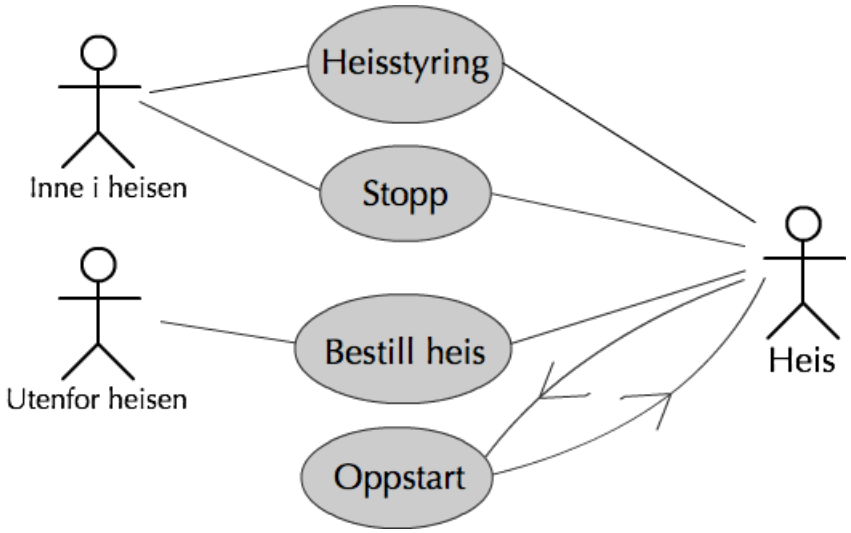
1 Usecases



Bestillingsknapp: Knappene opp og ned som befinner seg i hver etasje.
Etasjeknapp: Knappene inne i heisen som bestemmer hvor heisen skal gå

Oppstart

Precondition: Ingen
Trigger: Program startes
Suksesscenario:

1. Sjekk etasjesensor
2. Kjør heis opp til den kommer til en etasje
3. Stopp heisen

Utvidelser:

1a. Hvis heis er i etasje, hopp til punkt 3.

Suksessgaranti: Heis i ro i kjent etasje
Minimal garanti: samme som suksessgaranti

Stopp

Precondition: Ingen
Trigger: Stopp-knapp trykkes eller obstruksjon aktiveres når heisen er i bevegelse mellom etasjer
Suksesscenario:

1. Heisen stoppes
2. Alle bestillinger slettes fra bestillingsliste
3. For å komme ut av stopp-modus må obstruksjon være av og etasjeknapp må trykkes

Suksessgaranti:

1. Heisen står stille
2. bestillingene slettes

Minimal garanti: Samme som suksessgaranti
Scenario:

Bestill Heis

Precondition: Stoppknappen har ikke blitt trykket, heis ikke i oppstartsmodus
Trigger: Bestillingsknapp trykkes inn
Suksesscenario:

1. Bestilling i etasje blir lagt til i ordreliste
2. Heisen fortsetter med andre ordre. En ordre vil slettes etter at den blir ekspedert
3. Hvis heisen ikke har flere ordre videre i retningen den sist kjørte, vil den snu og ekspedere ordre i motsatt retning.
4. Når heisen kommer til riktig etasje, stopp heisen
5. Fjern etasjen fra ordrelisten, åpne døren
6. Døren vil holdes åpen så lenge det er obstruksjon.
7. Når obstruksjonen fjernes vil døren holdes åpen i 3 sekunder.

Utvidelser:

2a. Hvis heisen kommer til riktig etasje i samme retning som bestillingen, hopp til punkt 4.

6a. Hvis det ikke er obstruksjon, hold døren oppe i 3 sekunder.

Suksessgaranti:

1. Heisen står i etasjen
2. Dør åpen i minimum 3 sekunder, deretter lukkes den hvis det ikke er obstruksjon
3. Heisen vil fortsette i samme retning som bestillingen antyder

Minimal garanti:

1. Ingen

Bestilling fra etasjeknapp

Precondition: Heis ikke i oppstartsmodus
Trigger: Etasjeknapp trykket
Suksesscenario:

1. Bestilling i etasje blir lagt til i ordreliste
2. Heisen med samme retning som før, og stopper for å ekspedere ordre.
3. Hvis heisen ikke har flere ordre videre i retningen den sist kjørte, vil den snu og ekspedere ordre i motsatt retning.
4. Når heisen kommer til riktig etasje, stopp heisen
5. Fjern etasjen fra ordrelisten, åpne døren
6. Døren vil holdes åpen så lenge det er obstruksjon.
7. Når obstruksjonen fjernes vil døren holdes åpen i 3 sekunder.

