var först med att använda moderna produktionsmetoder.[4] Den första industriella revolutionen har sagts vara den viktigaste händelsen sedan domesticeringen av djur och växter.[5]

Den första industriella revolutionen övergick sedermera i den andra, även kallad den *teknologiska*, revolutionen. Det var under denna revolution som idéer kring det löpande bandet och storskalig, ny infrastruktur baserad på metaller så som järnvägsnät och telegrafnät kunde förverkligas. Det var även i samband med den andra industriella revolutionen som fabriker började elektrifieras.[6]

Den tredje, *digitala*, revolutionen kom i och med uppfinnandet och spridningen av logiska kretsar och programmerbara datorer. Utöver ökad produktivitet förde den även med sig väsentliga nya möjligheter för utbyte av tankar, idéer och information. Den digitala revolutionen är idag i en övergångsfas, i vilken den fjärde industriella revolutionen är på väg att ta över.[7]

1.1.2 Industri 4.0

Den fjärde industriella revolutionen står nu för dörren. Digitalisering och intelligenta system ska revolutionera industrin på nytt. Industrin ska bli smartare och mer flexibel. Den tyska regeringen myntade begreppet Industri 4.0, och det är just Tyskland som ligger i framkant av utvecklingen av den smarta industrin.[8] Även Sveriges regering har tagit fram en plan för uppdateringen av den svenska industrin till version 4.0.[9][10]

En viktig del i den nya industrin kommer vara den *smarta fabriken*[11], där koncept som *IIoT* (Industrial Internet of Things), *big data*[1] och *cyber-physical systems*[12] är grundläggande.

IIoT är namnet på det stora nätverk av uppkopplade apparater inom industrin som spås komma inom snar framtid. Inom IIoT ska stora delar av de framtida produktionsmaskinerna och sensorerna inom fabriken vara sammankopplade och ha möjlighet att utbyta information.[13]

Begreppet big data har många olika tolkningar, men en gemensamt för dessa är att de grundar sig i att stora datamängder ska inhämtas, analyseras och användas. En av förutsättningarna för big data är möjligheten att kunna samla in dessa stora datamängder, något som underlättas till stor del av IIoT.[14]

Med cyber-physical system menas system som bygger på den sömlösa integrationen av fysiska och datoriserade delar som är tätt sammanlänkade.[15] Exempel på sådana system är självkörande bilar, robotkirurgisystem och i en tillverkningsindustriell kontext även industriella robotar och tillverkningsutrustning.[16]

Med hjälp av dessa ska framtidens fabriker kunna producera unika produkter medan man håller kvar vid de ekonomiska fördelarna med massproduktion. Det kommer allt mer vara produkterna som styr sin egen produktion, istället för att ha de förutbestämda steg i vilka produkterna nu produceras.[8][17]

Mycket av den förändring som industrin idag ser kommer av den snabba utvecklingen av konsumentelektronik som börjar få genomslag i industrimiljöer. Billiga datorer och uppkopplingslösningar gör idag att "intelligens" kan ges till industrimaskiner och sensorer. Detta gör att en allt högre nivå av automation kan uppnås då produktionssystemen ges möjlighet att optimera sig själva och att decentraliseras till en mer distribuerad styrning.[8][17]

Den industriella tjänstesektorn utgör en stor del av Sveriges ekonomi med ca en miljon jobb och 77 % av Sveriges exportvärde[18]. Det finns alltså en stark motivation från den svenska statens sida att främja övergången till den nya typen av industri. Följande citat är tagna ur rapporter som Näringsdepartementet publicerat:

"Regeringen har tagit fram en nyindustrialiseringsstrategi som ska bidra till att Sverige ska vara världsledande inom modern och hållbar industriell produktion – en produktion som är smart, flexibel, resurseffektiv och samtidigt erbjuder en attraktiv arbetsplats."[9]

"Industrin genomgår en strukturomvandling som drivs av globalisering, digitalisering och omställningen mot en grön resurseffektiv ekonomi. Omvandlingen

mot en mer digitaliserad och hållbar industriell produktion skapar möjligheter att stärka svensk industris konkurrenskraft, som därigenom kan bidra till ökad sysselsättning och en hållbar tillväxt.”[10]

Som citaten tyder på så anser regeringen att satsningen på nyindustrialiseringen och att ligga i industrins framkant utgör en viktig del för framtida Sverige, då en väldigt stor andel av landets export utgörs av den industriella tjänstesektorn som nämnts tidigare. Detta för att bibehålla arbeten och kanske till och med öka sysselsättningen. Eftersom traditionella industriarbeten kommer att försvinna i takt med att automatiseringen ökar, är det viktigt att ligga långt fram i utvecklingen för att acklimatisera när nya jobb skapas genom att komplexiteten i varu- och tjänsteutbudet ökar.[19]

1.1.3 Spårning

En nödvändig del för att de smarta systemen i industri 4.0 ska kunna fungera är information om dels systemet självt men också detaljerna det behandlar. Att utvinna denna information är i princip vad som i detta projektet kallas spårning. Alltså avses med spårning att spåra detaljer som tillsammans utgör en produkt genom tillverkningsprocessen och att följa varje steg för detaljerna och då se vad som sker med dem i processen.

Spårning av detaljer används redan i många tillverkningsindustrier idag. Metoderna för att kunna spåra detaljer i industriell tillverkning och i varulager har utvecklats under de senaste decennierna.

En av branscherna där spårning av individuella detaljer spelar stor roll är textil- och tvätteriindustrin där man på senare tid har påbörjat arbetet med att gå från manuell inläsning av streckkoder till automatisk RFID-identifikation för plagg.[20] Mycket här handlar om att ta bort den mänskliga faktorn vilket gör systemet mer robust och dessutom snabbare. Man kan med hjälp av RFID-systemet spåra och läsa av hundratals plagg samtidigt. De individuella ID kan även användas för att enklare automatisera processer.[21]

Inom tillverkningsindustrin finns det flera existerande lösningar för att spåra detaljer och maskiner. Företaget Axxos Industrial Systems AB erbjuder produkten Axxos OEE som mäter tillgänglighet, anläggningsutnyttjande och kvalitetsutbyte. Datan samlas in genom att koppla upp sig mot maskiner för att spåra händelser i systemet. Det finns även möjlighet för operatörer att rapportera orsakskoder i systemet. Detta görs för att ge en bild över problem som uppstår i maskiner, möjliga flaskhalsar som är ett uttryck för att ett moment i produktionen stannar upp produktionsflödet och förluster i form av kassationer.[22] Axxos Visualize, som är ett tillägg till Axxos OEE, är ett webbaserat visualiseringsverktyg som visar den insamlade datan.[23] Ett annat företag som erbjuder en liknande lösning är Good Solutions Sweden AB med produkten RS Production som även den samlar in data från maskiner och operatörer för att ge en produktionsuppföljning i realtid.[24]

Men det är inte bara inom industrin som spårning har kommit till användning. Spårning av individer är något som idag används i flera butiker och varuhus runt om i världen, där man samlar in bland annat personernas vägar genom butikerna, tiden som spenderas i vissa delar av butikerna, vilka varor man kollar på, hur länge man kollar på dessa, och en del annan information. I dessa tillämpningar är det främst bildigenkänning[25] samt WiFi och BLE (*Bluetooth Low Energy*) som används för att följa kundernas mobiltelefoner i butikslokalerna.[26]

Liknande teknik har även föreslagits för användning inom vården där vårdtagare kan spåras i hemmet med hjälp av WiFi eller BLE. Man skulle med ett sådant system kunna få varningar om vårdtagaren skulle ge tecken på att befinna sig i en utsatt situation.[27]

1.1.4 Dataframställning

Under projektets gång genomfördes en intervju med en projektledare från industrin, se bilaga A, han var kritisk till att spåra komponenter på detaljnivå. Han menade att automatik är smidigt då systemet gör samma fel varje gång och till slut upptäcks det. Därmed tas den mänskliga faktorn bort om någon är trött eller har en dålig dag och gör misstag. Så spårning är nödvändigt, men det krävs dock för mycket tid och resurser för att spårning på detaljnivå ska kunna motiveras.

Han avslutar med att säga att en fördel med spårning i industrin är att det är lätt att få fram information om produktionen, och komponenter är väldigt bra på att förmedla stora mängder uppgifter om sitt tillstånd och vad de egentligen gör. Men problemet ligger på mottagarsidan då ett system ska samla in alla komponenters statusmeddelanden. Den här informationen är i dagsläget svår att få någon användning av. Då det är krångligt att presentera all data på ett meningsfullt sätt för en operatör och stor kunskap samt erfarenhet krävs när en person ska gå in och förstå sig på alla meddelanden, vilket kräver mer resurser.

1.2 Syfte

Syftet med det här projektet är att ta fram en generell lösning för att samla in och presentera information om de unika detaljerna som utgör produkter i en tillverkningsprocess, och även att integrera detta spårningssystem med styrningen av tillverkningsprocessen. Det finns bland annat tankar om att systemet ska kunna reagera på den spåringsdata som genereras för att korrigera fel eller optimera processen. Utöver de funktionella delarna ska systemet även kunna presentera data på ett tilltalande och användbart sätt för en användare eller operatör. Allt detta syftar till att effektivisera implementationen av industriautomation genom att förenkla samarbetet mellan människa och maskin, och även ge produktionssystemet möjlighet att själv kunna korrigera moment.

2 Problembeskrivning

Det finns idag produktionssystem där man inte har full översikt över hela processen. Man skulle i de fallen kunna tjäna på att införa mer översikt, dels för att få en bättre chans att upptäcka fel i tillverkningen, men även för att kunna få idéer på hur processen kan förbättras.

Ett problem är att vi endast har tillgång till ett småskaligt industrisystem att testa på, och det innebär att oväntade problem mer sällan uppstår, vilket inte alltid går att likna vid verkliga industriella miljöer där tillverkningsprocesserna snabbt blir mycket mer komplexa. Även om detaljer i tillverkningsindustrin för det mesta bearbetas av enbart robotar, sker ändå en del av monteringen manuellt, vilket leder till att de tidsskillnader som finns i industrin inte kommer visa sig på samma sätt som här i testningen i labbmiljö, eftersom robotar tar lika lång tid på sig varje gång de gör ett moment, medan det för en operatör oftast inte gäller. Problemet med detta är de olika tiderna som fås fram kommer bli samma hela tiden och kanske inte lika intressanta att undersöka då detta är en orealistisk situation.

Det är idag inget problem att samla in stora datamängder från produktionssystem. Problemet ligger istället främst i att man måste ta hänsyn till den begränsade mängd data en operatör kan ta till sig och tolka. Detta framgår tydligt i den utförda intervjun med Niklas Lagergren, se bilaga A.

Ytterligare problem uppkommer när man till exempel vill skicka, visualisera, eller använda data. Lösningförslag kommer alltså behöva testas för att utvärdera robusthet, om allt fångas av spåringsenheterna och även kapacitet, hur många detaljer som kan spåras. Det bör även undersökas och utvecklas några användningsområde för datan för att se vilken nytta man kan få ut från den.

Problemet med att spåra kan delas upp i tre delproblem: att identifiera vilken data som ska samlas in, hur man ska samla in den datan och sedan hur man ska använda datan för att uppnå önskad effekt.

2.1 Vilken data ska samlas in

För att klara av att spåra detaljer så måste det bestämmas vilka data som behöver samlas in. Detta för att inte inhämta överflödiga data som är svår att använda, det är även viktigt att tillräckliga data samlas in för att vara säker på att spårningen av detaljerna inte blir för lågupplöst. Detta leder till nästa steg i problemet där det måste undersökas vilken data som faktiskt går att samla in.

2.2 Hur ska datan samlas in

För att unikt identifiera ett objekt så finns det olika identifieringstekniker. Varje teknik har sina för- och nackdelar, och de potentiella användningsområdena för flera av dem måste utredas. Eftersom spåringsenheterna ska passa in i befintliga system måste varje teknik granskas efter eventuella hinder eller krav som kan försvåra implementeringen. Exempel på sådana hinder kan vara placering, kostnad och prestanda.

