NTNU
Kunnskap for en bedre verden

Institutt for teknisk kybernetikk

TTK4235 - Tilpassede datasystemer

Heisprosjekt

Gruppe 5

Finn Gross Maurer

Tarek El-Agroudi

Mars, 2021

Innhold

1 Sammendrag

Denne rapporten diskuterer systemutviklingen for styrings- og bestillingsystemet for en enkel heismodell i henhold til den pragmatiske V-modellen (Mathisen (2021)). Den overordnede arkitekturen karakteriseres ved separate moduler for styringssystemet, kø-logikk, hardware og en timer. Kvalitet i henhold til kravspesifikasjoner ble sikret ved enhetstesting samt en omfattende integrasjonstest. Lett kopling mellom moduler gjør systemet skalerbart og leselig, men koden preges av noe overflødighet og unødvendig repetisjon.

2 Funksjonsspesifikasjon

Det tilpassede datasystemet skal drive en heismodell på 4 etasjer. Heisen kan få OPP- og NED-bestillinger fra hver etasje utenom i 1. og 4. etasje, der den henholdsvis bare kan få en OPP-bestilling og en NED-bestilling. Fra innsiden av selve heisen kan den få en bestilling til hver etasje. Disse bestillingene simuleres ved å trykke på knapper på en betjeningsboks. Alle bestillingsknapper har spesifikke lamper knyttet opp mot seg. I tillegg har heisen en obstruksjonsbryter som skal simulere obstruksjoner i døren og en stoppknapp for nødsituasjoner. Hver etasje er også utstyrt med en Hall-effektsensor.

Detaljerte kravspesifikasjoner ble gitt fra oppgaven og vil derfor ikke bli videre spesifisert her. Da det blant annet i testseksjonen blir referert til denne, er spesifikasjonene lagt ved i Appendix A.

3 Overordnet arkitektur

I designet av heisen på overordnet arkitekturnivå ble det bestemt at kontrollsystemet til heisen og køen burde være to uavhengige moduler, slik at kø–modellen enkelt kan byttes ut uten å gjøre endringer i kontrollsystemet. Det ble gjort med hensyn til at kø-logikken lett skulle kunne endres uten å måtte endre på kontrollsystemet, noe som utdypes i seksjon 6.

Programflyten styres av en tilstandsmaskin og en tilhørende kontrollmodul, Elevator Systems Control, samt en separat tilstandsmaskin for køen. Dette viste seg å være et naturlig valg, da det er lett å finne gjentagende tilstander for en heis og da heisens oppførsel er veldig avhengig av den fysiske tilstanden den befinner seg i.

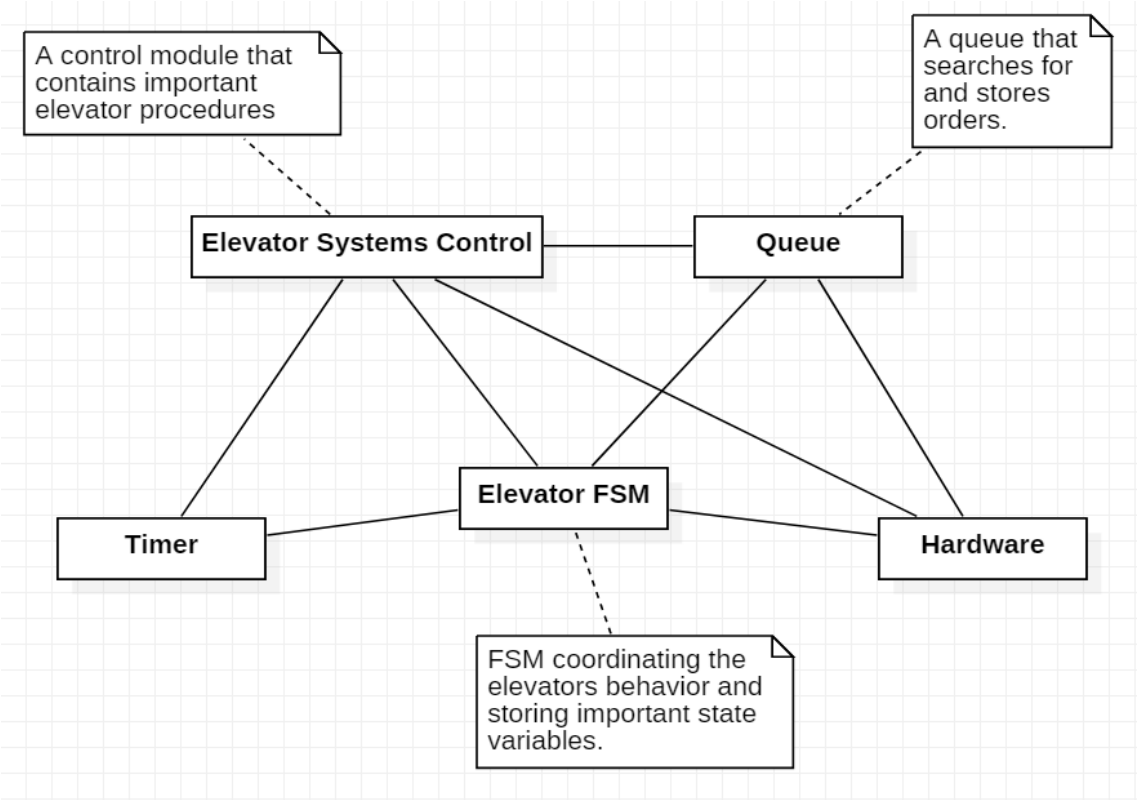
I tillegg er det behov for en enkel timermodul og et grensesnitt ned mot hardwaren. Den overordnete arkitekturen er gjengitt i figur 1, og diskutert nøyere i seksjon 4.2.

