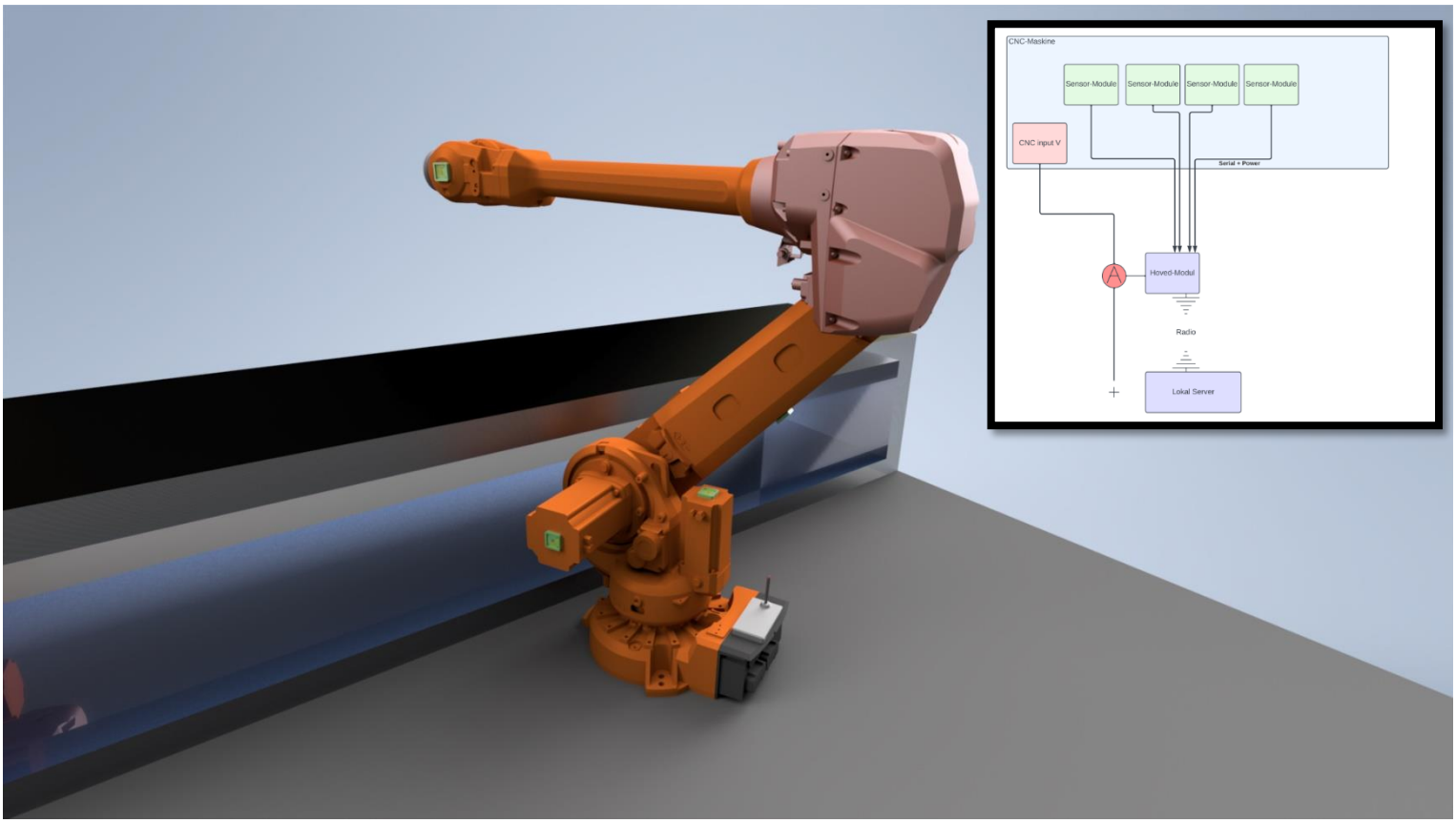
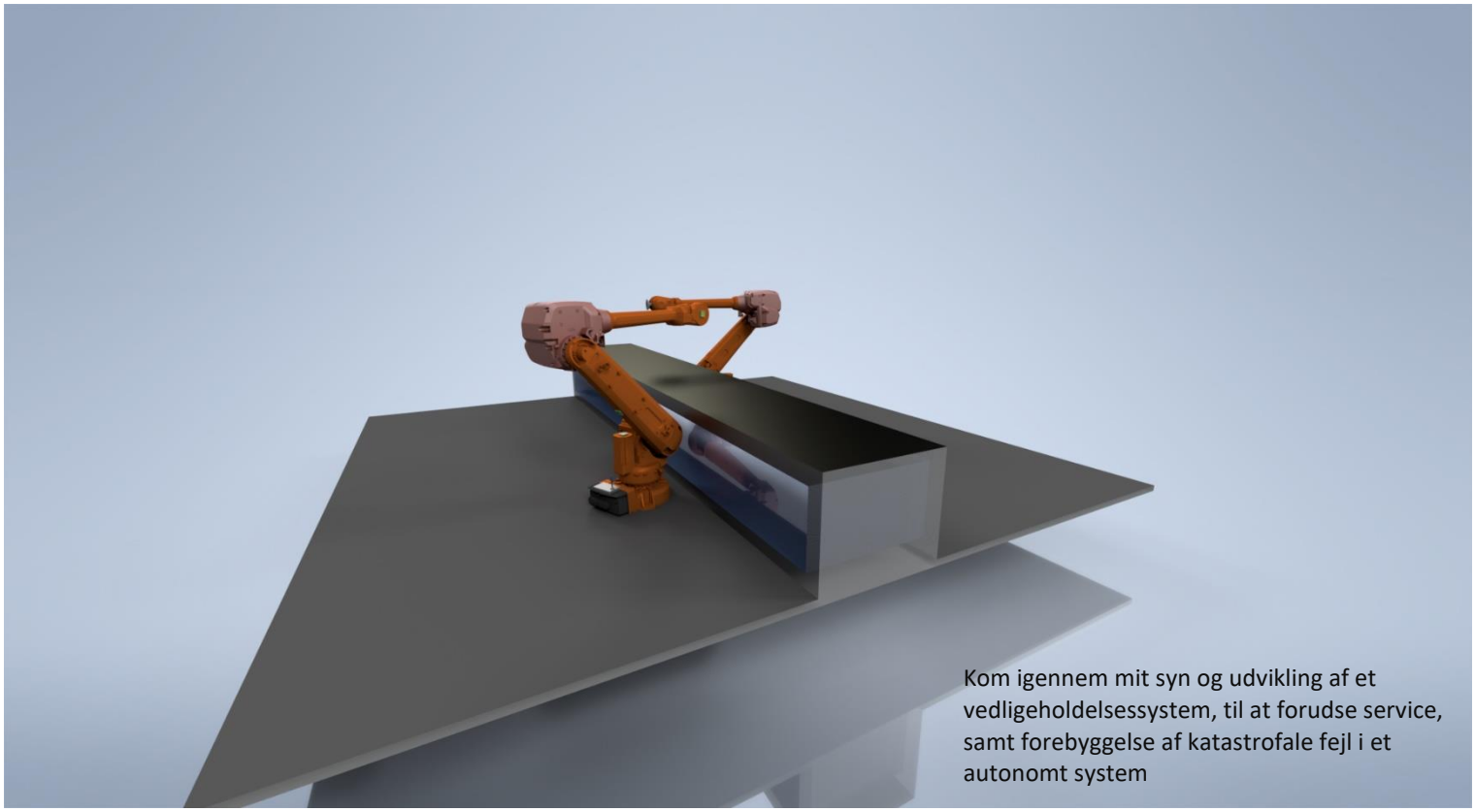