2.3 Hur ska datan användas

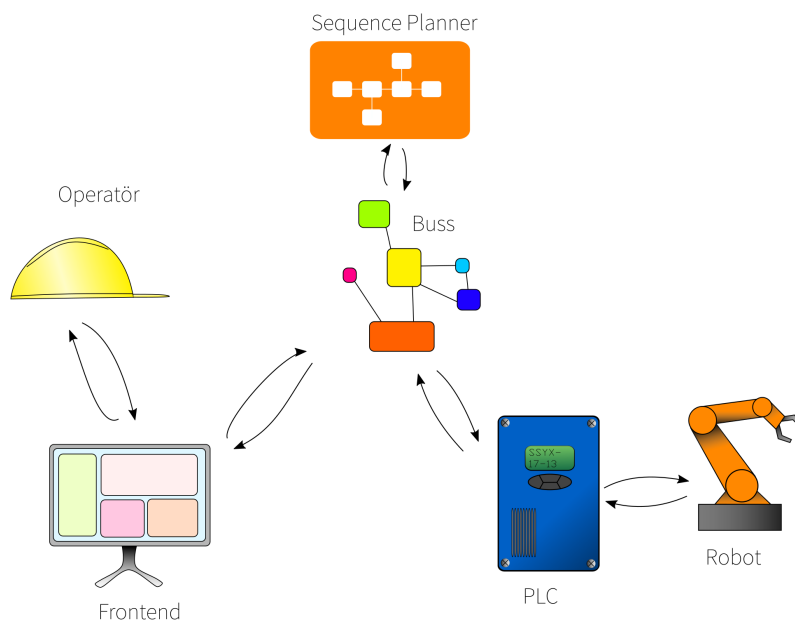
Man kan använda datan för olika visualiseringar som ger direkt återkoppling om hur produktionssystemet beter sig. Det kan vara olika scheman och grafer för den insamlade datan men det kan också vara mer kompletta visualiseringar som 3D-representation som i realtid

visar hur datorn ser systemet eller färgdiagram för att se var det går långsamt eller snabbt. En operatör skulle kunna använda den här informationen för att stänga av systemet vid kritiska fel eller för att manuellt styra det om så krävs. Ett annat användningsområde för datan är att använda det i automationsdelen av mjukvaran, där den med hjälp av datan bland annat kan lista ut vilken optimering som ska göras eller förutspå vad som kan gå fel i framtiden och automatiskt korrigera för det.

Undersökningar måste alltså göras för att se vilka sätt som passar bäst för att visualisera den data som tas fram. Hänsyn måste även tas till vem datan ska presenteras för, oftast är det kanske en operatör men inte uteslutande.

3 Produktionssystemet i PSL

Det här projektet är en fortsättning av tidigare projekt som gjorts på Chalmers Tekniska Högskola.[28][29] Under år 2015 konstruerades och utvärderades ett koncept för styrning av ett automatiserat produktionssystem.[28] Fokuset låg på utveckling av en standard som skulle koppla ihop forskningen på Chalmers Tekniska Högskola med industrins dåvarande standard för PLC-logik, och då främst den standard som fanns på Volvo Car Corporations. Projektet bestod även av framtagning av en extern utvecklingsmiljö för PLC-logik och robotprogrammering.[28] Året efter utfördes ett kandidatprojekt[29] som skapade ett webbgränssnitt för att beställa klosstorn byggda av robotar, detta webbgränssnitt ligger till grund för det arbete som utförs i detta projekt. I figur 1 visas de ingående delarna i produktionssystemet där en spåringsdel inte har implementerats och är den del som det här projektet ska skapa. Nedan följer beskrivningar av de ingående delarna i systemet.

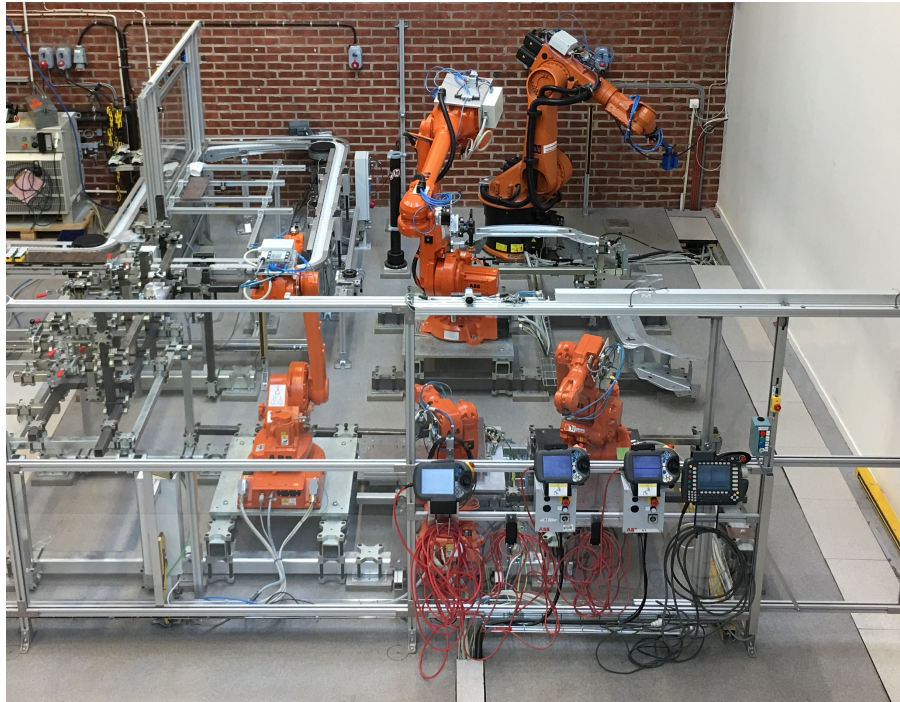


Figur 1: Ett diagram som ger en överblick av produktionssystemet.

3.1 Robotcell

Robotcellen består huvudsakligen av ett transportband med fyra hissar samt fem stycken industriella robotar, fyra stycken av märket ABB och en av märket KUKA, som kan flytta objekt från hissarna till och från olika arbetsbänkar där själva byggandet utförs. Det finns även RFID-läsare, som förklaras mer ingående i kapitel 4, monterade vid sidan av transportbandet som ger möjlighet att läsa av de objekt som är märkta med RFID-taggar. I detta projekt används transportbandet och tre av de robotarmar som finns, alla av märket ABB. I figur 2 syns hela robotceller, för att bygga klosstornen är det den stora mittersta roboten samt de två mindre robotarna framför den.

Då klosstorn beställs flyttar robot **R2** inkommande paletter till robot **R4** och **R5**:s arbetsplatser som visas i figur 3 och det krävs två paletter med klossar för att bygga ett färdigt klosstorn. Arbetsplats 1-2 tillhör robot **R4** och 3-4 robot **R5**. Paletterna placeras växelvis på arbetsplats 1-2 och 3-4. Så om robot **R2** placerar en inkommande palett på



Figur 2: En bild av robotcellen i PSL. I den bakre delen till vänster syns transportbandet och i mitten står de fem robotarmarna.

plats 1 och 3 för ett beställt klosstorn så kommer nästa klosstorn med tillhörande paletter med klossar placeras på plats 2 och 4.

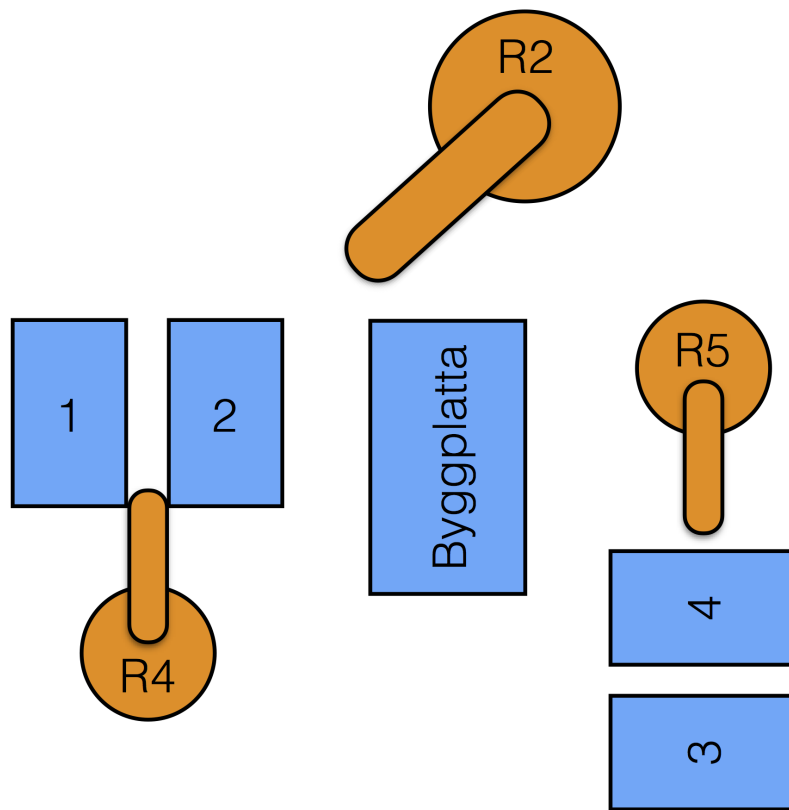
3.2 PLC

På den lägsta nivån styrs robotcellen av en *Programmable Logic Controller*, förkortat PLC. PLC:n kommunicerar via styrsignaler med alla individuella delar av robotcellen och håller även koll på vilka tillstånd alla delar befinner sig i. För PLC:n kan man definiera vad som kallas operationer, vilket är en godtycklig sekvens av grundläggande handlingar. Ett exempel på en operation skulle kunna vara att den stora roboten hämtar en palett genom att den rör sig till hissen, ner, greppar tag i en detalj, rör sig upp, bort till arbetsbänken, ner igen och till sist släpper detaljen.

3.3 Sequence Planner

En nivå ovanför PLC befinner sig *Sequence Planner*[30], förkortat SP. SP är uppdelad i två huvuddelar: SP-core och SP-gui.

SP-core hanterar planeringen av hur, när och i vilken ordning alla operationer ska utföras för att bygga klart produkten. Sedan kommunicerar SP-core med ett agnostiskt lager som består av olika s.k. *abilities*. En ability beskriver en komplett funktion som kan utföras och pratar med ett resurslager som består av olika *virtuella enheter* som ska representera fysiska enheter som till exempel PLC:n. I varje virtuell enhet finns en *driver* som implementerar de olika protokollen för att kommunicera med enheterna. I PLC-exemplet så finns det ett särskilt protokoll för att prata med den och det är det protokollet som PLC-drivern implementerar. I varje lager abstraheras de flesta komplexiteter i koden bort vilket gör att man kan skriva generiska moduler som inte behöver veta exakt vad



Figur 3: Illustration över robotarnas byggplats. Fälten märkta 1-4 visar robot **R4** och **R5**:s arbetsplatser där inkommande paletter med klossar kan placeras. På byggplattan monteras det färdiga klosstornet

de kommunicerar med och därför inte behöver oroa sig om varje lagrets detaljer. I PLC-fallet så kan lagren ovanför prata med PLC-drivern vilken kommer att ställa in exakt de variabler som behöver ändras för att få den effekten som önskas.

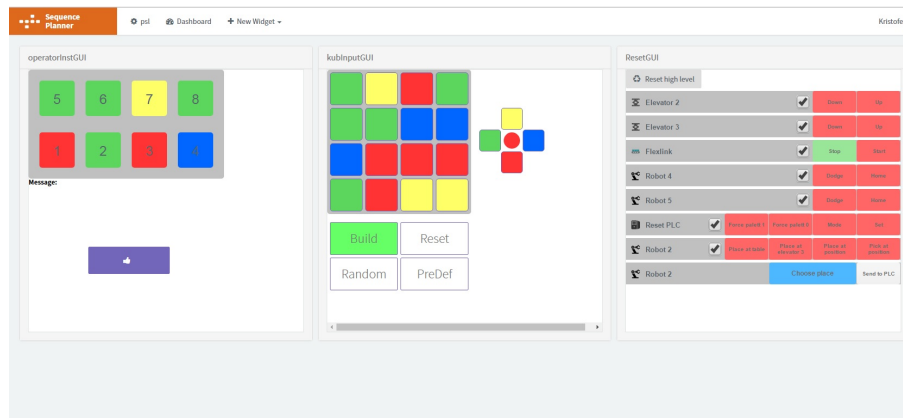
SP-gui, som syns i figur 4, är användargränssnittet för SP-core, en webbapplikation som kommunicerar med SP-core. SP-gui visualiserar datan den får in från SP-core och låter även operatören styra vissa delar av produktionssystemet. Exempelvis när man ska bygga så specificerar man en produkt i användargränssnittet, varpå SP-gui skickar den informationen till SP-core. Sedan svarar SP-core med information om hur klossarna ska placeras på paletten och SP-gui visar upp det visuellt för operatören.