3.1 Design av styringssystemet

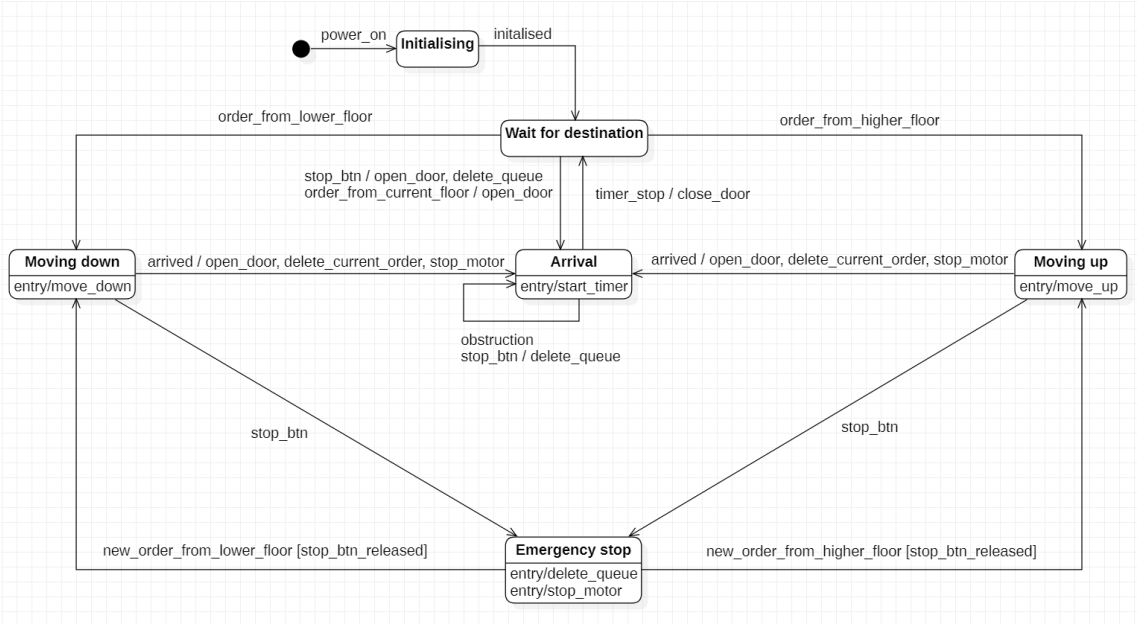
Heisens styringssystem tar kun inn neste destinasjon gitt av køen, for å sikre lett kopling mellom denne og kø-modulen. Tilstandsmaskinen består av seks tilstander:

Initialising, Wait for destination, Moving up, Moving down, Arrival og Emergency Stop.

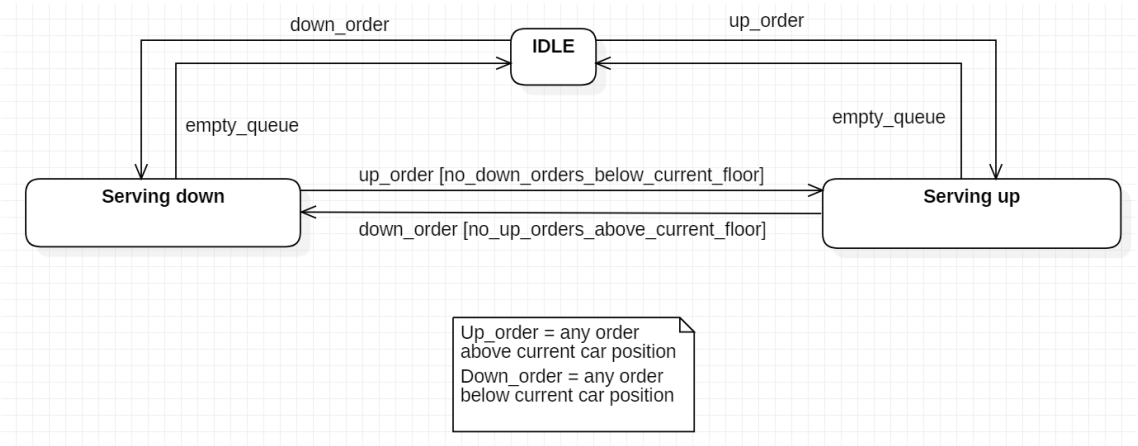
Tilstandsdiagrammet er gjengitt i figur 2. Flyten mellom tilstandene bestemmes av hvor neste destinasjon er i forhold til gjeldende etasje, samt signaler fra stoppknappen og obstruksjonsbryteren. For eksempel, vil transisjonen order_from_higher_floor aktiveres når neste destinasjonen er fra en høyere etasje enn heisen.



Figur 1: Overordnet systemarkitektur. Innholdet i figurene er på engelsk for å bedre kunne knyttes opp mot koden. FSM er forkortelsen for begrepet 'finite state machine', altså tilstandsmaskin.



Figur 2: Tilstandsdiagram for heisen.