PREDICTIVE MAINTENANCE



Kom igennem mit syn og udvikling af et vedligeholdelsessystem, til at forudse service, samt forebyggelse af katastrofale fejl i et autonomt system

Christian LS Jørgensen
Chri10c9@Eamv.dk

Mit syn på hvordan et system muligvis kan bygges og udvikles som styre service og produktion overvågning på CNC-maskiner.

Intro

Under besøget af BILA-data fik vi fremvist en Case.

Hvorfor?

Jeg har alene valgt at tage et forsøg på at komme frem med min ide til hvordan dette muligvis kan løses. Jeg har ikke det rigtige udstyr til at kunne lave nogle forme for test af forskellige sensorer. Derfor vil det meste der er skrevet i denne rapport være baseret på tanker og erfaring fra tidligere projekter. Jeg har igennem hele mit liv være lidt af en nørd når det kommer til teknologi og har altid synes det er sjovere at bygge selv end at finde en som sælger det jeg mangler. Jeg har igennem tiden lavet lidt IOT.

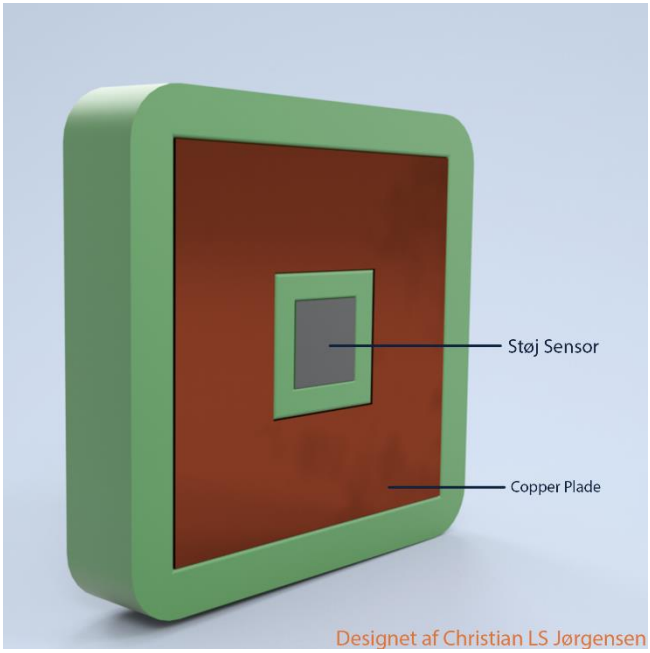
Jeg studerer lidt pt. Som datamatiker hos EAMV, men jeg læser videre til software ingeniør når jeg færdig og den eneste grund til at jeg ikke læser det nu er placeringen på uddannelsen.

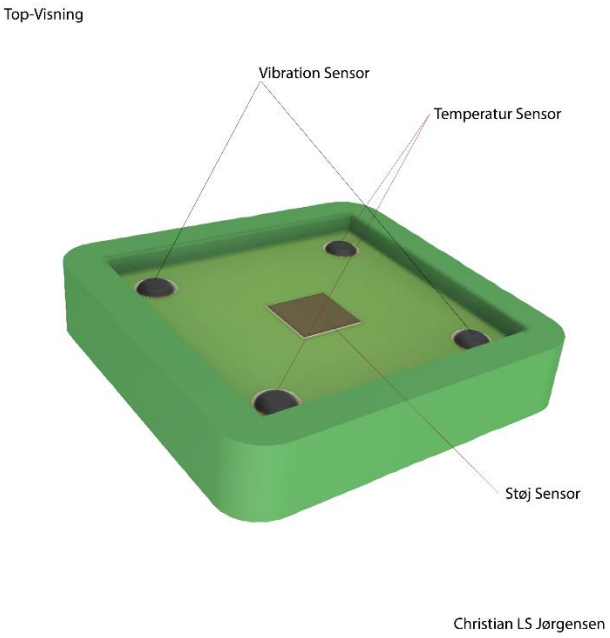
Til Slut…

Da jeg selv har et eksamensprojekt snart, er der noget som nok kunne bruge mere arbejde, men har ikke haft tiden til at færdiggøre det. Jeg har dog alligevel valgt at sende det jeg har lavet indtil videre, hvis der skulle være noget som var interessant.