3.4 Operatören

Operatörens roll är att beställa torn i GUI:t för att sedan placera klossar på palleter som förflyttas på transportbandet. När klosstornet sedan är monterat och levererat av robotarna ska sedan operatören plocka ut det färdiga klosstornet vid robotens avlämningsplats.

3.5 Bussen

SP och PLC:n kommunicerar med varandra via en *meddelandebuss* där alla som är kopplade kan skicka och ta del av efter meddelanden. Det används inom det nuvarande systemet två bussar: *Akka* och *Active MQ*. Dessa bussar används inte i detta projekt, en annan buss valdes istället vilken introduceras senare i arbetet.



Figur 4: Användargränssnittet i SP, så kallad SP-gui, erbjuder ett gränssnitt med funktioner som operatören kan använda för att designa och beställa klosstorn (vänster och mitten) samt för att manuellt styra vissa delar av produktionssystemet (höger).

4 Utveckling av identifieringssystemet

Ett av de grundläggande problemen när detaljer i ett produktionssystem ska spåras är att samla data. För detta ändamål finns en mängd olika tillvägagångssätt och tekniker som redan används i dagens olika industrier. En viktig del av arbetet är att sovra bland dessa, finna kandidater och sedan implementera dem i PSL.

Flera olika alternativ för att spåra detaljerna undersöktes, bland annat RFID, QR-koder och streckkoder. Andra möjliga alternativ är även NFC och bildigenkänning. De lösningar som undersöktes vidare och även testades i cellen är listade nedan i 4.2. I slutändan användes enkortsdatorer utrustade med kameror för att identifiera QR-koder.

I detta arbete kommer inget spårningssystem för att hitta detaljer på godtycklig plats i lokalen presenteras. Inte heller kommer detaljer som ej är markerade med korrekt märkning att kunna spåras. Detaljerna förväntas ha unika ID-nummer som kan läsas av från deras märkning.

4.1 Problem

Problemet med spårningen i industriella tillämpningar kan sammanfattas i två huvudpunkter: identifiering och lokalisering.

Identifiering av individuella detaljer och produkter kräver att man kan urskilja alla delar i en process även om de till utseendet är lika. Detta gör det svårt att använda metoder som försöker se skillnad på detaljer. Man måste på något sätt tilldela alla sådana detaljer ett unikt ID.

Lokalisering av detaljer är ytterligare ett problem. Ett av de större problemen är att man ofta har processen som man vill övervaka inomhus. Inomhusmiljön gör att enkla spårningslösningar så som GPS (Global Positioning System) eller andra GNSS (Global Navigation Satellite System), globala positioneringssystem, tappar mycket av sin positioneringsmöjlighet. Dessutom är precisionen man kan få ut av GNSS system inte bättre än 5 cm under ideala förutsättningar, antaget att man inte använder de krypterade signaler, [31] vilket inte alltid uppfyller de krav som ställs på ett positioneringssystem i en industriell miljö.



Figur 5: Bilder på hur de olika indentifieringsteknikernas koder ser ut.

4.2 Utvärdering av identifieringstekniker

För att identifiera unika detaljer så krävs det att de märks med någon form av avläsningsbar ID-kod. Nedan följer en utvärdering av de vanligaste teknikerna för att identifiera detaljer.

4.2.1 RFID

RFID är en teknik som undersökts då den är mycket populär i industrin idag.[32] RFID står för *Radio-frequency Identification* och använder sig av elektromagnetiska fält för att spåra objekt. En RFID-tag, som kan ses i figur 5a, implementeras på eller innanför detaljen som behöver identifieras och till skillnad från streckkoder, som utvärderas i listpunkten nedanför, kräver man inte att RFID-taggen är synlig för att den ska kunna läsas av och eftersom RFID kommunicerar med elektromagnetiska fält så räcker det med att avläsaren är nära nog för att känna av fältet. RFID-taggar delas in i två typer, aktiva och passiva, den aktiva varianten kräver att en strömkälla kopplas till taggen men ger möjlighet till ett mycket längre avläsningsavstånd jämfört mot den passiva varianten.[33] Den passiva drivs av RFID-läsaren vars elektromagnetiska fält fångas av taggens antenn som i sin tur ger taggen ström. Taggen är även inkapslad så tålighet med smuts och väta är hög.

En annan fördel med RFID-tekniken är möjligheten att skriva data, vanligtvis mellan 200 och 8000 bitar, och även att skriva över gammal data på taggarna.

4.2.2 Streckkoder

En annan teknik som testats är streckkoder, som visas i figur 5c. Streckkoder är koder för optisk avläsning och representerar data i form av streck och siffror. En rent teoretisk utvärdering gjordes för eventuell implementation av streckkodsläsare. Fysiska tester i systemet gjordes med Raspberry Pi[34] med tillhörande Raspberry Pi NoIR V2-kamera[35]. En dedikerad streckkodsläsare hade presterat betydligt bättre då de läser av snabbare än kameran men har mycket svårare att läsa av flera koder samtidigt, särskilt om det kommer åtta detaljer på en gång, vilket är fallet i PSL, kan det bli svårt att läsa av alla samtidigt. Industriella streckkodsläsare kostar även en hel del mer än vad budgeten tillät och det hade behövts flera stycken för att kunna följa detaljerna i produktionssystemet.

4.2.3 QR

QR-koder, som visas i figur 5b togs fram under 1994 i Japan av företaget Denso Wave där det huvudsakliga användningsområdet skulle vara att spåra bilar under tillverkningsprocessen. En QR-kod kan man förenklat säga är som en tvådimensionell streckkod, och har en högre lagringskapacitet än streckkoder. Den kan lagra olika sorters data såsom strängar, binärdata, URL, med flera, och behöver inte heller speciella läsare utan kan läsas med en vanlig kamera tillsammans med anpassad mjukvara. QR-koder kan också läsas även om de är skadade,[36] vilket innebär att även om vissa delar av koden ej är läsbara så kan informationen ändå utvinnas.

4.2.4 Val av teknik

Nackdelen med att använda RFID är att det är en dyr lösning, speciellt eftersom varje detalj måste ha en tagg och läsarna är dyra.[37] Budgeten för det här projektet var för liten för att investera i RFID och testa det ordentligt. Det hade även varit svårt att montera dessa på ett bra sätt. Därför avfärdades RFID och istället undersöktes andra lösningar.

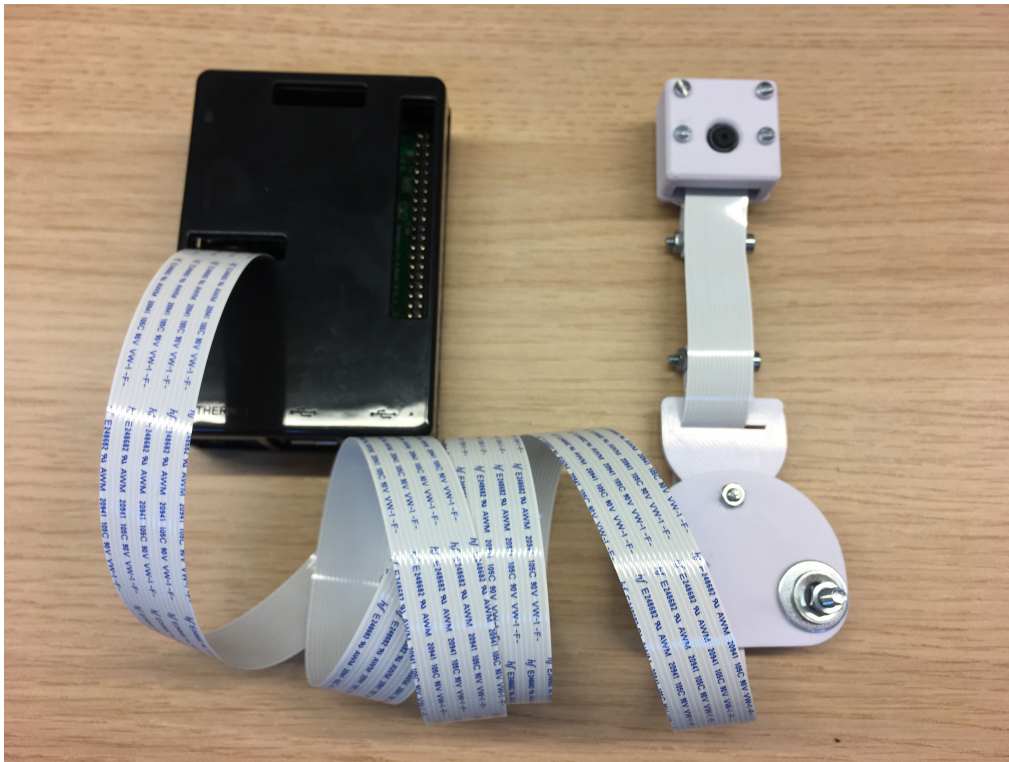
Det fanns många skäl att QR-koder valdes framför övriga alternativ, men ett av de viktigaste var kostnaden, att jobba med QR-koder är mycket billigare än att jobba med RFID. Ett annat skäl är bekvämligheterna med QR-koder, som till exempel möjligheten att få ut koordinater för varje QR-kod så man vet exakt på vilken plats den befinner sig på vid tillfället av avläsning, något som de andra metoderna inte klarar. En annan fördel är möjligheten att läsa av flera detaljer samtidigt med enbart en läsare. Om den möjligheten saknas kan problem uppstå om detaljer transporteras i grupp, exempelvis på pall eller liknande.

Streckkoderna utvärderades för att se om möjligheten fanns att inte behöva stoppa bandet som med QR-koderna, men det fungerade inte.

I slutändan ansågs QR vara mest lämpligt.

4.3 QR-avkodning

Varje spårningsenhet består av en Raspberry Pi enkortsdator tillsammans med en Raspberry Pi NoIR V2-kamera. Kameran är specialgjord för just denna enkortsdator och kopplas direkt in i kameraporten med en en meter lång *FFC-kabel* (Flexible Flat Cable). Standardoperativsystemet *Raspbian* körs och hålls så minimalt som möjligt. Både enkortsdatorerna och kamerorna sätts i skyddande lådor för att minska risken för skador. Vissa delar har tillverkats specifikt för denna setup till exempel kamerahållare för att kunna placera och vinkla kamerorna. Delarna till kameralådan har hämtats ifrån *Thingiverse*[38], en hemsida där användare kan lägga upp sina CAD-modeller som sedan 3D-printades. I figur 6 nedan syns både kameramodellen samt en Raspberry Pi som är sammankopplade genom en FFC-kabel. I projektet användes fyra uppsättningar med en enkortsdator och en kamera för att kunna följa detaljerna genom hela tillverkningsprocessen.



Figur 6: En bild på kameramodellen där själva kamerahållaren syns till höger som är sammankopplad genom FFC-kabeln med en Raspberry Pi till vänster.

Till en början skickades rådata från kameran i form av en videoström från enkortsdatorn till en central dator som behandlade datan och sedan gjorde QR-avkodningen. Problemet som uppstod med denna lösning var att det ofta förekom artefakter i videostrommen, d.v.s. bilden blev skadad, vilket gjorde det svårt att utföra avkodningen. Istället utfördes avkodningen på själva enkortsdatorn vilket tog bort problemet med att skicka kameradatan, men införde ett nytt problem med att enkortsdatorn inte var tillräckligt kraftfull för att hinna avkoda allt som åkte förbi. Detta fick som konsekvens att spårningsdatan oftast blev fel och i princip oanvändbar.

Biblioteket som användes för att känna igen och avkoda QR-koderna heter ZBar och är ett C-baserat bibliotek för igenkänning av diverse streckkoder.[39] Vissa förbättringar gjordes i själva biblioteket. Bland annat byttes `malloc`-funktionen, en funktion som ofta används för minnesallokering i C, ut mot en *slice allocator* för att öka minnesprestandan. En slice allocator erbjuder en utrymmesmässigt effektivare sätt att allokera lika stora minnesblock. Eftersom enkortsdatorerna har begränsat minne så hjälper det prestandan genom att få ner minnesanvändningen av programmet.