Figur 3: Tilstandsdiagram for køen.

3.2 Design av køsystemet

Køsystemet kjennetegnes ved at den fysiske kjøreretningen avgjør hvilken ordre heisen vil betjene. Den vil dermed på vei opp bare betjene OPP- og INNSIDE-bestillinger, før den til slutt enten går i en IDLE tilstand eller betjener høyeste ned-bestilling. Det motsatte gjelder for heisen når den er på vei ned. Dermed vil heisen så lenge den får bestillinger gå i en kontinuerlig løkke der den veksler mellom å BETJENE OPP og BETJENE NED i samsvar med heisens bevegelse. Disse tilstandene er gjengitt i figur 3.

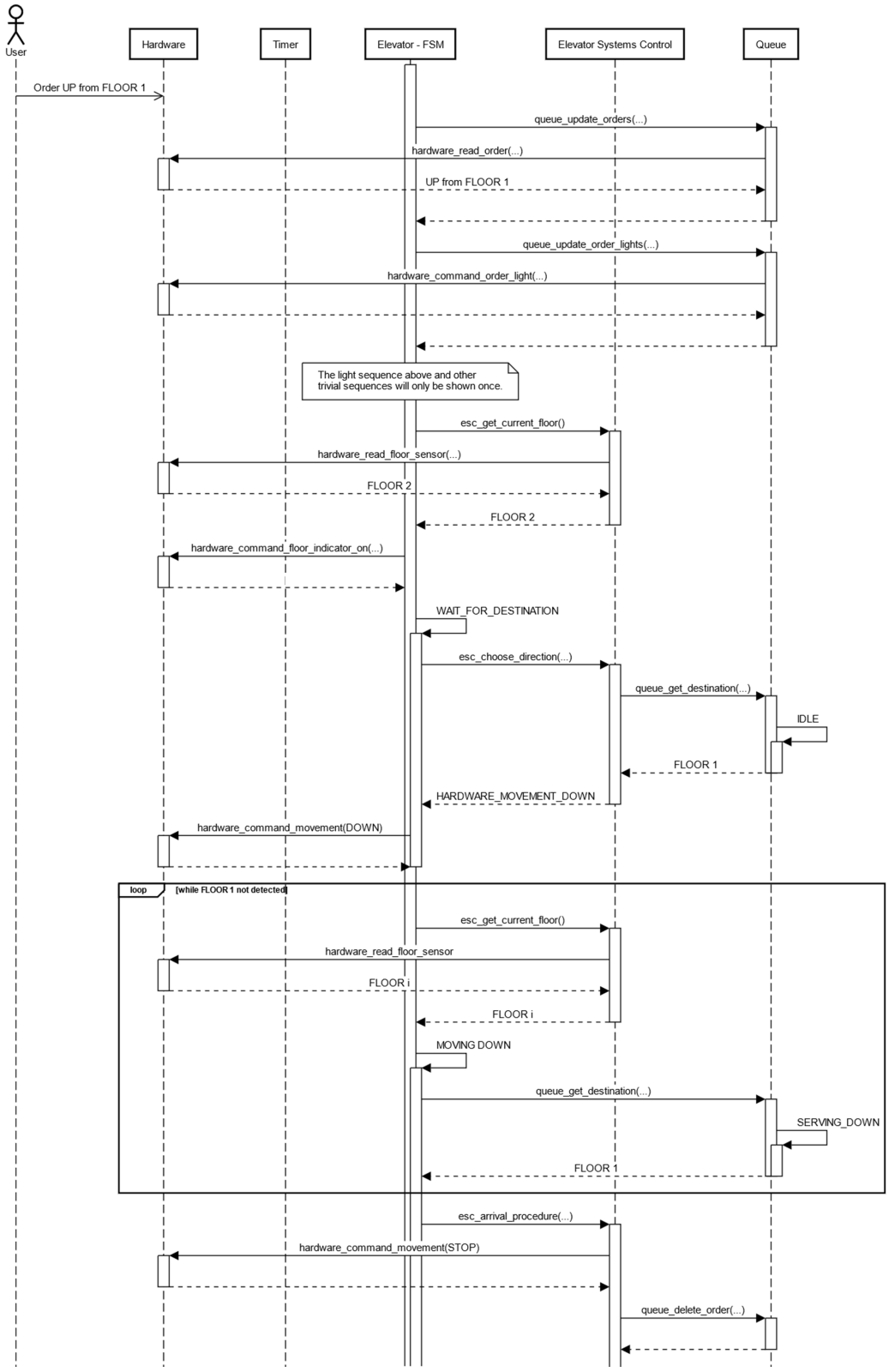
Inspirasjonen til køsystemet er hentet fra heisen på El-bygget E/F, NTNU Gløshaugen.