Sensor-Modul

Jeg har lavet en model hvordan en sensor muligvis kunne fremstilles. Kobberpladen har en overlap på 1.1 ± 0.01





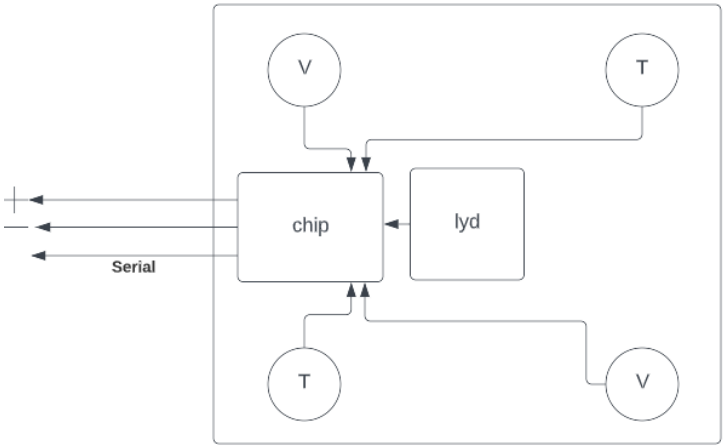
Sensor modulet består af en række sensorer, som har til opgave at opsamle data fra støj, vibration og temperatur. Det opsamlet data bliver sendt direkte til hoved-modulet, som samler og sender det videre til en lokal server, som har til opgave at bearbejde det.

Jeg har valgt at bruge 2 vibration og 2 temperatur sensorer. Det har jeg valgt for at mindske fejl-data, og nedsætte falske alarmer. I dette design er der ikke taget højde for fejl, angående støj sensor, men det vil helt sikker være en fordel. Fordi jeg har valgt at lave en kobberplade som skal opsamle vibrationer og varme, skulle alle sensorer som har kontakt med pladen måle det samme, hvilket gør det nemt at finde fejl i evt. sensor modul.

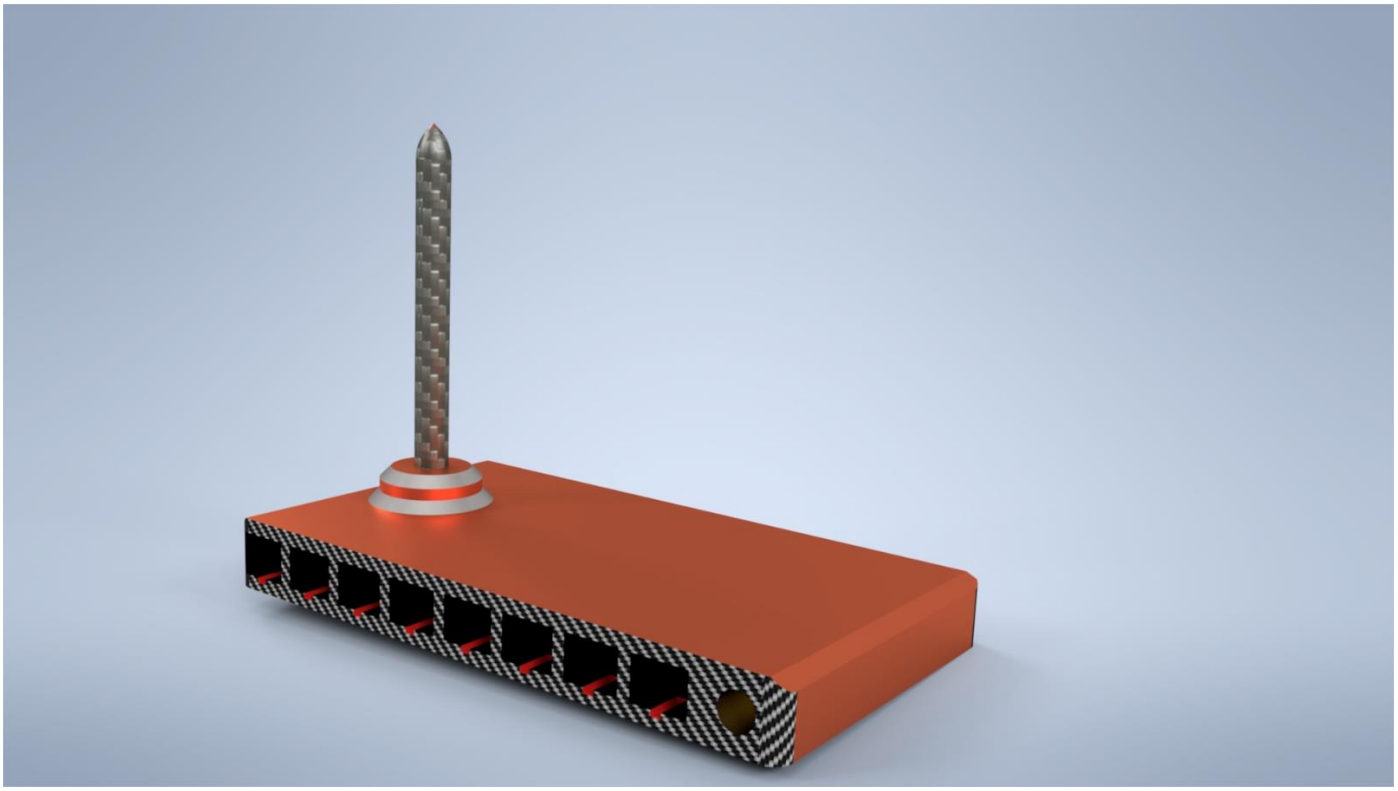
Sensor-modulet er tilsluttet hovedmodulet ved hjælp af en Serial-protokol. Jeg har valgt den tilgangs metode, fordi at sensorerne skal alligevel have en strømtilslutning, og derfor passer det med godt at fører en Serial forbindelse med. Opsamlingen sker ved hjælp af en mikroprocessor, som skal sende de givet data til hoved-modulet. Sensor-modulet skal af design ikke vide hvor den sidder eller hvem den er, det er op til hoved-modulet at give den et ID via sit Serial Input port. På den måde er det nemt at sætte en ny til uden nogen form for opsætning af software mm. Hvordan de skal monteres på de forskellige komponenter, har jeg ikke nogen erfaring I så derfor har jeg lavet et design som mangler top. Jeg har en ide om at det muligvis kan monteres med 4 stærke magneter, men om det vil skabe andre problemer. Det har jeg ikke muligheden for at teste!