Kameraupplösningen optimerades så att den var tillräckligt liten för att avkodningen skulle vara snabb samtidigt som den var stor nog för att kunna skanna av alla QR-koder med bra precision. Till slut flertråddades igenkänningsprocessen så att varje bild behandlades på en separat tråd. Enkortsdatorerna har fyra kärnor, som var för sig kan hantera max en tråd samtidigt, därför skedde en märkbar prestandaökning när programmet flertråddades. Den kunde då behandla fyra bilder samtidigt istället för en.

Datan från avkodningarna, och även ytterligare framtida data som spårningssystemet kommer att generera, kommer i det här projektet att skickas till en central meddelandebuss i syfte att möjliggöra flera olika tjänster att komma åt datan. Den meddelandebuss som används i detta projekt är Apache Kafka[40]. I Kafka skickas alla meddelanden till bussen

under så kallade *topics*, vilka är kategorier som tillåter de som lyssnar på bussen att endast lyssna på det de är intresserade av. Topics kan liknas vid ämnen på en nyhetssajt, så som till exempel politik, sport, ekonomi.

Fördelar med Kafka är att den har bra prestanda vid stora mängder data, att den håller koll på vilken ordning meddelanden kommit in och att den kan köras distribuerat, dvs på flera datorer samtidigt, vilket passar in bra med industri 4.0.

Efter att avkodningen är färdig skickas en kort JSON-sträng med kamera-ID, tid och informationen från QR-koden till meddelandebussen som introducerades ovan. JSON står för *JavaScript Object Notation* och är ett sätt att strukturera data i strängformat vilket gör datan lätt att behandla och skicka runt. Ett exempel på ett sådant meddelande är

```
{ "camera" : "camera_1", "time" : 1494407770, "payload" : 1,
  "corners" : [{ "x" : 0, "y" : 100}, { "x" : 0 "y" : 0},
  { "x" : 100, "y" : 0}, { "x" : 100, "y" : 100}] }.
```

Ett exempel på en tjänst som prenumererar det topic där avkodningsdatan finns är en *webbsocket-tjänst* som utvecklats för att läsa av meddelandena och skicka datan vidare genom en web socket till diverse webbapplikationer som behöver den. En web socket är en ny teknik som har introducerats med HTML5, och precis som en vanlig socket så är en web socket en nätverkskoppling där man kan skicka data fram och tillbaka. Innan det introducerades behövdes inflexibla metoder som vanliga HTTP-requests för att kommunicera med en web server. Den tjänsten är också skriven i C och använder sig av biblioteket *libsoup* för HTTP-servern och web socket-hanteringen[41].

Kamerorna hade svårt för att identifiera QR-koder och streckkoder då de rörde sig på transportbandet. Detta löstes genom att identifieringen utfördes då de stod stilla. Det var även problematiskt att läsa av flera streckkoder samtidigt då de blandades ihop när datorn skulle läsa av bilden. Dock fungerade det betydligt bättre när man använde sig av QR-koder och flera lyckade försök gjordes med upp till åtta QR-koder samtidigt.

Själva tidsstämplingen av data kan vara ett stort problem när det kommer till distribuerade system, där flera olika och osynkroniserade delsystem inhämtar data samtidigt. Fel i synkronisering mellan systemen kan leda till felaktig data som i sin tur kan leda till fel om denna används i kritiska processer.[42]

Om projektet hade haft en större budget hade det varit möjlighet att undersöka bättre utrustning så som streckkodsläsare och även speciella QR-kodsläsare. Dessa hade troligtvis presterat betydligt bättre än vad Raspberry Pi-kamerorna gjorde, då de läser av betydligt snabbare. Kanske hade det även varit möjligt att läsa av koderna när de rörde på sig.

4.4 Kommunikation med PLC:n

Utöver data från kamerorna kommer data även att hämtas direkt från produktionssystemet. För att få en översyn över vad som händer fysiskt i tillverkningen så krävs det att signaler till och från de maskiner och anordningar som arbetar i processen samlas in. Genom att spåra de förändringar som sker i PLC:n så kan man få information om detta och därefter lista ut detaljernas tillstånd. Det krävs dock att de moment som sker i processen är väldefinierade och separerade så att inga tvivel om vad som egentligen sker kan uppstå. Ett exempel kan vara en svarvoperation som utförs i processen, där måste man bestämma om in- och utmatning ska ingå i själva svarvoperationen eller om dessa är separata. Ett annat exempel kan vara en transportsträcka, där måste det bestämmas var denna startar och var denna slutar. Operationerna och resurserna beskrivs i PLC:n med hjälp av *states* som innehåller tillståndsvariabler, dessa representeras i projektet av heltalen 0 (*not ready*), 1 (*ready*), 2 (*executing*) och 3 (*finished*).

Själva mjukvaran för att läsa av PLC-variabler är skriven i C och använder sig av ett open-source bibliotek som heter Snap7[43]. Med detta bibliotek får man fullständig tillgång till PLC:n och man kan läsa av, lägga till och ändra variabler och därmed styra systemet. Programmet lyssnar på meddelanden från meddelandebussen som har olika instruktioner för vad den ska göra. Exempelvis kan det vara en instruktion om att ändra en variabls värde eller att börja läsa av en variabls tillstånd och skicka det tillbaka genom bussen. I programkoden finns ett variabel kallad *tick* som definierar hur ofta programmet läser av PLC:ns tillstånd. Anledningen till att programmet måste läsa av alla PLC-variabler med jämna mellanrum är för att PLC:n inte skickar ut någon signal när en variabel förändras. Man måste konstant läsa av variablerna och själv bestämma om en förändring har skett. Det är viktigt att tick-värdet ställs in rätt, eftersom ett för högt värde kan göra att händelser missas, samtidigt som ett för lågt värde kan komma att kräva för mycket prestanda.

5 Databehandling

Från identifieringssystemet och produktionssystemet kommer stora mängder data till meddelandebussen. För att få fram användbar information av datan krävs att den samlas, organiseras och bearbetas. För det ändamålet skapas en databas med ett par tillhörande tjänster som gör just detta.

Tanken är att datan ska kunna användas för att ge en överskådlig bild av hur produktionssystemet har agerat genom en historik på vad som hänt. Historiken skulle kunna användas vid felsökning. Om systemet kör fast skulle man kunna få upp en logg på allt som hänt tills den tidpunkten och kunna leta bakåt i tiden efter misstänksamma beteenden. Även så kallade *nyckeltal*, även kallade *KPI*:er från engelskans *Key Performance Indicator*, tas fram. Nyckeltal används idag inte bara inom tillverkningsindustrin utan även inom många andra affärsområden och offentliga verksamheter och skulle kunna hjälpa operatören upptäcka ineffektiviteter i produktionssystemet.[44]

För att ta fram historik och nyckeltal krävs att datan lagras på något sätt, att det skrivs lämpliga algoritmer som behandlar datan och tjänster som tillgängliggör den behandlade datan.

5.1 Val av lagringsteknik

Valet av lagring stod mellan SQL och NoSQL. SQL står för *Structured Query Language*[45] och är en standard för en typ av programmeringsspråk som används för att bygga och hantera databaser som lagrar data i tabellform. SQL bygger på att det finns en struktur i datan i form av relationer. En av fördelarna med strukturen är att man kan göra komplicerade, sammansatta sökningar i databasen som då kan skicka tillbaka datan på precis den form som behövs.

NoSQL står för *Not Only SQL*[46] och har inte samma krav på struktur i datan. Idén för NoSQL uppkom redan på 60-talet,[46] men det har fått ett uppsving på senare år på grund av den ökande mängden data som måste av hanteras av tjänster som Facebook, Google och Amazon. De främsta användningsområdena för NoSQL är när stora, ostrukturerade datamängder hanteras.[47]

SQL valdes i det här projektet över NoSQL eftersom datan som kommer från identifieringssystemet och produktionssystemet har etiketter och därmed är strukturerad. Datan kan därför naturligt sättas in i tabeller och sökas efter på mängder av olika sätt, vilket är hjälpsamt eftersom det ger maximal flexibilitet när algoritmerna som gör rådatan användbar ska skrivas. NoSQL har heller inget naturligt stöd för s.k. JOIN-operationer, som sätter ihop tabeller med relaterad data, vilket är av intresse när data i form av adresser från PLC:n ska översättas till någonting mer vänligt för människor att ta till sig.

En fördel med NoSQL är att det minskade kravet på struktur gör det enklare att fördela upp databasen över flera olika datorer. Det innebär att NoSQL har vissa övertag när mängden data som hanteras skalas upp, men det sågs inte som ett tillräckligt argument för att använda NoSQL i det här fallet.[48]

5.2 Insättning i databasen

En av databastjänsterna läser in data från meddelandebussen och lagrar den nästan exakt som den är i databasen. Undantaget är att tjänsten har en enkel algoritm för att associera all data med en order. Produktionssystemet håller inte ordning på ordrar, utan det har implementerats här för att kunna ge operatören en mer sorterad överblick av vad som skett, och även för att kunna associera nyckeltal till specifika ordrar.

Datan som kommer in från kamerorna är uppdelad i ett meddelande per avläst kod, och all den datan aggregeras till större avskanningshändelser för att möjliggöra fler typer av sökningar bland datan. Datan som kommer från PLC:n är buntar av meddelanden som kommer med jämna mellanrum och ger en bild av hur alla PLC:ns variabler är vid en viss tidpunkt. Både kameradatan och PLC-datan stämplas med en order med den enkla algoritmen som nämndes ovan. Förutom det sätts alltså all data in precis som den ser ut när den går genom meddelandebussen och behandlingen av datan görs istället endast när det sker en förfrågan, eftersom det skulle vara slöseri med resurser att behandla data som aldrig används.

Databastjänsten är skriven i Java med motiveringen att det erbjuder ett bra gränssnitt till PostgreSQL via Java Database Connectivity[49], förkortat JDBC. Java gör det även enkelt att koppla till meddelandebussen som användes och läsa meddelandena.

Sedan finns även en HTTP-tjänst som går att kontakta via en förfrågan för att få ut relevant data från databasen. Exempelvis går det att få cykeltider, orderhistorik, vilka detaljer som finns i systemet, med mera. Genom dessa förfrågningar kan användargränssnittet visa upp datan för slutanvändaren.

5.3 Förfrågningar och heuristiker

Ytterligare en Javatjänst sköter kommunikationen mellan användargränssnittet och databasen. Den tar emot förfrågningar och svarar via HTTP-protokollet. HTTP valdes här istället för websockets som användes tidigare eftersom förfrågningarna som skickas till databasen rör saker som inte är lika tidskritiska som innan. Förfrågningar skickas inte kontinuerligt utan endast när operatören klickar på vissa saker i användargränssnittet. En annan anledning för HTTP är att Javas standardbibliotek gör det enkelt att implementera en HTTP-server.

De inkommande HTTP-förfrågningarna som förfrågningstjänsten hanterar innehåller nyckel-värden-par. Nyckeln *requestType* antas alltid finnas i JSON-strängen och dess motsvarande värde talar om vilken typ av information som ska skickas tillbaka. Ytterligare nyckel-värde-par kan förekomma beroende på vilken typ av förfrågan som skickats, och de ger då mer information om hur förfrågningen till databasen ska utformas.

Vid förfrågningar så behandlas datan och överförs till två olika former. Dels historik på allt som hänt, för antingen en order eller en detalj, och dels till nyckeltal som till exempel cykeltid, vilket tiden det tar för en order att bli klar från beställning till leverans av slutprodukt.

Ett exempel på en heuristik är den som används för att ta fram historiken för en detalj. Här blir implementeringen specifik för produktionssystemet i PSL. Det som i princip sker är att algoritmen tar fram den sista tiden som detaljen i fråga sågs av den första kameran. Den tiden jämförs sedan med första tiden som en viss hiss aktiverades i produktionssystemet, vilket avgör om detaljen låg på den första eller andra paletten som skickades in i systemet. Med den vetskapen kan de händelser som behandlat antingen den första eller andra paletten plockas ut och läggas till historiken. Utöver tiden som detaljen sågs i kameran plockas även dess koordinater ut, och används för att uppskatta var på paletten detaljen befinner sig. Med den informationen kan ytterligare några händelser som behandlat detaljer på specifika platser på paletten hämtas från databasen och läggas till i historiken. Historiken sorteras sedan efter tid och skickas tillbaka till användargränssnittet som en JSON-sträng via HTTP.