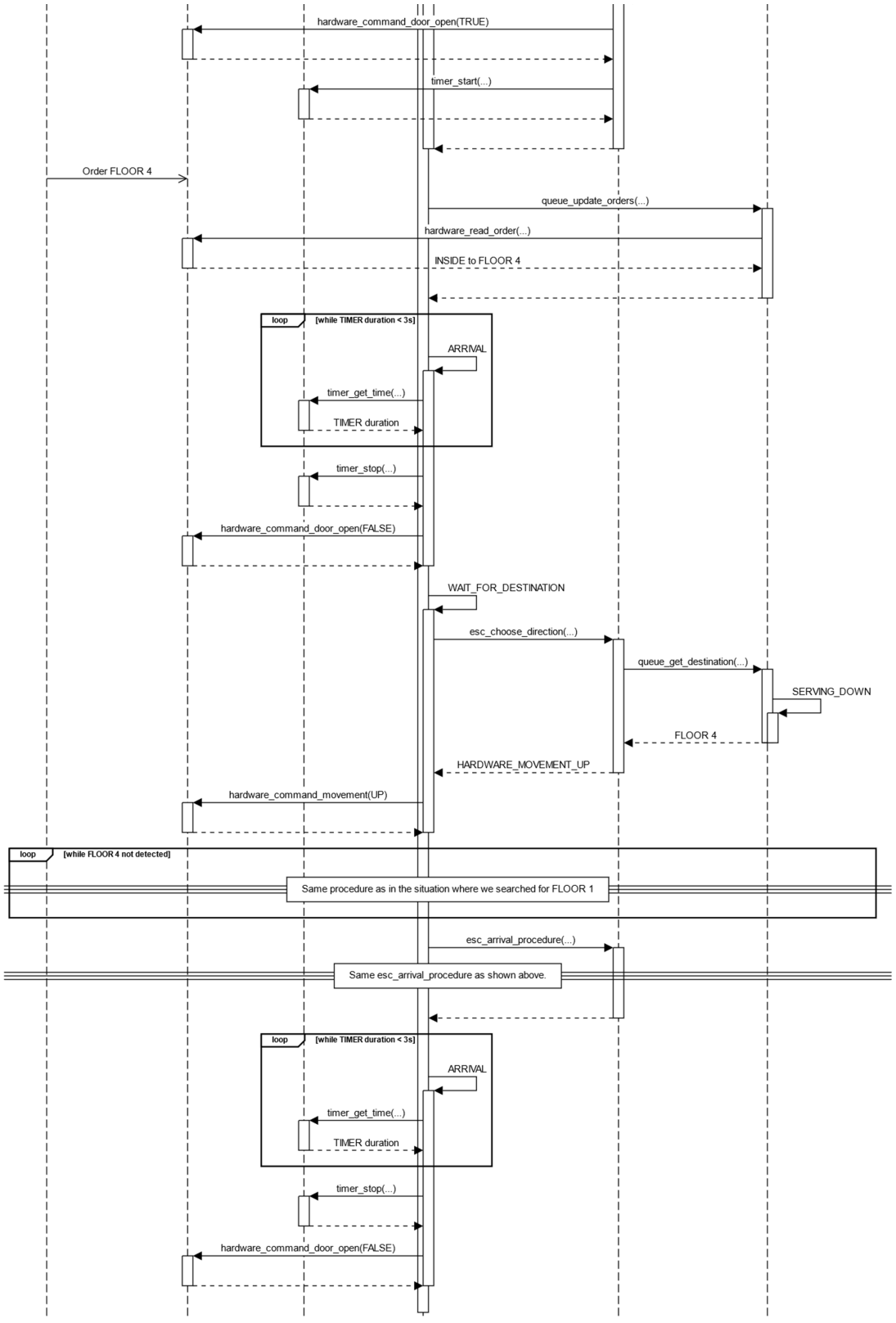
3.3 Kommunikasjon mellom systemets moduler

For å illustrere kommunikasjonen mellom enheter, ble en heissekvens gitt av oppgaven. Denne er gjengitt i tabell 1. Et tilhørende sekvensdiagram vises i figur 4. Sekvensdiagrammet viser bare det som er avgjørende for hvert enkelt steg i sekvensen. Det vil for eksempel være nødvendig å sjekke for nye ordre i hver klokkesyklus, men det er blitt unnlatt i diagrammet utenom når det er viktig for sekvensen. Gjentagende sekvenser blir bare vist en gang i detalj. For å enklere kunne sammenlikne sekvensdiagrammet med implementasjonen, ble det i etterkant valgt å benytte reelle funksjonsnavn i diagrammet.