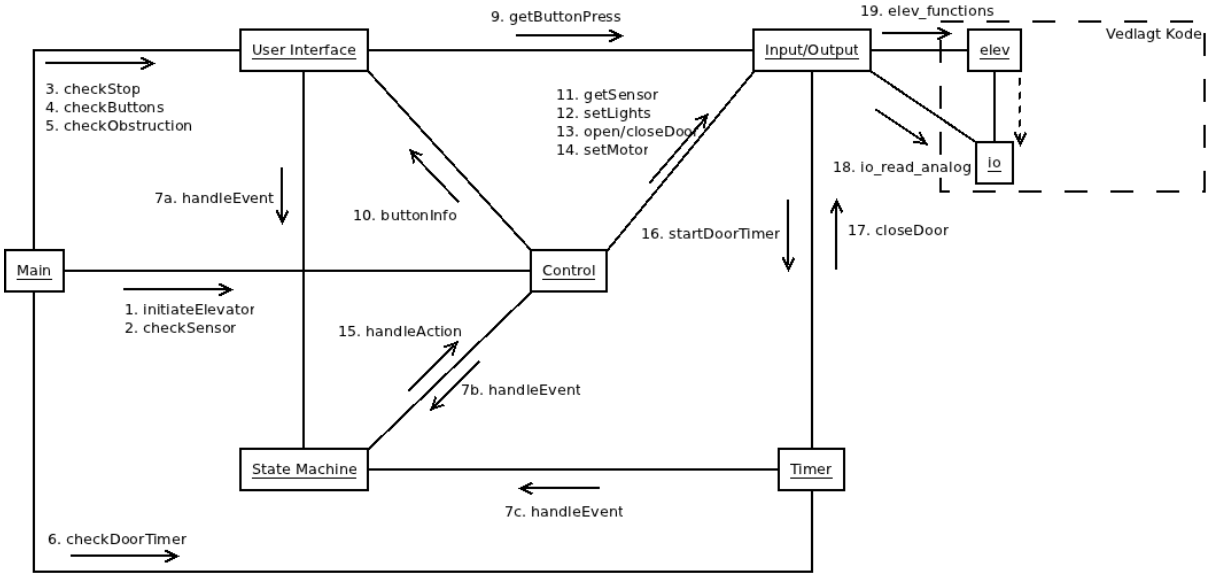
Utvidelser:

2a. Hvis heisen kommer til bestillt etasje, hopp til punkt 4.

Suksessgaranti:

1. Heis til etasje
2. Dør åpen i minimum 3 sekunder, deretter lukkes den hvis det ikke er obstruksjon

Minimal garanti: Samme som suksessgaranti



Figur 1: Overordnet systemarkitektur

2 Systemarkitektur

1. initiateElevator kalles hver gang programvaren starter opp, for å definere heisen.
2. checkSensor sjekker sensoren.
3. checkStop sjekker stoppknappen.
4. checkButtons sjekker for bestillinger.
5. checkObstruction sjekker for obstruksjon.
6. checkDoorTimer sjekker dørtimeren.
7. handleEvent kalles i tre klasser, når forskjellige hendelser oppstår.
   (a) den kalles i ui-klassen når en knapp blir trykket inn, slik at bestillingen blir registrert i ordrelisten, eventuelt håndterer en nødstopp. Den sjekker også om en obstruksjon blir fjernet, slik at heisen eventuelt kan begynne å kjøre.
   (b) i control-klassen genereres det en hendelse når heisen ankommer en etasje, slik at heisen kan avgjøre om den skal stoppe eller ikke. Den kalles også når en ny ordre kommer inn.
   (c) Den siste klassen som kaller handleEvent-funksjonen er timer-klassen. Den genererer en hendelse når døren lukkes.
8. handleAction kalles hver gang tilstandsmaskinen får et kall på handleEvent, og det er spesifisert en handling for denne nåtilstanden og hendelsen.
   (a) handleEmergencyStop kalles når heisen enten får stoppsignal gjennom stoppknappen, eller at den får inn obstruksjon mellom etasjene.
   (b) handleDestination setter en ny retning for heisen, og starter motoren
   (c) handleDestinationFromIdle gjør det samme som 8b, men denne sjekker først om bestillingen kommer fra nåværende etasje. Dette er nødvendig siden heisen står med dørene lukket, og må følgelig åpne de dersom den har en ordre i nåværende etasje, før den kjører videre.
   (d) handleDestinationFromEM starter motoren fra nødstopptilstanden. Denne vil, i tillegg til det samme som 8c, også slukke stopplyset.