6 Datavisualisering

För att kunna förmedla den information som inhämtats är det viktigt att den presenteras på ett förståeligt sätt. Operatören ska snabbt och enkelt kunna få ut den information som är viktig för denne.

Visualiseringen kommer att ske på huvudsakligen två olika sätt. För det första kommer det finnas realtidsvisualisering i form av ett Gantt-schema och en karta som med hjälp av data direkt från spårningssystemet visar detaljernas och produktionssystemets rörelser i realtid. För det andra så kommer det finnas möjlighet att genom databastjänsten få ut en användarvänlig logg på vad som hänt hela produktionssystemet och även individuella detaljer i systemet, samt nyckeltal. Tanken är att realtidsspårningen ska användas vid drift och loggen vid felsökning.

Att på ett effektivt sätt presentera data är en vetenskap i sig, och är ett problem med en uppsjö lösningar som i många fall har låg prioritet. Grafik kan i många fall vara det bästa sättet att illustrera den insamlade och analyserade datan, där alternativet skulle vara att presentera tabeller med värden eller liknande.[50] Presentationen av data spelar även en viktig roll för hur denna data uppfattas av en användare. Samma data kan både tolkas och upplevas olika beroende på hur den presenteras.[51]

För att presentera och visualisera den data som samlas in har fyra metoder valts: ett Gantt-schema, en karta, tabeller över nyckeltal och historik. Dessa beskrivs i detalj nedan.

6.1 Gantt-schema

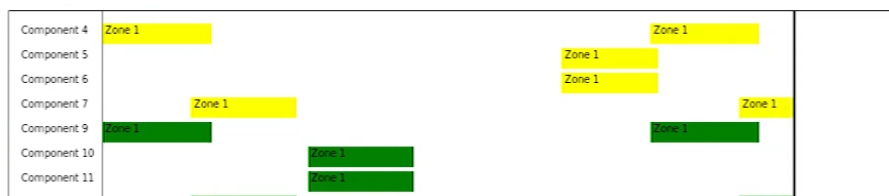
Ett Gantt-schema är ett liggande stolpdiagram som används vid projektplanering. Varje stolpe representerar ett moment och stolpen visar när momentet är tänkt att påbörjas respektive avslutas. Hela schemat ger på så sätt en överblick över hur projektet väntas fortskrida.

Användningen av Gantt-schemat i det här spårningssystemet kommer att avvika en del från detta. Varje rad kommer att representera en detalj i produktionssystemet och stolparna på raden (det kan finnas flera) kommer att representera zoner eller operationer i systemet. Dessutom kommer Gantt-schemat inte vara en planering, utan kommer ge en historik över vad som hänt samtidigt som det uppdateras i realtid. Figur 7 visar hur schemat såg ut i ett tidigt prototypstadium. I exemplet åkte detaljer runt i par på paletter och passerade då och då ett par kameror som tillsammans utgjorde en zon.

Sist finns webbapplikationen som tidigare nämnts kommunicerar genom en websocket-tjänst med Kafka. I webbapplikationen finns flera så kallade *widgets* som visualiserar datan. Det finns ett dynamisk Gantt-schema som visar i realtid vart detaljerna befinner sig. Den läser av när en detalj passerar en kamera och markerar att detaljen antingen har gått in i en eller lämnat en zon. En zon är när detaljen befinner sig mellan två kameror och registreras av den första kameran och därmed kommer in i zonen. Sedan registreras den av den andra kameran och då har den lämnat zonen men går även in i en ny zon. När detaljen befinner sig i en zon kommer Gantt-schemat att animera de olika raderna så att man får visuell återkoppling på hur lång tid det tar.

Webbapplikationen läser av när en detalj passerar en kamera och markerar att detaljen antingen har gått in i eller lämnat en zon. En zon är när detaljen befinner sig mellan två kameror och registreras av den första kameran och därmed kommer in i zonen. Sedan registreras den av den andra kameran och då har den lämnat zonen men går även in i en ny zon. När detaljen befinner sig i en zon kommer Gantt-schemat att animera de olika raderna så att man får visuell återkoppling på hur lång tid det tar.

Själva implementeringen är skriven i ren Javascript utan några bibliotek. Anledningen till det är att det var enklare att göra den funktionaliteten som behövdes från början istället för att försöka modifiera och omvandla ett befintligt bibliotek eller kodbas. HTML5-canvas användes för att göra själva ritningen av Gantt-schemat och genom websockets kunde den kommunicera med websocket-tjänsten som läser av relevanta meddelande från bussen.

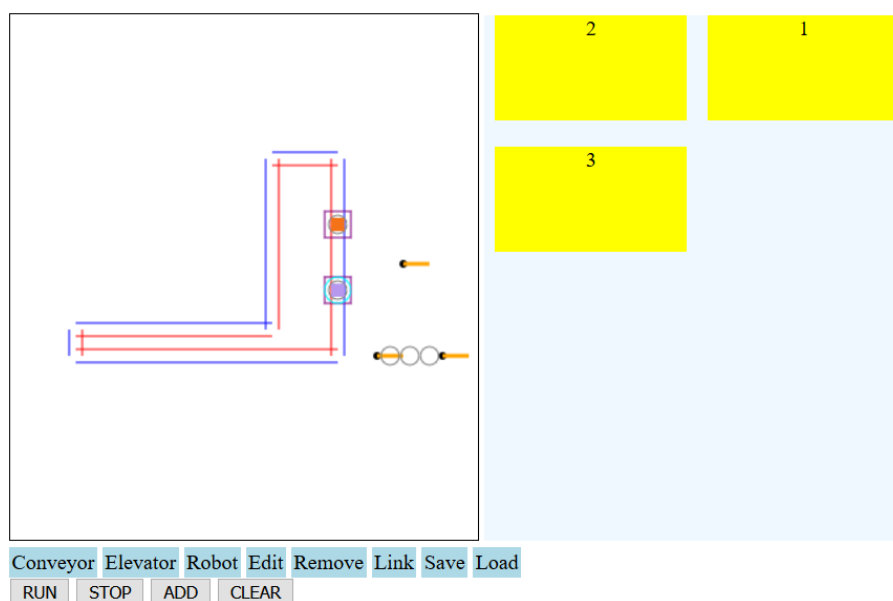


Figur 7: Ett Gantt-schema som i realtid visar vilka detaljer som befinner eller har befunnit sig i en viss zon.

Genom Gantt-schemat kan man få en tydlig bild över vilka processer som har påverkat en detalj under dess tid i produktionssystemet. Med den informationen skulle man kunna identifiera flaskhalsar och andra otrevligheter i systemet.

6.2 Karta

För att kunna följa detaljers väg genom produktionssystemet och för att även få en intuitiv bild av hur produktionssystemet fungerar samt var detaljer befinner sig implementerades en enkel karta. En bild av kartan så som den ser ut under körning visas i figur 8.



Figur 8: En interaktiv karta över produktionssystemet.

I figur 8 återges transportband som röda och blå streck där rött representerar bandets högra sida i färdriktningen och blått den vänstra. De två större fyrkanterna representerar områden som kameror eller sensorer har uppsikt över i produktionssystemet. De mindre, helt fyllda lila och brandgula fyrkanterna representerar grupper av detaljer. De orangea och svarta delarna i kartan representerar robotar i produktionssystemet. De grå cirkelarna är olika former av stopp eller avställningsplatser. Till höger om kartan kan en större bild

ses av de detaljer som grupperats tillsammans, den fås om man i kartan trycker på en grupp.

För att förenkla skapandet av nya kartor över produktionssystem implementerades en enkel grafisk lösning där användaren kan klicka och placera nya systemdelar, så som transportband, stopp och avställningsplatser. Under kartan finns knappar som används för att göra detta. Här finns även knappar för att starta kartapplikationen.

Kartan kan köras både i ett *statiskt* och ett *aktivt* läge. I det statiska läget uppdateras detaljernas positioner endast utifrån den data som kommer från identifieringssystemet. Positionerna förblir oförändrade tills de ses på nytt någon annanstans. Man får alltså en ”senast sedd här”-bild av produktionssystemet. Delar utan kameratäckning kan då inte presenteras för en operatör.

I det aktiva läget körs en enkel modell av produktionssystemet som antar att detaljerna flyttas efter givna mönster. Denna modell uppdateras sedan av den data som ges av spårningen men i områden utan kameratäckning uppdateras detaljer utifrån den interna modellen av produktionssystemet. Detaljer och grupper av detaljer kan röra sig längs transportband och kan även flyttas av robotar i kartan.

Detta aktiva läge är inte fullt testat och behöver ställas in för det systemet som ska modelleras. Flera aspekter av modellen måste ställas in bland annat bandens och robotarnas hastigheter.

Risken med det här läget är att man kan få en falsk bild av vad som pågår i produktionssystemet, och att man på grund av det fattar felaktiga beslut baserat på en felaktig modell.

Kartan är skriven i samma stil som Gantt-schemat med standard Javascript, HTML5 canvas och websockets.

6.3 Historik och nyckeltal

Tjänsterna som sköter databasen tillgängliggör information i främst två olika former. Dels går det att få ut historiken på vad som hänt hittills i produktionssystemet för antingen en order eller en detalj, och dels går det att få diverse nyckeltal.

Historiken för en viss detalj är en lista av allt som hänt detaljen sorterad efter tid. I användargränssnittet finns möjlighet att specificera för vilken detalj och orderhistoriken ska hämtas. Listan fås från databastjänsten via en HTTP-förfrågan och visualiseras genom att den helt enkelt skrivs ut i användargränssnittet. Även nyckeltal, som till exempel cykeltid och produktivitet, skrivs helt enkelt ut efter begäran.

För att visualisera nyckeltalen används tabeller i användargränssnittet. I figur 9 kan tabellerna ses så som de visas för operatören. I den övre vänstra tabellen kan man se de ordrar som finns i systemet, i den övre högra tabellen ses de detaljer som finns i den valda ordern. I de nedre tabellerna ses historiken för vald order och vald detalj.

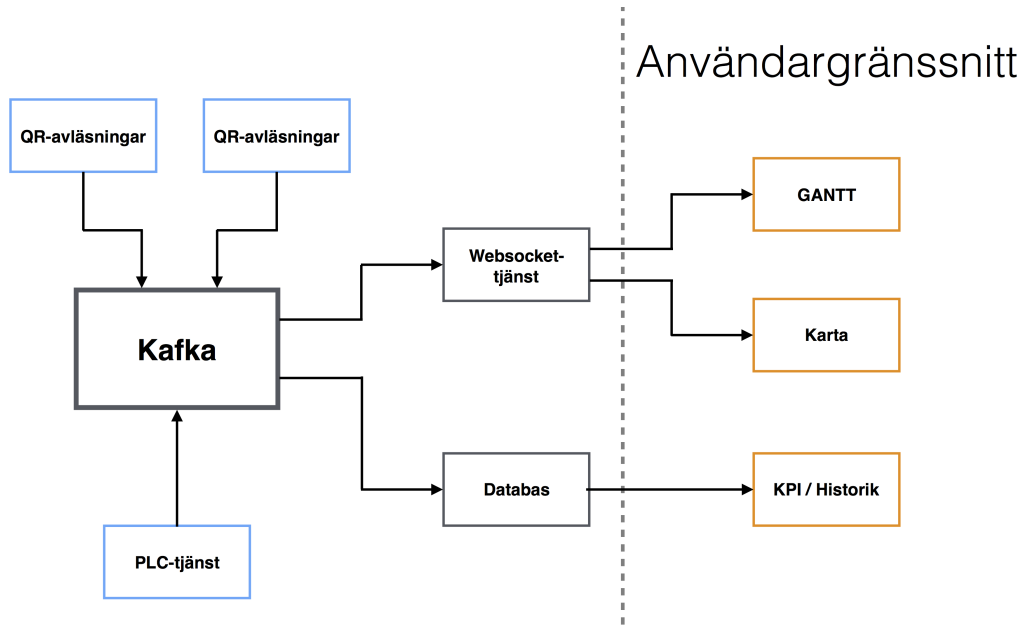
I de två övre tabellerna kan man klicka på ordrar eller detaljer för att få upp deras historik.