Mikrokontrollers opgave…

- Opsamle data fra sensor.
- Send data via Serial til hoved-modul
- Vent
- Gentag.

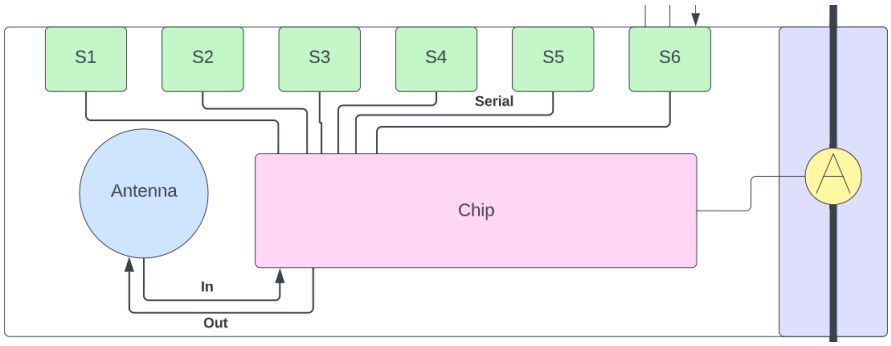


Hoved-Modul



Hoved modulet har til opgave at samle alt data og mærker det inden det sendes videre. Det er også muligt at give dette modul adgang til at stoppe maskinen hvis en kritisk fejl opstår. Den er meget tæt med alle andre sensor, og kan derfor være utrolig hurtig til at stoppe det inden alt for store fejl kan forekomme. Og derfor nedsætte udgifterne ved vedligeholdelse.

Til højre se vi et forenklet diagram over elektriske forbindelser. S1[1] -> S6 er input-porte, som består af en Serial og power forbindelse, som gør det nemt at tilslutte flere sensormoduler. De er alle sammen forbundet til en mikrocontroller, som har til opgave, at sende data videre via en antenne forbindelse. I



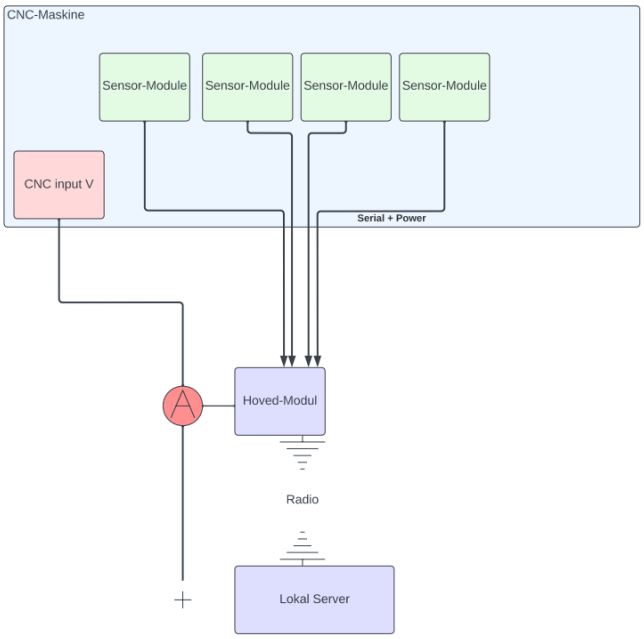
Jeg har valgt at tage radio i brug fordi det er rimelig stabilt ikke afhængelig af andre parameter såsom internet, som kan fejle.

"Vender tilbage"

[1] Serial forbindelse

Server-Modul

serveren har en opgave det er at opsamle data og sende det ud til de forskellige services, som virksomheden har valgt. Serveren har også til opgave at gemme det opsamlet data til senere brug.



Data Opsamling

Serveren vil systematisk spørge ud til hovedmodulerne om deres værdier. De sendes rå til serveren.

$$Hm{:}\, ID\{\, S1{:}\, v_1, v_2, v_n \gg S2{:}\, v_1, v_2, v_n \gg\}$$

En mulig måde at sende data til serveren kan være som det skrevet ovenover. Det virker ved at den skriver hvilket H-modul der er tale om. Derefter sendes alle de data som sensor-modulerne har opsamlet. $\gg$ siger at vi skifter til en anden sensor modul. Hele H-modulet er opdelt med $\{\ \}$ hvilket gør det nemt at sorterer.

Når serveren modtager det, vil det med intervaller gemme stikprøver på databasen som kan bruges til at se hvordan udviklingen er i et system. Når en klient tilslutter sig serveren vil de sidste data blive sendt og har derfor ikke direkte adgang til databasen. Jeg har valgt den fremgangsmåde fordi jeg ikke ser det nødvendigt at gemmer alt der kommer ind, men snarer gemme det med en fast tidsramme for nemmere at kunne analysere på det.

Serveren jeg har valgt at lave som en test bruger Tcp/ip og WPF som visualisering. Valget skyldes kun at jeg ikke har så meget erfaring med at arbejde med Asp.net eller andre web-servere endnu. Serveren har 2 porte som bliver brugt. Første port har kun et formål og det er at sende data ud til klienten og den anden har til opgave at modtage ordre fra klienten. Den kan bruges til at ændre server informationer eller andre ting. Den anden port er mere ment for service og ikke til alm brug for kunden.

En af de ting som jeg har lavet store overvejelser om er hvordan det skal sendes via Tcp/ip for at holde det minimalt.