(e) handleStop sørger for at heisen stoppper og åpner døra når den har en ordre i en etasje. Den fjerner også de aktuelle ordrene i etasjen.

(f) handleNewOrder sørger for å legge inn alle bestillinger som gjøres, og setter respektive lys.

9. getButtonPress sjekker IO-klassen, og henter informasjon om hvilken knapp som ble trykket inn sist.

10. buttonInfo henter informasjon om hvilken bestillingsknapp som ble trykket inn sist, for å legge inn rett ordre.

11. getSensor henter informasjon om hvilken etasje heisen er i.

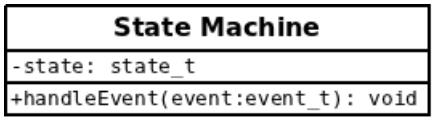
12. setLights er et samlebegrep for alle funksjoner som setter lys

13. openDoor og closeDoor håndterer døra.

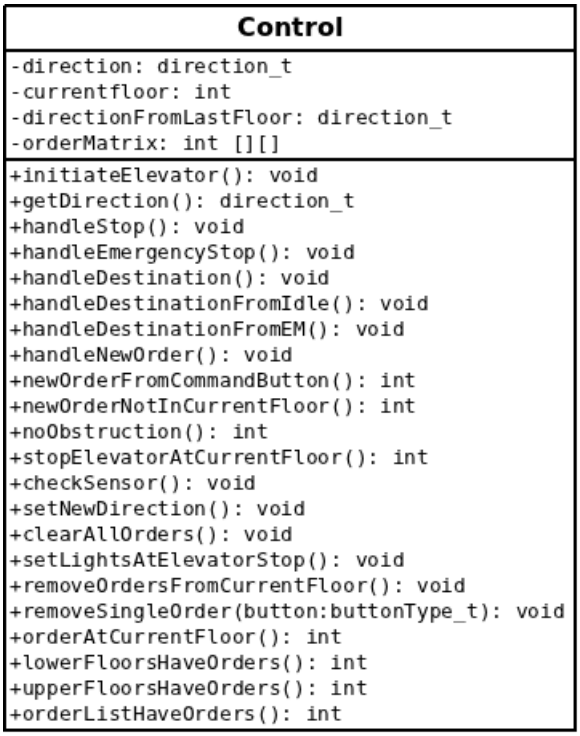
14. setMotor håndterer motoren.

15. startDoorTimer kalles når døra åpnes for å telle ned tre sekunder.

16. closeDoor kalles fra timer-klassen når timeren har telt ned.



Figur 2: Klassediagram for tilstandsmaskinen



Figur 3: Klassediagram for kontrollklassen

3 Klassediagrammer

Heisstyringa er delt opp i klasser. Måten vi har implementert dette på er at alle variabler er deklarert static for å holde de internt i en klasse/fil, og innføre set- og getfunksjoner der det trengs. Grunnen til denne oppdelingen er selvfølgelig å få en logisk oppdeling, hvor hver klasse har et veldefinert ansvarsområde. For hver klasse bør det gå greit fram hva de forskjellige variablene og funksjonene gjør, men det er gitt utdypende forklaring ved de mest sentrale.

3.1 Tilstandsmaskinen

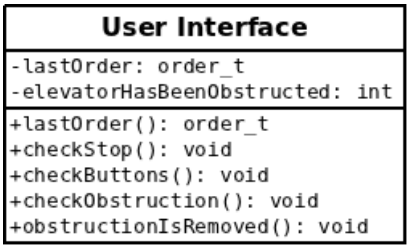
Tilstandsmaskinen holder orden på hvilken tilstand heisen står i.

handleEvent
: Bestemmer, sammen med tilstandsvariablen, nestetilstand og handling for heisen når den genererer en hendelse.

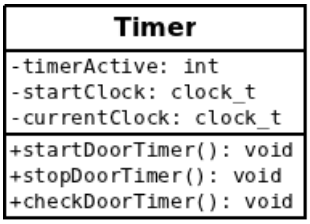
3.2 Kontroll

Kontrollklassen inneholder alle funksjoner og variabler som håndterer logikken og den fysiske funksjonen i heisen, som å bestemme retning, at motoren skal startes, hvilke lys som skal tennes, holde orden på bestillingene, og lignende.