All info		All orders	
Orders		Components	
ID		ID	
0		1	
1		2	
Order History for order 0		Component History for component 1	
Time	Resource	Action	
2001-2-27 16:55:2.674	conveyor	transport	
1973-11-30 4:43:20.944	robot1	assembly	
1973-7-24 8:21:20.834	robot2	transport	
Time	Resource	Action	
1998-12-5 3:31:16.847	conveyor	transport	
1999-7-29 4:17:27.314	robot1	assembly	
1999-1-12 7:46:25.235	robot2	transport	

Figur 9: *Tabeller som visar data för både ordrar och detaljer.*

7 Resultat

Resultaten av alla delar av projektet var för sig har redan presenterats, men här följer en sammanställning där systemet betraktas som en helhet.



Figur 10: Schematisk bild över de ingående delarna i spårningssystemet. De två rutorna med "QR-avläsningar" symboliserar de fyra kamerorna som används i projektet.

I figur 10 syns en schematisk bild på hur hela spårningssystemet är utformat. En enkel instans av en Kafka server körs på ett nätverk som är åtkomligt för både kamerorna och PLC-tjänsten och fungerar som vår meddelandebuss. Kamerorna, som i det här projektet är fyra stycken skickar data om allt de ser till meddelandebussen. Likaså PLC-tjänsten läser periodvis av alla tillstånd i PLC:n som styr hela produktionssystemet, och skickar den datan till meddelandebussen.

Datan i meddelandebussen konsumeras på två olika sätt. Den konsumeras direkt via websockets av vissa delar i användargränssnittet där datan används för att ge operatören en realtidsuppfattning om vad som händer i produktionssystemet. Den konsumeras även med jämna mellanrum av databastjänsten som helt enkelt lagrar all inkommande data i databasen. Sedan kan de mer loggbaserade delarna av användargränssnittet begära information från databasen, som till exempel en historik på allt som skett.

Användargränssnittet gör sitt jobb på så sätt att den visar upp all data som hade ansetts relevant och gjorts tillgänglig i databastjänsten.

7.1 Testning

Genom att använda produktionssystemet som tillhandahålls i PSL på Chalmers Tekniska Högskola kan det utvecklade spårningssystemet testas. Kameraenheterna kan strategiskt placeras på de ställen som anses kritiska och risken för påverkan på produkterna är stor, till exempel vid operatörsstationer, innan eller efter montering eller omlastningsstationer. I PSL erbjuder produktionssystemet alla dessa moment och anses därför lämpligt för testning.

7.1.1 Identifieringssystemet

Tester utfördes för att se hur väl kamerorna kunde avkoda QR-koderna på detaljerna när de passerade kamerauppsättningen under tillverkningsprocessen. Det som mättes var hur många detaljer som kändes igen, vilket sedan dividerades på det totala antalet som hade åkt förbi, vilket gav en viss träffsäkerhet. En kamera sattes upp ovanför en del av transportbandet där det fanns ett mekaniskt stopp. Fulla paletter, innehållande åtta detaljer vardera, skickades längs bandet till en kamera, stannade upp en vald tid och fortsatte sedan längs bandet. Först sattes tiden till 0 ms, vilket fortfarande innebar ett litet stopp eftersom stoppet behövde öppnas när paletten anlände, och då erhöles en träffsäkerhet på 86,3 %. När paustiden ökades till 500 ms erhöles en träffsäkerhet på 100 %, vilket ansågs nödvändigt vid den här kameran för att kunna göra en förutsägelse om komponenternas framtida resa genom produktionssystemet.

Även kamerorna vid byggplatserna och den kamera som var fastmonterad på den stora roboten hade en träffsäkerhet på 100 % då detaljerna var stillastående tillräckligt länge för att kamerorna skulle kunna avläsa QR-koderna.

Prestandaökningen som skedde då QR-avkodningen delades på flera trådar hann inte mätas empiriskt, men det skedde en märkbar förbättring i antalet avkodningar per tidsenhet.

7.1.2 Användargränssnittet

Användarvänligheten av användargränssnittet hann inte testas, till exempel genom att låta utomstående personer få pröva på att använda det, och det är därmed oklart hur intuitivt det egentligen är.

8 Diskussion

8.1 Skalbarhet

Vår lösnings genomförbarhet på storskalig nivå är osäker. Naturligtvis med en större budget skulle vissa delar kunna förbättras och bytas ut, beroende på vilken industri lösningen tillämpas på. För en industri så kanske streckkoder med streckkodsläsare fungerar bättre, medan för en annan industri RFID kanske lämpar sig bättre. Dessutom är det tvivelaktigt att en etablerad industri skulle använda sig av Raspberry Pi med tillhörande kameramodul för spårningen, utan snarare speciella industriövervakningskameror[52] med möjlighet att läsa av många gånger fler QR-koder eller streckkoder. I vissa tillverkningsprocesser kan det vara så att detaljerna för att bygga en produkt helt enkelt är för många för att spårning på detaljnivå ska vara en gångbar lösning.

Skalbarheten av vår metod faller på dataaggregationen. Insamlingen bör gå bra eftersom att alla kameror kör parallellt. Meddelandebussen Kafka kan köras distribuerat och även SQL-databaser kan hantera stora mängder data.[53] Aggregationen däremot är något som i detta fall måste skötas manuellt. Någon måste sätta sig in i tillverkningsprocessen och förstå hur denna är uppbyggd för att sedan skriva algoritmer som kan översätta datan till förståelig information för operatören.

8.2 Modularitet

För att kunna installera och använda det framtagna spårningssystemet så måste följande saker finnas tillgängliga.

- Det måste finnas möjlighet att placera QR-koder på de detaljer som ska spåras.
- Runt ställen där detaljer rör sig måste det finnas tillräcklig belysning samt möjlighet att montera kameror så att de får god uppsyn över detaljerna.
- Det måste finnas möjlighet att koppla enkorts datorerna till ett nätverk, fördelaktigen Wi-Fi så att man inte behöver dra kablar till enkorts datorerna.

Ett sätt att utvärdera modulariteten av det framtagna spårningssystemet är att försöka använda det på ett godtyckligt system utanför Chalmers. Tyvärr har det inte funnit möjlighet att få tillgång till ett sådant system och således är systemet endast testat i PSL på Chalmers. Dock på grund av valet att använda Kafka som en meddelandebuss ligger det i systemets natur att vara modulär. Det är smidigt att koppla in specialanpassade tjänster som skrivs i valfritt språk efter användarens behov.

Med det framtagna spårningssystemet används nu två olika meddelandebussar i PSL: en för produktionssystemet och en för spårningssystemet. Tanken från början var att allt skulle köra på en gemensam buss, nämligen Kafka, men det blev för komplicerat. Istället utvecklades ett separat spårningssystem med en Kafkabuss. Produktionssystemet fick behålla sin tidigare meddelandebuss och kör därmed inte på Kafkabussen som används i spårningssystemet. Detta gör att spårningssystemet är helt avskilt från produktionssystemet och därmed mer modulärt och enklare att använda i andra produktionssystem. Dock blir det svårare om spårningssystemet ska ha möjlighet att påverka robotarna när två olika bussar används. I praktiken är det fortfarande möjligt att på något sätt koppla in sig i produktionssystemet för att styra det, men det kan bli krångligare än man hade integrerat spårningssystemet i produktionssystemet.

8.3 Spårningssystemets brukbarhet

Identifieringen av detaljer som gjorts i projektet med Raspberry Pi-lösningen kanske inte är aktuell i industrin, då det redan finns bättre lösningar tillgängliga som bland annat RFID-taggar eller streckodsläsare. Dock kräver dessa att investeraren har ett större kapital än vad som fanns att förfoga över i detta projekt.

Att spåra klossar kan liknas vid att spåra större detaljer i industrin vilket gör att projektet fortfarande är användbart även fast det inte alltid är intressant att spåra på detaljnivå inom industrin. Den del av arbetet som är mest aktuell att använda sig av är mjukvaran som gjorts och sättet att visa upp informationen. Då bland annat Gantt-schemat och kartan kan hjälpa till att effektivisera genom att visa upp datan på ett bättre och smidigare sätt för operatören.

8.4 Estetik

En nackdel är att QR-koderna inte är särskilt estetiskt tilltalande, speciellt inte om de är fastsatta med tejp eller något annat synligt. Koderna måste fästas på alla detaljer som ska spåras, och vill man inte att koderna ska vara kvar i slutprodukten måste de dessutom avlägsnas när produkten är färdig. Dock kan man enkelt använda sig av klistermärken med QR-koder på och sedan placera dessa på ett sätt så att de inte syns på utsidan av slutprodukten.

8.5 Möjliga förbättringar av existerande funktionalitet

8.5.1 Spårningssystemet

För att komma runt problemet med att inte kunna identifiera koder medan de rör sig skulle en bättre kamera med högre bildfrekvens kunna användas. Detta skulle i sin tur antagligen kräva mer prestanda från enkorts datorn när denna ska behandla fler bilder per sekund. Ett sätt att öka prestandan skulle kunna vara att dela upp bildigenkänningen på fler trådar. Processorerna i enkorts datorerna som användes hade fyra kärnor, och i körning låg en av kärnorna på full belastning medan de andra inte användes.

Ett annat alternativ för att öka prestandan i bildbehandlingen skulle kunna vara att använda ett grafikkort då dessa är vida överlägsna när det kommer till många typer av bildbehandling.[54][55] I PSL hade detta varit överflödigt, men om mängden detaljer som passerar kamerorna per tidsenhet ökar markant så kan det här komma att bli nödvändigt.

Alternativt kan enkorts datorerna och kamerorna ersättas med specialgjorda QR-kodsavläsare som fungerar likt en vanlig streckodsläsare. Dessa läsare kan gör igenkänningen och avkodningen tiotals gånger per sekund och har därför bättre prestanda och träffsäkerhet.[56]

8.5.2 Användargränssnittet

Användargränssnittet utvecklades mest för att testa olika sätt att visualisera datan, och ser därför inte särskilt polerad ut. Den skulle kunna paketeras snyggare och göras mer enhetlig rent stilmässigt.

8.6 Möjlig ytterligare funktionalitet

Vidareutveckling av spårningssystemet är möjlig på ett flertal sätt, till exempel genom att låta systemet reagera på informationen som samlas.

Man kan ha hantering för operatörsfel, förslagsvis så att om operatören sätter fel detaljer på bandet gentemot vad SP instruerar, så känner systemet av att detaljerna som är på bandet inte kan bygga produkten som specificerats och skickar således tillbaka detaljerna till operatören som får rätta till sitt misstag.

En annan möjlighet är att systemet inte kräver att man sätter rätt detalj på rätt plats, alltså att inte behöva placera klossarna på rätt plats på paletten, utan att man bara kan osorterat sätta in detaljerna som krävs för att bygga den specificerade produkten. I det fallet kan man också tänka sig att man kan sätta in så många detaljer som helst och systemet sorterar fram rätt detaljer för att bygga den specificerade produkten, och skickar tillbaka överflödet, eller alltihop om detaljerna inte kan bygga produkten.

Vidare skulle datan som samlats in från systemet kunna användas för att köra en simulering av hela systemet, förslagsvis med kartan som implementerats i projektet. Detta skulle dock kräva att en modell av systemet togs fram, vilket skulle kunna vara oerhört tidskrävande beroende på systemet.

Eftersom mängden data som samlas in är så stor så skulle datautvinningsalgoritmer kunna vara ett sätt att utvinna användbara underliggande mönster och företeelser i datan[14]. Detta skulle kunna vara en stor fördel för framtidens smarta produktionslinor, då man skulle få en ökad förståelse för hur maskinerna i produktionssystemen fungerar i olika situationer.[57]

Framtida produktionssystem skulle med fördel kunna använda fler sensorer som samlar in data från olika källor för att med större säkerhet kunna övervaka och även förutse cellens beteende.

Fokus borde även läggas på att undersöka möjligheterna att tillämpa maskininlärningsmetoder för big data för att hitta mönster och intressanta aspekter i produktion från den stora mängd data som har potential att samlas in. En användning av big data som skulle vara till nytta är artificiell intelligens, vanligen förkortat A.I. A.I:n skulle exempelvis med hjälp av datan försöka förutspå om det går långsamt eller till och med blir stopp i systemet. Efter det har skett kan den ta förebyggande åtgärder så att de inte sker igen. Den skulle även kunna användas retroaktivt på datan och försöka lista ut om det finns områden som kan förbättras och optimeras. Tillsammans med en mer överblickande statistisk analys, kan man få bättre kontroll över systemets prestanda.