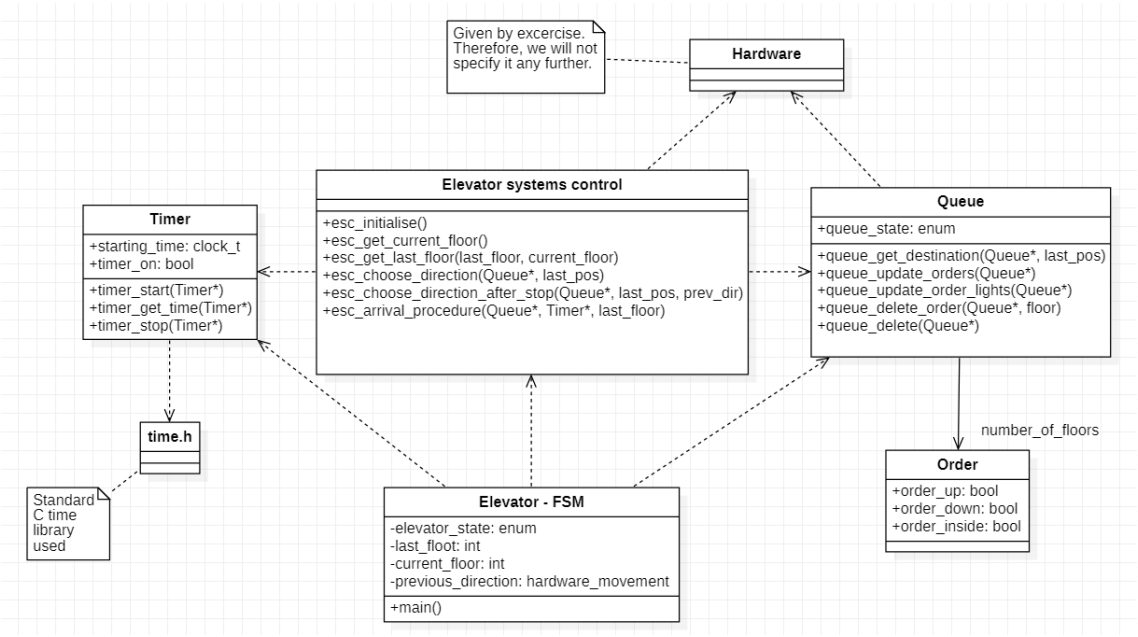
Tabell 1: Sekvens gitt av oppgaven

1. Heisen står stille i 2. etasje med døren lukket
2. En person bestiller heisen fra 1. etasje
3. Når heisen ankommer går personen inn i heisen og bestiller 4. etasje
4. Heisen kommer til 4. etasje og, personen går av
5. Etter 3 sekunder lukker dørene til heisen seg


User
Hardware
Timer
Elevator - FSM
Elevator Systems Control
Queue
Order UP from FLOOR 1
queue_update_orders(...)
hardware_read_order(...)
UP from FLOOR 1
queue_update_order_lights(...)
hardware_command_order_light(...)
The light sequence above and other trivial sequences will only be shown once.
esc_get_current_floor()
hardware_read_floor_sensor(...)
FLOOR 2
FLOOR 2
hardware_command_floor_indicator_on(...)
WAIT_FOR_DESTINATION
esc_choose_direction(...)
queue_get_destination(...)
IDLE
FLOOR 1
HARDWARE_MOVEMENT_DOWN
hardware_command_movement(DOWN)
loop
[while FLOOR 1 not detected]
esc_get_current_floor()
hardware_read_floor_sensor
FLOOR i
FLOOR i
MOVING DOWN
queue_get_destination(...)
SERVING_DOWN
FLOOR 1
esc_arrival_procedure(...)
hardware_command_movement(STOP)
queue_delete_order(...)



Figur 4: Sekvensdiagram for tabell 1.



Figur 5: Klassediagram for heisen.

4 Moduldesign

I denne seksjonen vil modulenes oppbygning, som kommer fram i figur 1, bli presentert. Et utdypende klassediagram er vist i figur 5. Hver modul består av en tilsvarende header- og c-fil. Timer- og kø-modulen er bygget opp slik at man i hovedprogrammet kan lage timer- og kø-instanser. Det ble gjort slik da det gir en mulighet for å ha flere instanser av både timere og køer samtidig. Dette kan være nyttig for et eventuelt oppsett med flere heiser enn bare én. Hardware-modulen er gitt av oppgaven og derfor går vi ikke nøyere inn på den.

4.1 Elevator - Finite State Machine

Hovedprogrammet består av heisens tilstandsmaskin, og lagrer viktig data som bestemmer heisens oppførsel. Tilstandsmaskinen er realisert som en switch som tar inn en enum av typen Elevator-State. Valget for switch framfor for eksempel funksjonspekere falt av den enkle grunn at det gjør koden enklere å lese. Heisens FSM ble implementert inne i main filen, da den utgjør kjernen til systemet.

Et tiltak som ble innført for å sikre lesbarhet og kodekvalitet var å implementere alle tilstandstransisjoner direkte i tilstandsmaskinen, og ikke inne i Elevator Systems Control. Grunnen til dette var at det skal være mulig å følge hele programflyten fra main, uten å måtte steppe inn i funksjoner.

4.2 Elevator Systems Control - ESC

ESC er delen av systemet som utfører oppgaver som den blir tildelt av FSM-en. Dens to hovedområder er å utføre mere kompliserte hardware pollinger som FSM-en ikke gjør direkte, samt gjøre operasjoner rettet mot køen. Den finner blant annet ut av hvilken etasje heisen befinner seg i og gir riktig kjøreretning basert på køens neste destinasjon. Den tar seg også av initialiseringen av heisen og prosedyren når heisen ankommer en ønsket destinasjon.

Modulen er sterkt knyttet til FSM-en og de til sammen kan betegnes som styringssystemet. Den ble separert som egen header- og c-fil for å gjøre koden mer leselig og enklere å vedlikeholde.

4.3 Queue

Køen er implementert som en liste av "Order"-structs, en for hver etasje. En "Order"består av 3 bools som inneholder informasjon om det er OPP, NED og INSIDE bestilling i gjeldende etasje. Indeksen til listen brukes for å holde styr på hvilken etasje bestillingen kommer fra. Dette muliggjør enkel sjekking av bestillinger i en bestemt etasje, uten å måtte iterere gjennom etasjene.