Igennem lidt forsøg er jeg nået til at det sendes via en string som er meget simplificeret. En string som kan sendes via tcp kan se sådan ud.

$$\#Plast1!\, \$Robot\ 1!\, @TopArm!\, 20!\, 1!\, 23!\, 98!\, \$Robot\ 2!\, @TopArm!\, 20!\, 1!\, 23!\, 98!$$

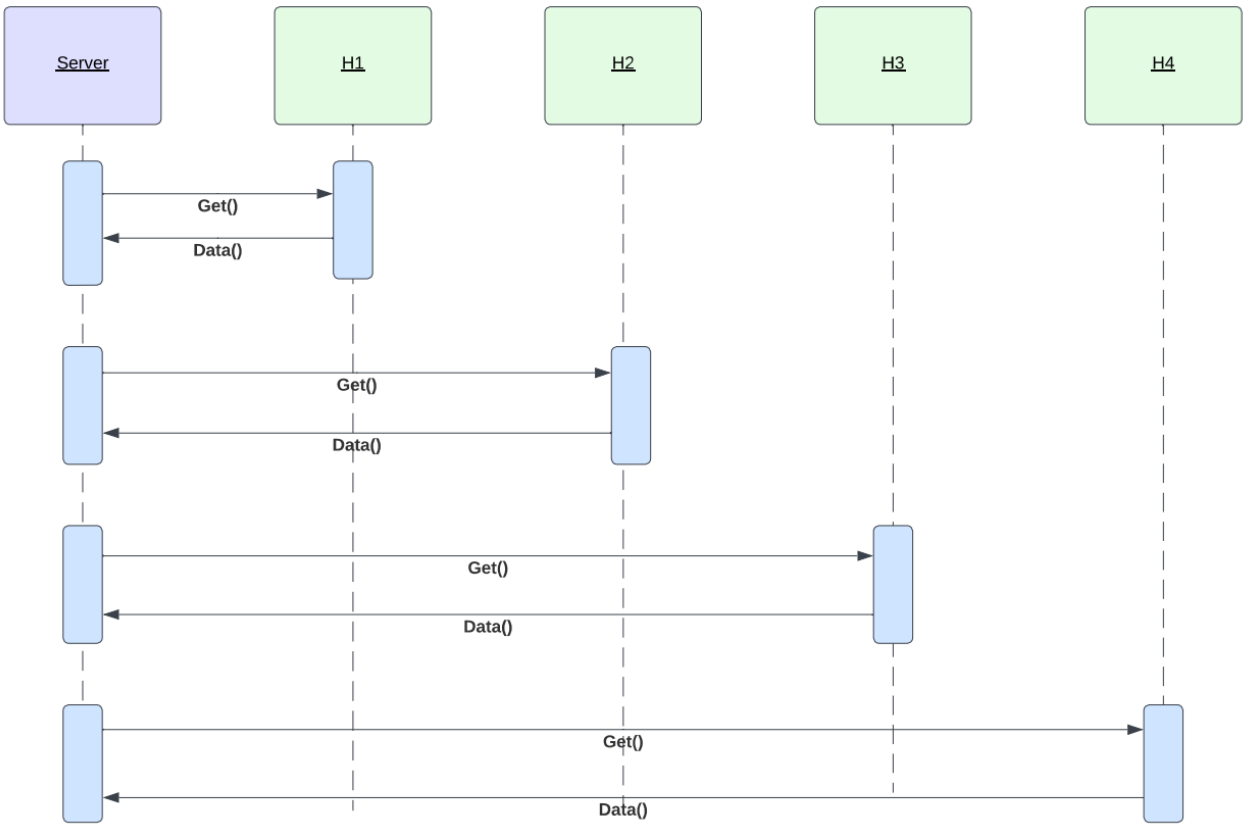
Forskellige chars har hver deres betydning.

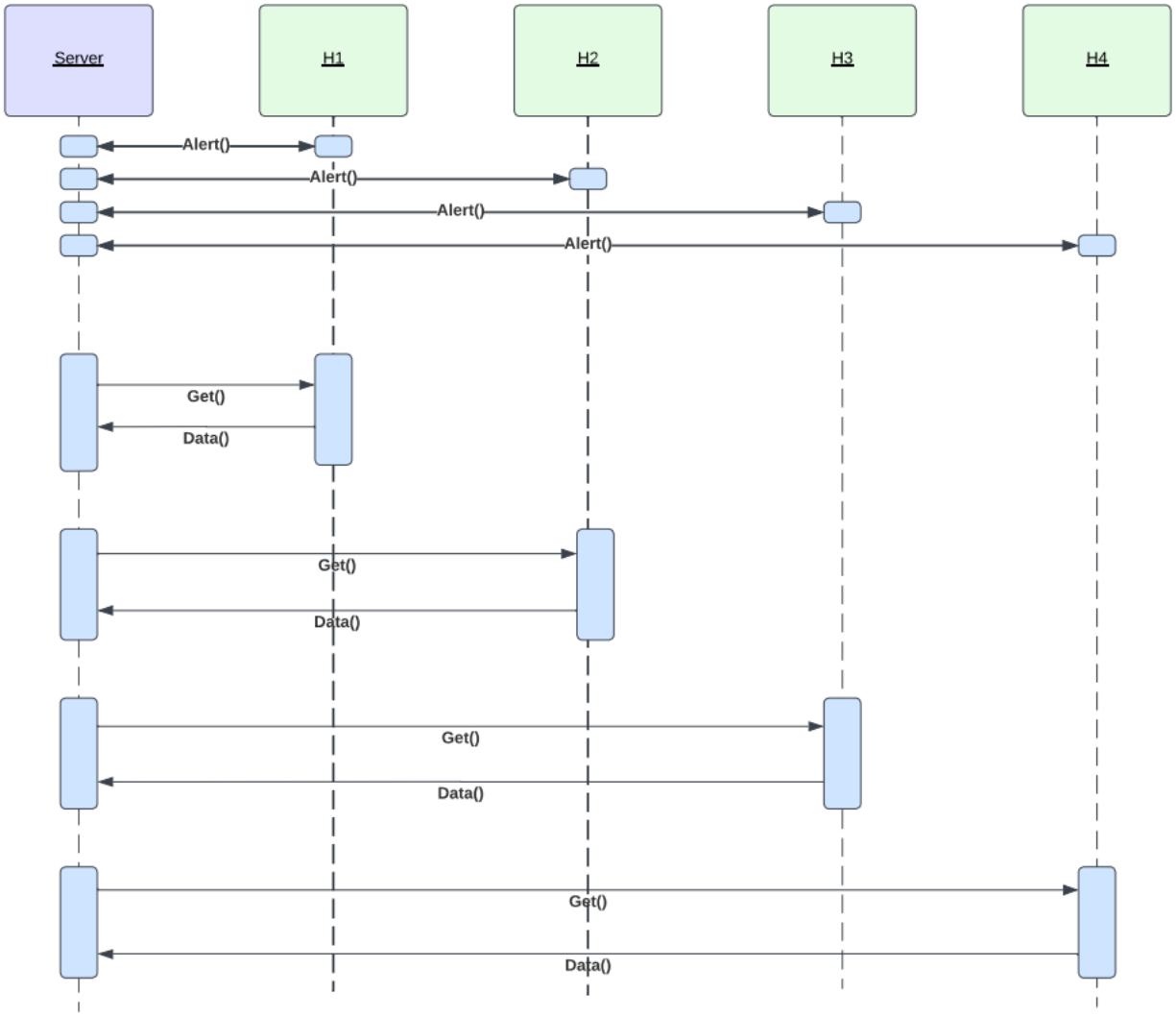
$\# = systemlinje$ systemlinje er en hel produktions linje
$\$ = HovedModul$ tilsluttet en CnC maskine. Og har tilsluttet x antal sensormoduler
$@ = SensorModul$ har et x antal sensorer.
$! = Slut$ fortæller bare at det er slutning på noget information.