9 Slutsats

I projektet har det identifierats att det fanns en behov att kunna spåra detaljer som finns i systemet på individuellt nivå samt i vilket tillstånd de befinner sig. Genom uteslutning kom det fram till att QR-koder var den mest anpassade lösningen efter situationen för att spåra individuella detaljerna i produktionssystemet. Koderna har kunnat identifieras med hög träffsäkerhet och informationen om var de befunnit sig har tillsammans med information om alla händelser som skett i produktionssystemet samlats i en databas. Databastjänster har skrivits som på förfrågan kan behandla datan och skicka tillbaka relevant information om tillverkningsprocessen till ett användargränssnitt där den visualiserats på en mängd olika sätt för operatören.

Användningen av en meddelandebuss fungerade väl och gjorde det möjligt att utveckla tjänster som använde datan oberoende av varandra. Detta är en fördel för spårningssystemets modularitet, vilket gör att det med lite anpassning skulle kunna sättas in i befintliga produktionssystem och sedan utvecklas vidare efter systemets framtida behov.

Referenser

- [1] Jay Lee, Hung-An Kao och Shanhu Yang. "Service innovation and smart analytics for industry 4.0 and big data environment". I: *Procedia Cirp* 16 (2014), s. 3–8.
- [2] J. Manyika m. fl. *Disruptive technologies: Advances that will transform life, business, and the global economy*. Tekn. rapport. McKinsey Global Institute, maj 2013.
- [3] Thomas K McCraw. *Creating modern capitalism: how entrepreneurs, companies, and countries triumphed in three industrial revolutions*. Harvard University Press, 1997.
- [4] Pat Hudson. *The industrial revolution*. Bloomsbury Publishing, 2014.
- [5] Deirdre McCloskey. "Review of The Cambridge Economic History of Modern Britain". I: *Times Higher Education Supplement* (jan. 2004). Edited by Roderick Floud and Paul Johnson, <http://deirdremccloskey.org/articles/floud.php>.
- [6] Andrew Atkeson och Patrick J Kehoe. *The transition to a new economy after the second industrial revolution*. Tekn. rapport. National Bureau of Economic Research, 2001.
- [7] Jeremy Greenwood. *The third industrial revolution: technology, productivity, and income inequality*. 435. American Enterprise Institute, 1997.
- [8] Heiner Lasi m. fl. "Industry 4.0". I: *Business & Information Systems Engineering* 6.4 (2014), s. 239.
- [9] Näringsdepartementet. *Smart industri*. <http://www.regeringen.se/regeringens-politik/smartindustri/>. 2016.
- [10] Näringsdepartementet. *Smart industri - en nyindustrialiseringsstrategi för Sverige*. Tekn. rapport. Regeringskansliet, 2016.
- [11] Shiyong Wang m. fl. "Implementing smart factory of industrie 4.0: an outlook". I: *International Journal of Distributed Sensor Networks* (2016).
- [12] Radhakisan Baheti och Helen Gill. "Cyber-physical systems". I: *The impact of control technology* 12 (2011), s. 161–166.
- [13] Paul Daugherty m. fl. "Driving unconventional growth through the Industrial Internet of Things". I: *White Paper, Accenture* (2015).
- [14] Danah Boyd och Kate Crawford. "Six provocations for big data". I: *A decade in internet time: Symposium on the dynamics of the internet and society*. Vol. 21. Oxford Internet Institute Oxford. 2011.
- [15] *Cyber-Physical Systems (CPS)*. Tekn. rapport. National Science Foundation, 2016. URL: <https://www.nsf.gov/pubs/2017/nsf17529/nsf17529.htm> (hämtad 2017-05-12).
- [16] Siddhartha Kumar Khaitan och James D McCalley. "Design techniques and applications of cyberphysical systems: A survey". I: *IEEE Systems Journal* 9.2 (2015), s. 350–365.
- [17] Shiyong Wang m. fl. "Towards smart factory for Industry 4.0: A self-organized multi-agent system with big data based feedback and coordination". I: *Computer Networks* 101 (2016), s. 158–168.
- [18] Näringsdepartementet. *Faktablad: Smart industri - en nyindustrialiseringsstrategi för Sverige*. Tekn. rapport. Regeringskansliet, 2016.

- [19] *Automatiseringen har tagit bort 450 000 jobb på fem år*. Dagens nyheter. 2015. URL: <http://www.dn.se/debatt/automatiseringen-har-tagit-bort-450-000-jobb-pa-fem-ar/> (hämtad 2017-05-03).
- [20] Rajkishore Nayak m. fl. "RFID in textile and clothing manufacturing: technology and challenges". I: *Fashion and Textiles* 2.1 (2015), s. 9. DOI: 10.1186/s40691-015-0034-9. URL: <http://dx.doi.org/10.1186/s40691-015-0034-9>.
- [21] Inc. InvoTech Systems. *InvoTech Systems HITEC 2012 UHF RFID Demo*. Youtube. 2012. URL: https://www.youtube.com/watch?v=_ZUr2DUE9Bg.
- [22] *Axxos OEE*. Axxos. 2017. URL: <http://axxos.se/produkter/axxos-oeo/> (hämtad 2017-04-28).
- [23] *Axxos Visualize*. Axxos. 2017. URL: <http://srv316.bluerange.se/produkter/axxos-oeo/axxos-visualize/> (hämtad 2017-04-28).
- [24] *Rs Production*. Good Solutions Sweden AB. 2017. URL: <http://www.goodsolutions.se/sv/produkter-och-tjanster/rs-production/> (hämtad 2017-04-28).
- [25] Raymond R. Burke. *How stores track your shopping behavior*. TEDxIndianapolis. 2014. URL: <https://www.youtube.com/watch?v=jeQ7C4JLpug>.
- [26] *Advanced People Counter Solution – Traffic 2.0*. RetailNext. 2017. URL: <https://retailnext.net/en/solution/advanced-people-counter-solution-traffic-2-0/> (hämtad 2017-04-26).
- [27] Debraj De m. fl. "Multimodal wearable sensing for fine-grained activity recognition in healthcare". I: *IEEE Internet Computing* 19.5 (2015), s. 26–35.
- [28] Matilda Anulf m. fl. *Flexibel utveckling av automatiserade produktionssystem*. Tekn. rapport. Institutionen för Signaler och System, Chalmers Tekniska Högskola, 2015.
- [29] Marcus Andersson m. fl. *Den flexibla klossfabriken*. Tekn. rapport. Institutionen för Signaler och System, Chalmers Tekniska Högskola, 2016.
- [30] K. Bengtsson, P. Bergagård och C Thorstensson. "Sequence Planning Using Multiple and Coordinated Sequences of Operations". I: *IEEE Transactions on Automation Science and Engineering* 9.2 (2012), s. 308–319.
- [31] Tony Murfin. "Look, No Base-Station! — Precise Point Positioning (PPP)". I: *GPS World* (2013). <http://gpsworld.com/look-no-base-station-precise-point-positioning-ppp/>.
- [32] Claire Swedberg. "Clothing, Car-Cover Manufacturers Track Work-in-Progress via RFID". I: *RFID Journal* (2016). URL: <http://www.rfidjournal.com/articles/view?15132>.
- [33] R. Want. "An introduction to RFID technology". I: *IEEE Pervasive Computing* 5.1 (jan. 2006), s. 25–33. ISSN: 1536-1268. DOI: 10.1109/MPRV.2006.2.
- [34] *Raspberry Pi*. RASPBERRY PI FOUNDATION. 2017. URL: <https://www.raspberrypi.org/> (hämtad 2017-05-02).
- [35] *PI NOIR CAMERA V2*. RASPBERRY PI FOUNDATION. 2017. URL: <https://www.raspberrypi.org/products/pi-noir-camera-v2/> (hämtad 2017-05-12).
- [36] *Error correction feature*. Denso Wave Incorporated. 2017. URL: http://www.qrcode.com/en/about/error_correction.html (hämtad 2017-05-09).

- [37] Thomas Watson. *Simple Cost Analysis for RFID Options – Choice Must Fit the Organization’s Needs and Budget*. 2015-04-28. URL: <http://itak.iaitam.org/simple-cost-analysis-for-rfid-options-choice-must-fit-the-organizations-needs-and-budget/> (hämtad 2017-05-12).
- [38] mexusbg. *Holder for Paspberry PI camera module*. 2017-03-07. URL: <https://www.thingiverse.com/thing:2158847> (hämtad 2017-03-28).
- [39] Jeff Brown. *ZBar bar code reader*. 2011. URL: <http://zbar.sourceforge.net/> (hämtad 2017-05-10).
- [40] URL: <https://kafka.apache.org/>.
- [41] *A HTTP client/server library for GNOME*. GNOME. 2017. URL: <https://wiki.gnome.org/Projects/libsoup> (hämtad 2017-05-12).
- [42] Fikret Sivrikaya och Bülent Yener. "Time synchronization in sensor networks: a survey". I: *IEEE network* 18.4 (2004), s. 45–50.
- [43] *Step7 Open Source Ethernet Communication Suite*. Snap7. 2017. URL: <http://http://snap7.sourceforge.net> (hämtad 2017-05-05).
- [44] Björn Brorström, Anders Haglund och Rolf Solli. *Förvaltningsekonomi*. Studentlitteratur, 2005.
- [45] Donald D. Chamberlin och Raymond F. Boyce. "SEQUEL: A Structured English Query Language". I: *Proceedings of the 1974 ACM SIGFIDET (Now SIGMOD) Workshop on Data Description, Access and Control*. SIGFIDET '74. Ann Arbor, Michigan: ACM, 1974, s. 249–264. DOI: 10.1145/800296.811515. URL: <http://doi.acm.org/10.1145/800296.811515>.
- [46] Meenu Dave Vatika Sharma. "SQL and NoSQL Databases". I: *International Journal of Advanced Research in Computer Science and Software Engineering* 2.8 (2012), s. 20–27.
- [47] C. Mohan. *History Repeats Itself: Sensible and NonsensSQL Aspects of the NoSQL Hoopla*. Tekn. rapport. IBM Almaden Research Center, 2013.
- [48] A. Ranta. *Database Systems - NoSQL*. lecture. Dec. 2016. URL: http://www.cse.chalmers.se/edu/year/2016/course/TDA357/HT2016/_downloads/lecture12b.pdf (hämtad 2017-05-12).
- [49] *The Java Database Connectivity (JDBC)*. Oracle. 2017. URL: <http://www.oracle.com/technetwork/java/javase/jdbc/index.html> (hämtad 2017-05-07).
- [50] Chun-houh Chen, Wolfgang Karl Härdle och Antony Unwin. *Handbook of data visualization*. Springer Science & Business Media, 2007.
- [51] Sari Kujala m. fl. "UX Curve: A method for evaluating long-term user experience". I: *Interacting with Computers* 23.5 (2011), s. 473. DOI: 10.1016/j.intcom.2011.06.005. eprint: /oup/backfile/Content_public/Journal/iwc/23/5/10.1016/j.intcom.2011.06.005/2/iwc23-0473.pdf. URL: +%20http://dx.doi.org/10.1016/j.intcom.2011.06.005.
- [52] *Industriövervakningskameror*. Baumer. 2017. URL: <http://www.baumer.com/fr-en/products/identification-image-processing/industrial-cameras/> (hämtad 2017-05-09).
- [53] *PostgresSQL - About*. PostgreSQL. 2017. URL: <https://www.postgresql.org/about/> (hämtad 2017-05-09).

- [54] Shuichi Asano, Tsutomu Maruyama och Yoshiki Yamaguchi. "Performance comparison of FPGA, GPU and CPU in image processing". I: *Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on*. IEEE. 2009, s. 126–131.
- [55] Ben Cope m.fl. "Implementation of 2D Convolution on FPGA, GPU and CPU". I: *Imperial College Report* (2006), s. 2–5.
- [56]  Datalogic MG112041-001-412B Barcode Scanner. URL: <https://www.barcodesinc.com/datalogic/part-mg112041-001-412b.htm#overview> (hämtad 2017-05-12).
- [57] Fadi Shrouf, Joaquin Ordieres och Giovanni Miragliotta. "Smart factories in Industry 4.0: A review of the concept and of energy management approached in production based on the Internet of Things paradigm". I: *Industrial Engineering and Engineering Management (IEEM), 2014 IEEE International Conference on*. IEEE. 2014, s. 697–701.

A Intervju med Teamster

Intervju 31 mars, 2017 med Niklas Lagergren hos Teamster AB.