Handlinger
: De forskjellige handlingene (handle$\cdots$) kalles av tilstandsmaskinen, og beskriver den overordnene



Figur 4: Klassediagram for UI-klassen



Figur 5: Klassediagram for timeren

oppførselen til heisen. Grunnen til tre forskjellige handleDestination-funksjoner er at man i nødstopp- og idletilstand trenger litt annerledes rutiner for å bestemme neste mål for heisen.

Sperre? Vakt?
newOrderFromCommandButton, newOrderNotInCurrentFloor, noObstruction og stopElevatorAtCurrentFloor er ?sperrer? tilstandsmaskinen bruker for å avgjøre om den skal utføre handlingene.

Checksensor
En lyttefunksjon som står og går i main-funksjonen for å avgjøre hvilken etasje heisen er i, og gi en handling når den ankommer en etasje.

Heisstyring
setNewDirection, clearAllOrders, setLightsAtElevatorStop, removeOrdersFromCurrentFloor og removeSingleOrder er funksjoner som påvirker retning, ordre og lys for heisen.

Sammeligning
De fire nederste funksjonene er hjelpefunksjoner som bestemmes av ordrene heisen har. Brukes for å bedre lesbarhet og for å unngå for mye nøsting.

3.3 User Interface

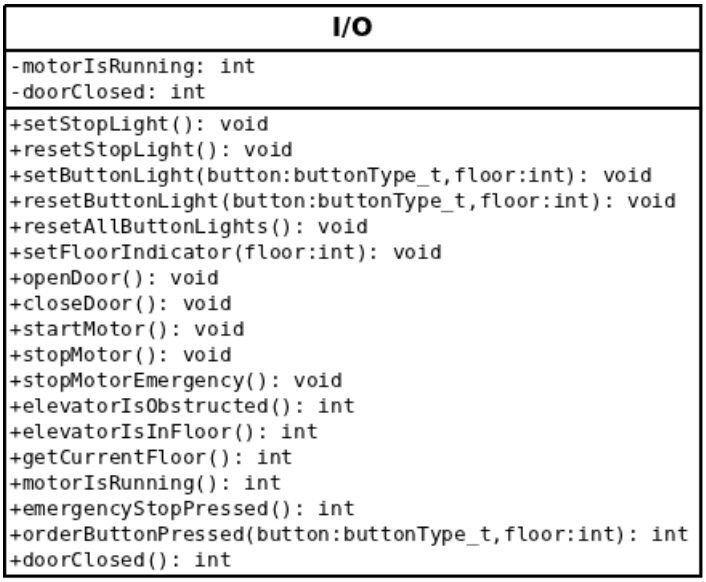
Denne klassen sørger for å håndtere input fra brukeren, som bestillingsknapper, stoppknapp og obstruksjonsføleren i døra. Den sørger for å sende bestillinger til kontroll.

Sjekkfunksjoner
checkButtons, checkStop og checkObstruction løper i main-funksjonen for å for å håndtere input fra brukeren. De genererer også hendelser for tilstandsmaskinen.

LastOrder
Denne kalles av kontroll-klassen for å returnere siste ordreknapp som ble trykket inn og legge bestilling i ordrelisten.

3.4 Timer

Timeren er en enkel klasse for å telle ned tre sekunder hver gang døren åpnes i en etasje. Den har to variabler fra time-biblioteket for å telle opp til 3 sekunder, og et heltall som holder orden på om den er aktiv eller iikke.



Figur 6: Klassediagram for I/O-klassen

checkDoorTimer
Denne står også og løper i main-funksjonen, og sjekker om nedtellingen er ferdig eller ikke.

3.5 I/O

I/O klassen er en egenlagt driver for å styre heisen. Dette er vår mest maskinnære klasse, som inneholder funksjoner for alle hardware-relaterte operasjoner som å skru av og på lys, gi signal fra sensoren og sette motorhastigheten. Disse funksjonene implementeres ved hjelp av medfølgende funksjoner i elev.c og io.c

Lysfunksjoner
Det er funksjoner for å skru lys av og på. Bestillingslampene kan skrus av en og en, eller alle på en gang som for en nødstopp. Etasjelyset har ingen resetfunksjon, da kun et skal tennes av gangen, og derfor kan det ordnes i setfunksjonen.

Heisaksjoner
Funksjoner for dør- og motorstyring. Det er egen motorstopp for nødsituasjoner, hvorfor?

Informasjon om Heis
Funksjoner som registrerer fysisk informasjon om heisen

Knapperegistreringer orderButtonPressed blir kalt fra kontroll hver gang man trykker en knapp, og legger etasje og knappetype inn i diagrammet.

vakt? sikring?
doorClosed er vakt? sikring? som brukes av tilstandsmaskinen.