Ved oppstart lages en kø-instanse der alle bestillingene er satt til null. Det er derfor til alle tider 3 bools for hver etasje i minne. Dette muligjør enkel oppdatering av ordrelys, samtidig som det ungår problemene med dynamisk allokert minne. Da queue_update_order_lights er såpass koplet til køen, ble den implementert som en del av kø-modulen og ikke i en ekstra lysmodul. Queue_get_destination er også en viktig funksjon, da den inneholder køens tilstandsmaskin, som inneholder logikken som til alle tider gir køens neste destinasjon. Det tas i bekostning å gjøre denne litt mer komplisert, for gevinsten vil være at man enkelt kan bytte kø-logikk ved å bare endre denne funksjonen.

Kø-modulen inneholder også en funksjon som poller hardwaren etter nye bestillinger og tilføyer dem til køen. I tillegg har den vanlige heis funksjoner som sletting av bestillinger og en funksjon som sletter alle bestillinger.

4.4 Timer

Timeren er en struct som lagrer starttidspunkt som antall klokkepulser siden programmet startet og en bool som forteller om timeren har startet eller ikke. Funksjonen clock() fra standard C-bibliotek time.h brukes til å sette en tidsreferanse som en clock_t variabel. Clock() brukes istedenfor time(), da time() ikke gir ønsket presisjon.

Funksjonen timer_get_time vil da ta inn en timer-instanse og returnere differansen mellom lagret starttidspunkt og clock() når funksjonen kalles, i sekunder. I tillegg er det funksjoner som starter og stopper timerene.

5 Testing

5.1 Enhetstesting

Under modulimplementasjon, ble enkle tester gjennomført for å sikre at hver modul fungerte som den skulle. Dette ble gjort gjennom relevante funksjonskall fra modulen i main. Disse tok som regel inn en hardkodet input og sendte resultatet til en variabel. GDB kunne deretter brukes for å sjekke om modulen ga forventet respons på en rekke innganger.

Kø-modulen ble testet ved å definere hardkodete variabler for køen, kø-tilstanden og etasjen i main. Det kunne da enkelt kjøres kall til, blant annet, funksjonen som kalkulerte neste destinasjon og lagre resultatet i en variabel. Dette ble testet med GDB for en rekke forskjellige bestillingskonfigurasjoner og kø-tilstander slik at modul-funksjonenes funksjonalitet kunne bekreftes.

Timer-modulen ble testet ved å sette opp en instans timer-structen i main, starte denne, og skrive til konsolen når en viss tid har gått. Dette ble kontrollert mot en stoppeklokke.

5.2 Integrasjonstest og verifikasjon

For å verifisere at heisen oppfylte heisspesifikasjonene (A), ble en test skrevet for hver spesifikasjon. Disse testene er gjengitt i tabell 2, og utgjør til sammen en integrasjonstest fordi mange av spesifikasjonene krevde at flere moduler samarbeidet. Denne testformen ble valgt for å sikre enkel og veldefinert verifisering av spesifikke kravspesifikasjoner.

For mange tester var det hensiktsmessig å systematisk utføre en rekke bestillinger. Det ble her sørget for at alle tre bestillingstyper (seksjon 4.3) ble kalt i hver etasje samt i etasjer som er over og under gjeldende etasje. En slik prosedyre betegnes i tabellen med utrykket "kjør bestillinger systematisk ".

Enkelte spesifikasjoner egnet seg ikke for kun individuelle tester, men ble kontinuerlig vurdert gjennom integrasjonstesten. Disse var spesifikasjonene Y1, R2 og R3, som henholdsvis omfattet "vanlig heisoppførsel "og at heisen ikke skulle trenge flere initialisering-runder eller kræsje gjennom kjøringen.

Det ble oppdaget flere spesifikasjonskonflikter, og disse ble taklet med utgangspunkt i Y1, altså vanlig heis-oppførsel. For eksempel, kolliderer spesifikasjonen S4 om at heisen skal stoppe momentant med spesifikasjonen O1, om at heisen ved oppstart alltid skal komme i en definert tilstand. Siden heisen i initialisering ikke skal ta bestillinger, vil den etter et momentant stopp aldri komme i en definert tilstand. Det ble derfor valgt å ignorere stopp-kall under initialisering, med opphav i at heisen er tom i denne situasjonen.

Tabell 2: Testing av kravspesifikasjoner (Appendix A). Se over for hva som legges i utrykket kjør bestillinger systematisk".