Fråga: Hur ser det ut hos företag i industrin idag? Använder ni övervakning? Hur går man till väga, RFID-taggar?

Niklas: Ja det gör man. Man använder främst RFID-taggar som man har på större komponenter t.ex. bilkarosser, används inte på mindre komponenter. Ser olika ut i olika industrier. Det är mycket storleken på komponenten som är intressant. Man har andra sätt att spåra mindre detaljer. Vet vilken kaross/bil man producerade vid ett visst tillfälle och så vet man vilken batch av material man använt vid det tillfället. Har de t.ex. en back med plastpluggar så får de enbart ta från just den lådan och scannar även in den i datorn så de vet vilka batcher som användes till vilka bilar. RFID:n ska även plockas bort i slutet. Har även varit med om att man använder sig av QR-koder, sätter dit QR-koder eller någon form av klisterlappar för att ha koll på komponenterna.

Fråga: Har systemet i industrin själv möjlighet att reagera om det är fel saker som kommer in?

Niklas: Ja men det detekterar man oftast på andra sätt än RFID-taggar. Det vanligaste är att man använder sig av givare och sedan är de detaljer som man monterar ihop unika. Antingen är de unika eller så skapar man unika detaljer trots att unikiteten kanske inte har någon funktion. Detta gör man för att givarna ska kunna detektera så att det är rätt detalj som man monterar.

Det är också ett sätt att spåra egentligen. Att titta på hur detaljen ser ut, det behöver inte vara RFID-taggar på något sätt. Det allra enklast att tänka sig är fotoceller som känner av någon del av detaljen som bara finns på just den detaljen som man är intresserad av att detektera. Har man då tre lika detaljer så kan man göra tre små piggas på dem på olika ställen till exempel. Det är ju riktigt bonnigt men det är oftast så man gör. Det är ju tre givare som kostar max 1000 kronor styck och oftast är det en kostnads*Fråga* också.

Fråga: Om det blir fel, hur återgärdas det?

Niklas: Det blir manuellt arbete efteråt. Det är väldigt sällan som det är något som ”självläker” eller återgärdas automatiskt. Har aldrig varit med om att man designat ett automatiskt system som själv fixar ett uppkommet fel.

Fråga: Om fel detaljer skulle komma in i cellen, säger cellen ifrån då eller fortsätter den bara arbeta?

Niklas: Ja den säger ifrån och rapporterar detta. Man kan tänka sig att den stannar och sen får man något larm och sen kommer någon och återgärdar felet. Ibland så är det något som man märker ganska snabbt och då får man återgärda det på plats där men oftast är det väldigt stökigt och då får man, till exempel på bandet på Volvo, ta ut karossen ur flöde. Sen får man ta ett beslut om det här går att reparera eller inte och i värsta fall skrotar man den men det är rätt ovanligt. Man försöker reparera då det är mycket pengar i varje kaross.

Fråga: Vad är det vanligaste typen av fel som sker i ett produktionssystem?

Niklas: Det vanligaste är rena driftstopp som egentligen inte påverkar produkterna utan mera påverkar tillgängligheten i produktionsapparaten. Att man skulle sätta ihop fel detaljer är ganska ovanligt. Då är det vanligare med driftstopp med en produkt som fortfarande är okej.

Det är en hel del manuella moment och alla komponenter kommer från någon underleverantör också som har gjort någonting och det kan vara i fler led också. Men att det är något fel på detaljerna som ska sättas ihop är rätt vanligt, att det är materialberoende

då. Kvalitetskontroll bakåt i kedjan är väldigt noga, att man får rätt material in i fabriken för det kan ställa till en del om det kommer in fel material eller om komponenterna skulle vara defekta från underleverantörerna och är det små skillnad på komponenterna så märks inte det vilket kan ge allvarliga konsekvenser. Det kan vara små skillnader som har åkt igenom kontrollapparaten och så märker man det långt senare i processen när man har byggt på en hel del saker som kanske är gömda bakom annat. Något sådant kan hända. Man har ganska bra kontroll på materialet ändå så det är inte så vanligt egentligen.

Fråga: Om det skulle ske ett driftstopp, vad får då operatören för information av systemet?

Niklas: Här finns en del att göra och just när vi pratar om industri 4.0 så är det en stor sak att man kan få bättre information direkt. Ofta är det bara att systemet stannar och då får man något larm vid stationen som är lite halvkryptiskt på någon panel. Men det kanske inte pekar ut var det har gått fel utan man får förlita sig på operatörens kunskap om att den personen vet om att vad det är som hänt. Så där finns det mycket att göra och att man då direkt kan peka ut att det är den där komponenten som har fallerat och hjälpa operatören. Man skulle kunna använda sig bättre av tekniken där så man snabbt kan avvärja sig felen.

På Teamster jobbar de mest med bilindustrin men det finns mycket annan tillverkning också naturligtvis. *Niklas* inbillar sig dock att det ser ungefär likadant i de flesta industrier och man lägger mycket krut på kvalitetskontroller av materialet som kommer in. Det är sällan som man står och kör en hel dag i produktionen och sen märker man efter flera timmar att det här håller inte. Den typen av problem har man byggt bort med årens gång. Har man ingen möjlighet att automatiskt kontrollera produkterna så vet man om det och gör istället stickprover. Det kan vara att det står gubbar vid linen och tar ut produkter var tioende minut och kollar så att allt funkar. Är det något som är fel då så har man bara kört i max tiominuter med felaktiga produkter. En manuellkontroll är väldigt vanlig i industrin.

Om det är någon repa eller liknande på en detalj så hade man i många fall kunnat kontrollera detta med någon visionkamera eller liknande men det blir ganska dyrt och om produkten ändras så måste dessa programeras om. En gubbe kan man bara ge nya instruktioner och sen ändrar han sig vilket ger betydligt större flexibilitet i den manuella kontrollen. Men automatisk kontroll förekommer också och det är egentligen bättre för då är man oberoende av den mänskliga faktorn och det strävar man åt att bli oberoende av. Ett automatiskt system gör ju fel men det gör samma fel varje gång och då upptäcker man det. Det är inte det att någon är trött på måndagsmorgon.

På Volvo har man dock gått tillbaka till mer manuellt arbete, tidigare har man varit väldigt högautomatiserad men nu har man fler personer i produktionen än vad man har haft tidigare. Det har hänt en liten justering i filosofin och man vill hålla nere investeringar. Den här personalen är ofta anställda via bemanningsföretag så de kan man kalla hem och säga att det inte får jobba. Vilket är bra om man har en väldigt ojämn försäljning.

Pendel svänger lite fram och tillbaka, man har en viss filosofi en viss tid. Generellt går det mot mer automatisering i industrin så det här på Volvo är ett undantag.

Fråga: Om ni får klagomål vad brukar de handla om?

Niklas: Det brukar handla mycket om dokumentation. Det tekniska brukar fungera men det är mer saker runtomkring, mera felsökning till operatören och larmhantering. Man upptäcker detta lite sent, att det här dokumentet kanske inte alls var till någon bra hjälp för operatören när det här felet inträffade. Just rent dokumentationsmässigt kan man få en del anmärkningar faktiskt. Men snarare det än att det tekniskt havererar. Där siktar man in sig och gör ett bra jobb, det är ju trots allt tekniker som jobbar på Teamster. Så

det finns mycket att göra med hjälp till operatören och skapa ett bra system som snabbt kan avhjälpa fel. Där är det oftast lite undermåligt kan tyckas.

Det finns väldigt bra hjälpmedel idag och man kan få statistik till förbannelse. Men det gäller att veta hur man ska använda det här och hur det ska presenteras till en operatör på bästa sätt.

Fråga: När vi kör vår övervakning av komponenter vilken data tror du är mest intressant att få ut?

Niklas: Ur ett designperspektiv är det intressant att veta den optimala sammansättningsmetoden alltså vilken ordning som är den bästa att göra detta på. Men ur ett uppföljningsperspektiv så är det mer att lagra data för varje byggt torn. Hur lång tid det tar har man oftast redan bestämt att ett visst momentet får ta ett visst antal sekunder och sker detta på ett automatiskt sätt som är lika tidskrävande varje gång så är det inga variationer. Det är skillnad om man bygger något manuellt, då kan man ha ganska stora variationer på hur lång tid det tar.

Ur ett automatiskt perspektiv är data lagring inte lika intressant utan det är snarare intressant att kolla upp var det inträffade något fel och logga detta felet. Den typen av feluppföljning kopplad till ett visst moment i produktionen är intressant. För att kunna förbättra just den delen i processen så jobbar man oftast mycket med att personer sitter i industrin och förbättra och bygger bort de vanligaste felkällorna. De använder sig mycket att larmloggar som de sitter och kollar på och ser vad det är för larm som kommer upp oftast.

En annan sak som man kan tänka sig att man loggar är tillståndet hos produktionsapparaten alltså vilket tillstånd utrustningen befann sig i under monteringen.

Frågor: Arbetar ni mycket med industri 4.0 på Teamster?

Niklas: Det har blivit lite av ett modeord och det ligger väl någonting i det men vi som jobbar i branschen tycker väl inte att det är så mycket nytt egentligen. Det är mer att industrin har paketerat ett begrepp som är lätt att sälja in till de som inte är så insatta. Vilket är bra på ett sätt då det kanske kan ge pengar till projekt och man kan motivera saker och ting på ett enklare sätt. Samtidigt går tekniken framåt med trådlös kommunikation osv. så det är väl inte fel. Men någon praktiskt nytta är svårt att se rent tekniskt, men det är svårt att säga då det finns många andra branscher än bara bilindustrin.

Men man fastnar i nästa stadium att den här komponenten är jätteduktig på att förmedla en massa uppgifter om sitt tillstånd och var den håller på med. Men problemet ligger snarare på mottagarsidan då, något system ska samla in de här komponenternas statusmeddelanden och rapporter om hur de mår. I det skedet ska man göra något vettigt med den här datan och det är det som är problemet att få till det.

Exempel: Om man tar en rullbana en vanlig frekvens omriktad drift. För några år sedan hade man ett interface till den här frekvensomriktaren, man hade kanske 30 ingångar och 30 utgångar med dubbelsignaler och hade kanske tio olika larm som du kunde få från den (överström, övermoment, övertemp lite olika felparametrar och så). Det var något som man kunde presentera för en operatör på ett operatörgränssnitt att nu är det överström i motorn här. Men idag kan du ansluta på finett till en sådan här frekvensomriktare och få ut tiotusen larm eller nåt och den kan berätta exakt vad den har för temperatur på olika ställen. Den kan berätta en massa olika data för dig och det är säkert jättebra men jag menar vi som integratörer kan inte presentera den här datan på ett vettigt sätt för en operatör. Det kräver en väldig expertkunskap ifall någon ska gå in och titta på alla de här olika larmen och förstå dem.

Det är här industri 4.0 har sin utmaning, vad vill man använda och vad vill du uppnå.

Det här har man förstått länge och det är inget nytt men det är ändå där som svårigheten ligger. Att ha någon verklig nytta av den här datan.

Lite om vårt projekt:

Tiderna är viktiga och speciellt när man har manuella moment så kan det skilja väldigt mycket på hur lång tid saker och ting tar. Det är vanligt att en viss operatör loggar in sig på en viss terminal. Så man vet vem det var som jobbade på stationen, så man har mycket koll på individnivå om man klarar sin cykeltid eller inte. Då tvingas operatören skriva en rapport och det är ganska vanligt på olika industrier att man går ner på individ nivå och granskar hur den personen arbetar.

Man gör bedömningen att det här momentet ska ta 8 sekunder och då vill man logga det att det verkligen uppnås. Så tider är typiskt en sådan sak som man loggar. Men jag har inte varit med om att massa event loggas. Det är rätt ovanligt, man brukar vara på större puckar och loggar totalcykeltid eller kanske loggar transporttid in till stationen så man loggar varje robotstid. Men att man skulle dela upp det ytterligare och säga att nu rörde sig roboten mellan hemmaläget och tippdressen och det tog så här lång tid, det gör man inte.

Ett tips är att man skulle ha ett moment mitt i så att tre klossar lägger robotarna och sen lägger en gubbe den fjärde och sen ska robotarna lägga resten. För så ser det ofta ut att man har delmoment som är manuella och där får man in en varierande faktor eftersom det tar olika lång tid och det hamnar lite snett. Det är rätt vanligt med delmoment som är manuella som kan ställa till en del.