På den måde kan jeg sende meget uden at det skal fylde meget. Den anden metode som jeg kender til er via klassen Serializable, men jeg har valgt at det vil blive for langt da der er x antal moduler og derfor kan det blive en stor byte[], som jeg gerne vil holde til et minimalt.

Det er op til klienten at oversætte disse data og på den måde spare vi kræfter på at serveren skal tage sig af det, som kan være en stor fordel hvis der er mange tilsluttet.

Kommunikation: H-modul og Server





Fortsætter på næste side!

Forrige side så vi på 2 forskellige sekvens diagrammer, disse diagrammer er meget ens, bort set fra at den ene inden den spørger efter data, tjekker om nogle af hovedmoduler, som sender et advarselssignal. Signalet skal bruges hvis der bliver opdaget noget kritisk. Og på den måde, hurtigt få sendt et signal ud til dem som arbejder med maskinen, det kan være et SMS-system eller lignende.

Ellers vil den tage kontakt til H-Modulet og bede om at få tilsendt sit data hvorefter det bliver gemt, og behandlet efter kundens ønske.

Server-Opbygning

Demo
(C# .net6 projekt)

Klienter

Farver og overblik

Når det kommer til at få et hurtig overblik over et systems helbred, så har jeg erfaret at lyskryds-metoden er yders effektiv. Metoden er meget simple og går på at rød er stop, gul er advarsel og grøn betyder at alt går som smurt. Ud fra disse farver kan en forbi passerende hurtig se hvordan systemets bevægelige dele har det. Og når den går fra grøn til gul, skal der snart laves service på maskinen. På den måde kan der planlægges ud fra produktionens side, hvornår det vil være bedst at stoppe maskinen. Og derfor sparepenge op kritiske stop.

Mobil

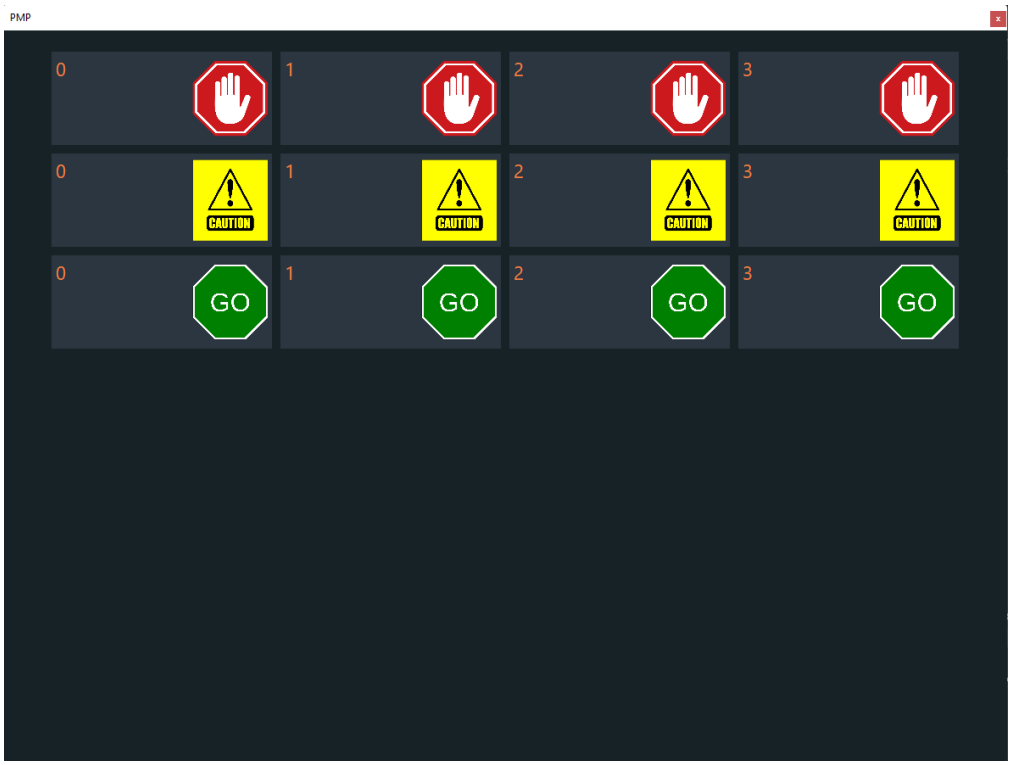
Jeg råber ikke hurra for mine tekniske evner inden for Gui design, så det hele kan godt blive lidt firkantet, men det beskriver rimelig min ide til hvordan en telefon app skal se ud. Hele idéen er at lave det så simpelt og så stort som muligt, alle maskinerne vil blive vist i en piloterings-liste, hvor dem med rød er kritiske fejl. Serveren vil også hvis der skulle ske en kritisk fejl sende en SMS ud til dem som er ansvarlig for maskinen.