Kravspek	Test	Akseptansekrav
O1 O2 O3	Start heisen mellom to etasjer/i en etasje. Utfør kontinuerlig en rekke bestillinger. Sett en watchpoint på køen og etasjevariabelen i GDB.	Kjører til den lavere etasjen og stopper eller blir i den definerte etasjen. GDB stopper først programmet (dvs køen og etasje-variabelen oppdaterer seg) når dette har skjedd.
H1	Kjør bestillinger systematisk.	Alle bestillinger betjenes. Obstruksjonsbryteren påvirker kun tiden døren holdes åpen.
H1 H2	1) Start i 1, NED i 3, OPP i 2 rett etter. 2) Start i 4. OPP i 2, NED i 3 rett etter. 3) Prøv å fang heisen mellom to etasjer (for eks. gjentatte bestillinger i 1. og 2., se om den tar en bestilling i 3.) 4) Kjør bestillinger systematisk.	1) Heisen betjener OPP i 2 og fullfører OPP betjeningen før den tar NED i 3. 2) Heisen betjener NED i 3 og fullfører NED betjeningen før den tar OPP i 2. 3,4) Heisen bytter på å betjene oppover og nedover, alle bestillinger betjenes.
H3 H4	Kjør bestillinger systematisk. Sett GDB watchpoint på køen.	Bestillinger i en etasje tømmes når heisen ankommer den. Når hele køen er tom, står heisen i ro.
L1 L2	Kjør bestillinger systematisk.	Bestillingslys tennes når bestillingene opprettes og slukkes når de er betjent.
L3 L4 L5	Kjør heisen oppover og nedover gjennom alle etasjer.	Et etasjelys tennes når heisen passerer etasjen. Andre etasjelys er av.
L6	Trykk, hold og slipp stopp knappen når heisen er i bevegelse og i ro.	Stoppknappen lyser når den er trykket inn og slukkes når den slippes.
D1 D2	Kjør bestillinger systematisk. Noter når døren åpner seg og hvor lenge (bruk stoppeklokke).	Døren åpnes i 3 sekunder etter ankomst i etasje med bestilling (også om bestilling i samme etasje). Når alle er betjent (og det har gått 3 sekunder), er døren lukket.
D3 D4	Test alle 4 kombinasjoner av stopp og obstruksjon når heisen er i en etasje.	Heisdøren åpner seg alltid ved stopp, ellers kun om det er en ubetjent bestilling i etasjen. Heisdøren lukkes 3 sekunder etter den siste av stopp/obstruksjon er slått av.
S1 S2	Kjør bestillinger systematisk. Test kombinasjoner av stopp/obstruksjon når heisen er i bevegelse og i ro.	Døren er kun åpen når heisen står i ro i en definert etasje.
S3	Kjør bestillinger systematisk. Test kombinasjoner av stopp/obstruksjon. Skru av heisen og start den i alle etasjer.	Heisen kjører aldri over øverste etasje, eller under første.
S4 S5 S6 S7	Kjør bestillinger systematisk. Trykk og hold stoppknappen når heisen er i ro og i bevegelse, og fortsett å gjøre bestillinger.	Heisen stopper momentant når stoppknappen trykkes. Bestillinger gjort før og under stoppen betjenes ikke. Heisen står i ro når stoppknappen slippes, helt til den får en bestilling, som den går til direkte.
R1	Kjør bestillinger systematisk. Bruk obstruksjonsbryteren når heisen er i bevegelse og i ro.	Heisen opererer normalt. Eneste effekt obstruksjonsbryteren har er lenger dør-åpnings-tid ved ankomst i etasje/åpen dør (se D3).
R2	Kjører Valgrind leak check.	Ingen minnelekasje detekteres.

6 Diskusjon

Et sentralt fokus i arbeidet har vært å ha så lett koplet interaksjon mellom modulene som mulig. Av dette hensyn ble heis-styring-systemet (Elevator Systems Control) implementert slik at det ville fungere helt uavhengig av hvilke kø-implementasjon som ble valgt. Dette gjorde det overordnette arkitekturdesignet mye enklere, ettersom styringssystemet kunne forholde seg til kun neste destinasjon gitt som et heltall, mens køsystemet kunne fokusere på å generere dette.

Vår første kø-implementasjon fokusere på når bestillinger ble gjort ovenfor hvor de er i forhold til heisen. Det visste seg i løpet av testingen at dette lot oss fange heisen mellom to etasjer, slik at heisen ikke oppfylte kravspesifikasjon H1 (seksjon A). Grunnet den adskilte implementasjonen av kø-systemet var det veldig enkelt å rette opp problemet, da vi kun måtte endre kø-logikken og ikke resten av koden.

Et annet eksempel hvor vi fokuserte på lett koplet modularisering var timeren, som vi valgte å implementere som en egen modul. Det kunne alternativt ha blitt løst på en enklere og like effektiv måte, ved å for eksempel definere en tidsvariabel. En modularisert løsning er derimot nyttig dersom timeren i fremtiden vil bli brukt i annen implementasjon, for eksempel for å holde orden på hvor lenge heisen har vært i en nødstopp.

Heissystemt vurderes til å ha god skalerbarhet. Først og fremst kan heissystemet enkelt utvides til flere etasjer ved å oppdatere en variabel for antall etasjer. Videre kan underetasjer implementeres ved å offsette systemet slik at etasjen med indeks 0 i koden tilsvarer til den laveste etasjen i heisen. I tillegg åpner heis-styring-systemets' uavhengighet fra køen for enkel skalering til mer avanserte kø-systemer, noe som er diskutert videre nedenfor. Vi vurderte tidlig i prosessen en kø som tok i bruk lenkede lister. Her ville man derimot måttet implementert og kalt en omfattende sorteringsalgoritme i hver programsyklus, som blir avansert ved mange etasjer.

For å sikre robusthet ble det bestemt å polle essensielle hardwarefunksjoner kun en gang i hver programsyklus. Dette ble gjort for å unngå at funksjoner som blant annet leste etasjesensorer ville ha forskjellige verdier i løpet av samme syklus. Ved en utvidelse kan robusthet forbedres ytterligere. Under testing ble det blant annet funnet at aktivering av strømforsyningen på labben fikk heisen til å registrere bestillenger eller aktivere lys.

Det største forbedringspotensialet er i middlertid overflødighet i deler av systemet. Et slikt eksempel er at vi har to separate tilstander for opp- og nedover bevegelse selv om funksjonaliteten i begge er meget lik. En enkel tilstand for bevegelse, som holdt styr på retning gjennom en variabel, ville derfor resultert i mer kompakt og leselig kode. På samme måte er det eneste skille mellom funksjonen chose_direction og funksjonen chose_direction_after_stop at tidligere heisretning har betydning. Det er unødvendig å definere en ny, nesten identisk funksjon for kun denne forskjellen.