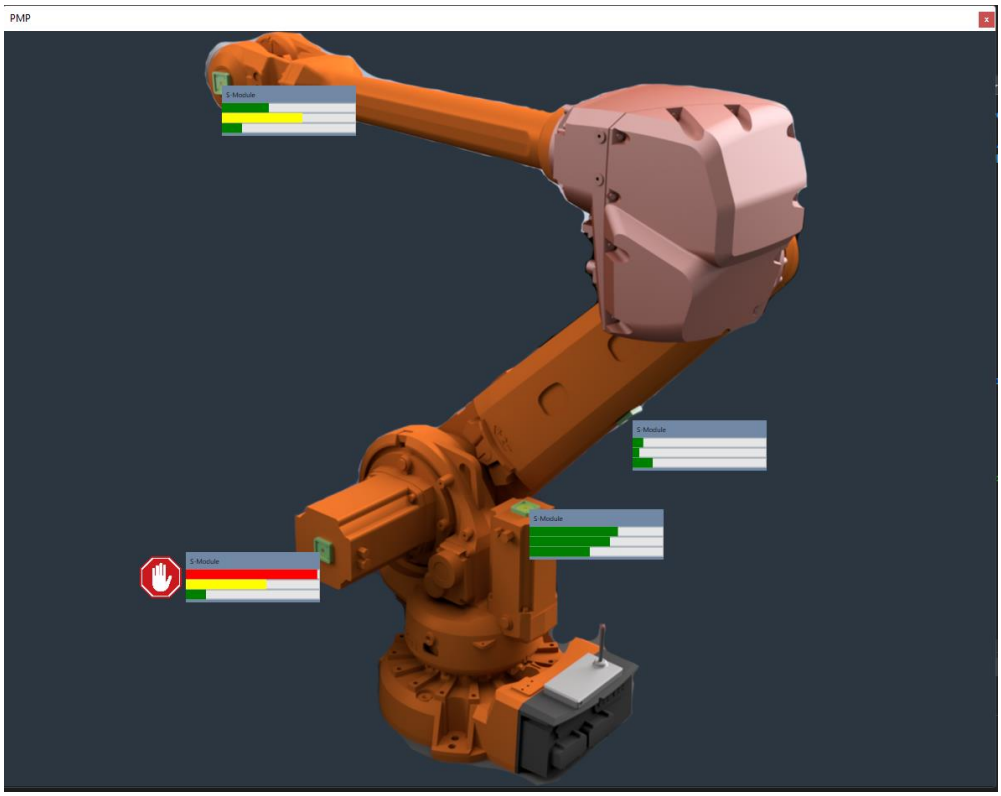
Når noget melder sig som gul (Caution) så vil der være en vurderingssag for personen som har ansvaret. Ved at klikke op boksen, vil de data som er for høje blive fremvist. Her vil det også være muligt at se større analyser af hvorfor den har meldt fejl. Det vil også være en god ide at lave en måde for operatøren at se om der allerede er bestilt service til maskinen eller systemet. På den måde, så giver det et hurtig overblik.

Jeg ser det som en god ide at have alle maskiner på listen, fordi så er man sikker på at der er en forbindelse. Hvis en maskine ikke vises, betyder det at der er sket et eller andet, som skal undersøges.

Desktop

Når programmet åbnes, vil der vises en liste over H-moduler som mangler opmærksomhed. Som det vises i prototypen, så vises der ikke hvilken linje det enkle H-modul er linket til. Det vil helt sikker gøres det hurtig og nemmere at hurtig se hvor der er sket en fejl. Der kan evt. også tilføjes hvilken S-modul som har for høje tal.



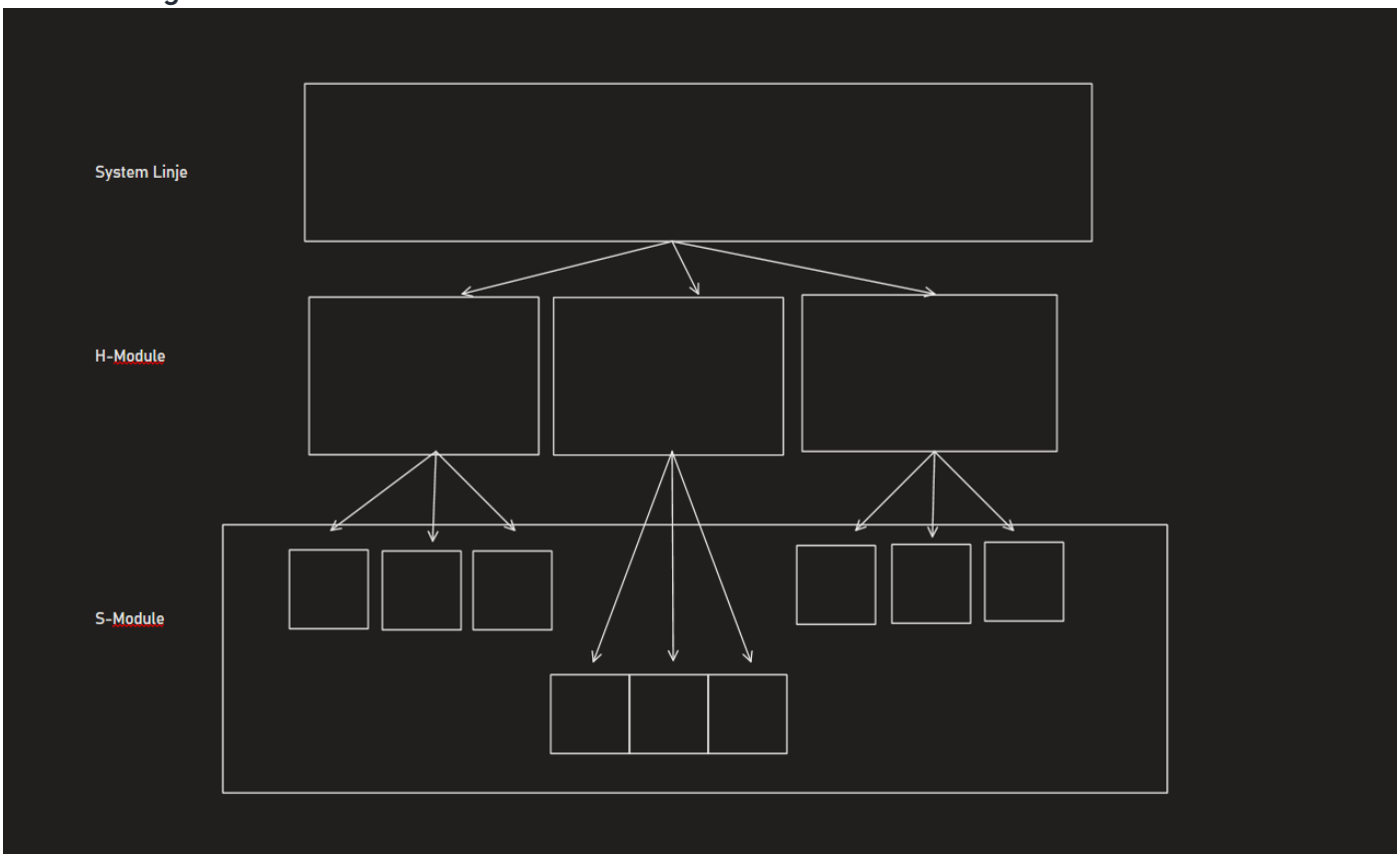


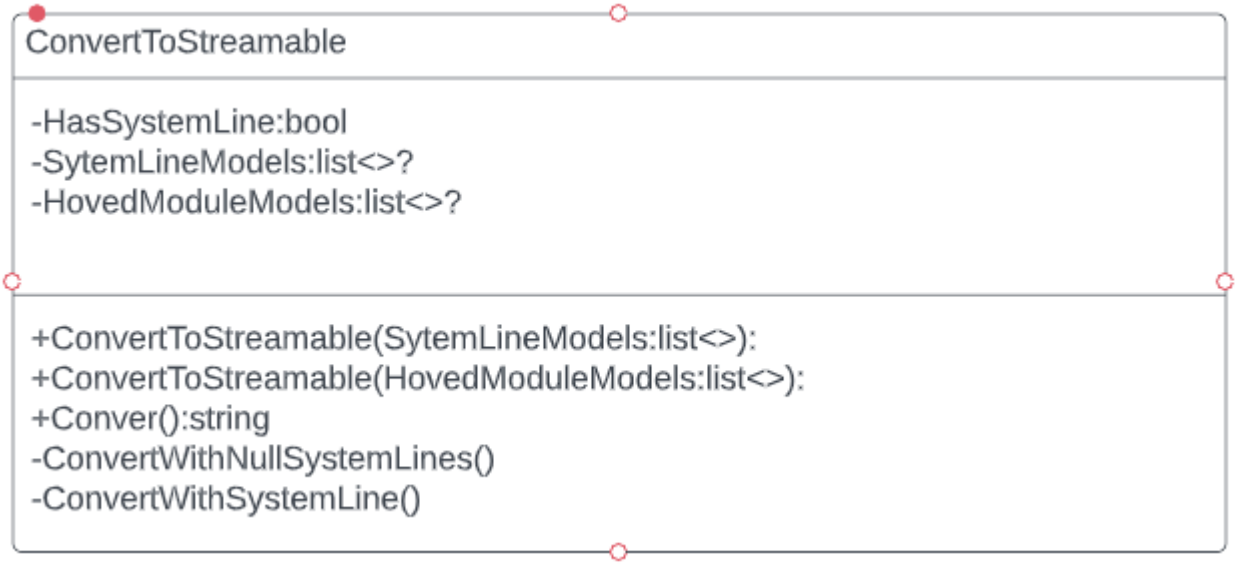
Når kunden går ind på et H-modul, så vil der vises et 3d visning af maskinen og dets sensormoduler samt deres placering på maskinen, på den måde er det hurtigt og nemt at analysere hvor fejlen kommer fra visuelt.

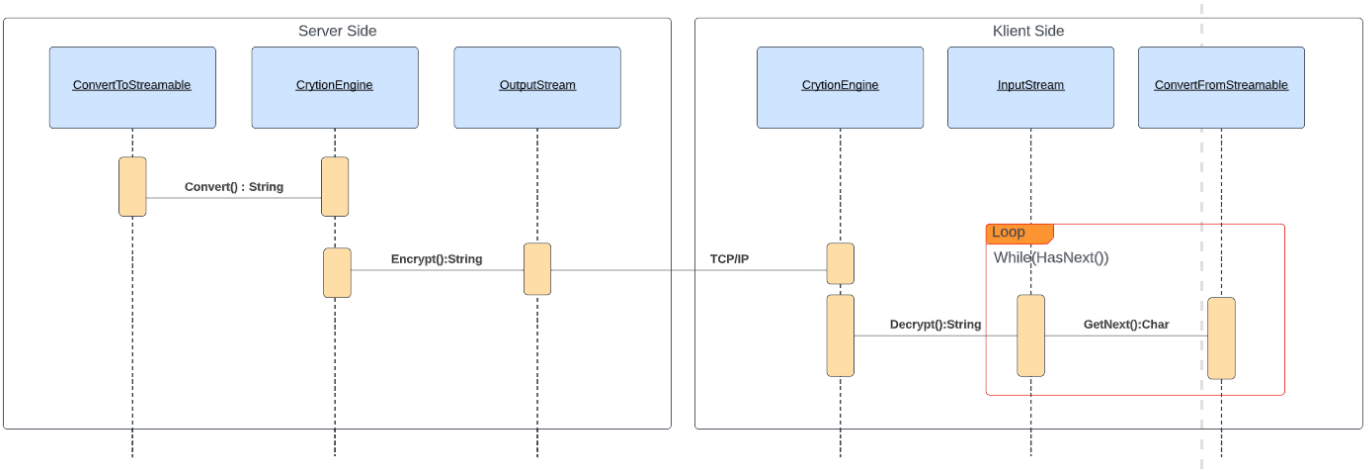
Arkiv

Forskellige demo typer som ikke er en del af rapporten endnu. Jeg har valgt at vedhæfte alt for at mere fremvise hvad jeg har arbejdet med.

Nedarvning af moduler






Server Side
ConvertToStreamable
CrytionEngine
OutputStream
Convert() : String
Encrypt():String
TCP/IP
Klient Side
CrytionEngine
InputStream
ConvertFromStreamable
Loop
While(HasNext())
Decrypt():String
GetNext():Char