En interessant utvidelse av kø-systemet kan være å ta sannsynlighetsmodeller i betraktning for å tilpasse heisen til spesifikke bruksområder. For eksempel, vil en heis i et kontorbygg kunne prioritere opp bestillinger på morgenen og ned bestillinger når arbeidsdagen er slutt, ettersom dette vil bidra til minket kø. For heiser med mange etasjer, kan også bestillinger prioriteres etter hvor lang tid de tar å utføre. Her vil modulariseringen av timer modulen være nyttig.

Referanser

Mathisen, Geir (2021). *Systemutvikling i henhold til V-modellen.* URL: https://docplayer.me/8512447-Systemutvikling-i-henhold-til-v-modellen.html (sjekket 19. mar. 2021).

Appendix

A Heisspesifikasjoner

Disse spesifikasjonene er hentet fra oppgaveteksten.

A.1 Oppstart

Punkt	Beskrivelse
O1	Ved oppstart skal heisen alltid komme til en definert tilstand. En definert tilstand betyr at styresystemet vet hvilken etasje heisen står i.
O2	Om heisen starter i en udefinert tilstand, skal heissystemet ignorere alle forsøk på å gjøre bestillinger, før systemet er kommet i en definert tilstand.
O3	Heissystemet skal ikke ta i betraktning urealistiske start-betingelser, som at heisen er over fjerde etasje, eller under første etasje idet systemet skrus på.

A.2 Håndtering av bestillinger

Punkt	Beskrivelse
H1	Det skal ikke være mulig å komme i en situasjon hvor en bestilling ikke blir tatt. Alle bestillinger skal betjenes selv om nye bestillinger opprettes.
H2	Heisen skal ikke betjene bestillinger fra utenfor heisrommet om heisen er i bevegelse i motsatt retning av bestillingen.
H3	Når heisen først stopper i en etasje, skal det antas at alle som venter i etasjen går på, og at alle som skal av i etasjen går av. Dermed skal alle ordre i etasjen være regnet som ekspedert.
H4	Heisen skal stå stille om den ikke har noen ubetjente bestillinger.

A.3 Bestillingslys- og etasjelys

Punkt	Beskrivelse
L1	Når en bestilling gjøres, skal lyset i bestillingsknappen lyse helt til bestillingen er utført. Dette gjelder både bestillinger inne i heisen, og bestillinger utenfor.
L2	Om en bestillingsknapp ikke har en tilhørende bestilling, skal lyset i knappen være slukket.
L3	Når heisen er i en etasje skal korrekt etasjelys være tent.
L4	Når heisen er i bevegelse mellom to etasjer, skal etasjelyset til etasjen heisen sist var i være tent.
L5	Kun ett etasjelys skal være tent av gangen.
L6	Stoppknappen skal lyse så lenge denne er trykket inne. Den skal slukkes straks knappen slippes.

A.4 Heis-dør

Punkt	Beskrivelse
D1	Når heisen ankommer en etasje det er gjort bestilling til, skal døren åpnes i 3 sekunder, for deretter å lukkes.
D2	Heisen skal være lukket når den ikke har ubetjente bestillinger.
D3	Hvis stoppknappen trykkes mens heisen er i en etasje, skal døren åpne seg. Døren skal forholde seg åpen så lenge stoppknappen er aktivert, og ytterligere 3 sekunder etter at stoppknappen er sluppet. Deretter skal døren lukke seg.
D4	Om obstruksjonsbryteren er aktivert mens døren først er åpen, skal den forbli åpen så lenge bryteren er aktiv. Når obstruksjonssignalet går lavt, skal døren lukke seg etter 3 sekunder.

A.5 Sikkerhet

Punkt	Beskrivelse
S1	Heisen skal alltid stå stille når døren er åpen.
S2	Heisdøren skal aldri åpne seg utenfor en etasje.
S3	Heisen skal aldri kjøre utenfor området definert av første til fjerde etasje.
S4	Om stoppknappen trykkes, skal heisen stoppe momentant.
S5	Om stoppknappen trykkes, skal alle heisens ubetjente bestillinger slettes.
S6	Så lenge stoppknappen holdes inne, skal heisen ignorere alle forsøk på å gjøre bestillinger.
S7	Etter at stoppknappen er blitt sluppet, skal heisen stå i ro til den får nye bestillinger.

A.6 Robusthet

Punkt	Beskrivelse
R1	Obstruksjonsbryteren skal ikke påvirke systemet når døren ikke er åpen.
R2	Det skal ikke være nødvendig å starte programmet på nytt som følge av eksempelvis udefinert oppførsel som for eksempel at programmet krasjer, eller minnelekkasje.
R3	Etter at heisen først er kommet i en definert tilstand ved oppstart, skal ikke heisen trenge flere kalibreringsrunder for å vite hvor den er.

A.7 Tillegg

Punkt	Beskrivelse
Y1	Oppførsel som ikke er "vanlig heisoppførsel" kan gi trekk på FAT-testen. Når det er sagt så er det bare å bruke sunn fornuft og eventuelt spør vitass eller foreleser om noe er uklart.