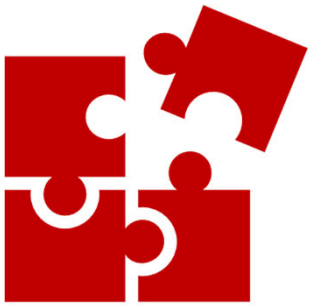
Sist oppdatert 12.9.2023

TTK 2 – Design og analyse for funksjonell sikkerhet 2023

Seminar 2 – 13.9.2023

MARY ANN LUNDTEIGEN
TEKNISK KYBERNETIKK NTNU



2. STPA analyse

Målsetning: Å belyse

I komplekse systemer der mye funksjonalitet er i programvare er tap av sikkerhet ofte et «**kontrollproblem***» og ikke nødvendigvis en konsekvens av fysisk svikt.

Systems theoretic process analysis (**STPA**) påstås å være en god metode for å avdekke slike kontrollproblemer og stille nye og bedre krav til styresystemet.

*Feil, avvik eller rett og slett uforutsett effekt av styringsalgoritmer og tilhørende kommunikasjon m sensorer og aktivert utstyr

Litteratur

Pensum:

- Slides

Støttelitteratur:

- Leveson, N. og Thomas, J.P. STPA handbook (2018)
- https://psas.scripts.mit.edu/home/get_file.php?name=STPA_handbook.pdf
- Artikkel: https://saemobilus.sae.org/content/2015-01-0274/ (Tilgjengelig via «preview» med NTNU nettverk & VPN)

SAE INTERNATIONAL

An Integrated Approach to Requirements Development and Hazard Analysis

2015-01-0274
Published 04/14/2015

John Thomas, John Sgueglia, Dajiang Suo, and Nancy Leveson
Massachusetts Institute of Technology

Mark Vernacchia and Padma Sundaram
General Motors Company

CITATION: Thomas, J., Sgueglia, J., Suo, D., Leveson, N. et al., "An Integrated Approach to Requirements Development and Hazard Analysis," SAE Technical Paper 2015-01-0274, 2015, doi:10.4271/2015-01-0274.

Abstract

The introduction of new safety critical features using software-intensive systems presents a growing challenge to hazard analysis and requirements development. These systems are rich in feature content and can interact with other vehicle systems in complex ways, making the early development of proper requirements critical. Catching potential problems as early as possible is essential because the cost increases exponentially the longer problems remain undetected. However, in practice these problems are often subtle and can remain undetected until integration, testing, production, or even later, when the cost of fixing them is the highest.

In this paper, a new technique is demonstrated to perform a hazard analysis in parallel with system and requirements development. The proposed model-based technique begins during early development when design uncertainty is highest and is refined iteratively as development progresses to drive the requirements and necessary design features. The technique is evaluated by applying it to a realistic but generic Shift-By-Wire design concept in two iterations with varying levels of detail. In addition, as the requirements and design evolve and change over time, the changes can be immediately analyzed for new hazards without repeating the entire analysis. The approach is also applicable even before requirements are developed, providing feedback when some of the most important decisions are being made instead of waiting for a finished design or model to begin an analysis. In this way, potential issues can be identified immediately and more efficiently, thereby reducing the need for future rework.

Introduction

Modern automobiles are incorporating more and more advanced software-controlled safety features such as lane keeping assist, automatic emergency braking, adaptive cruise control, and a growing number of automated or semi-automated systems. Increasingly complex behaviors are now being incorporated into safety-critical software and there are very few physical constraints that limit the complexity of the software elements. While this has allowed many innovative features that were not possible in the past, the additional complexity has made it much more challenging to develop these systems and perform adequate safety analyses. Software-related issues are frequently being discovered late in development, testing, and even during production when the problems are the most expensive to fix and when the range of practical solutions is the most limited.

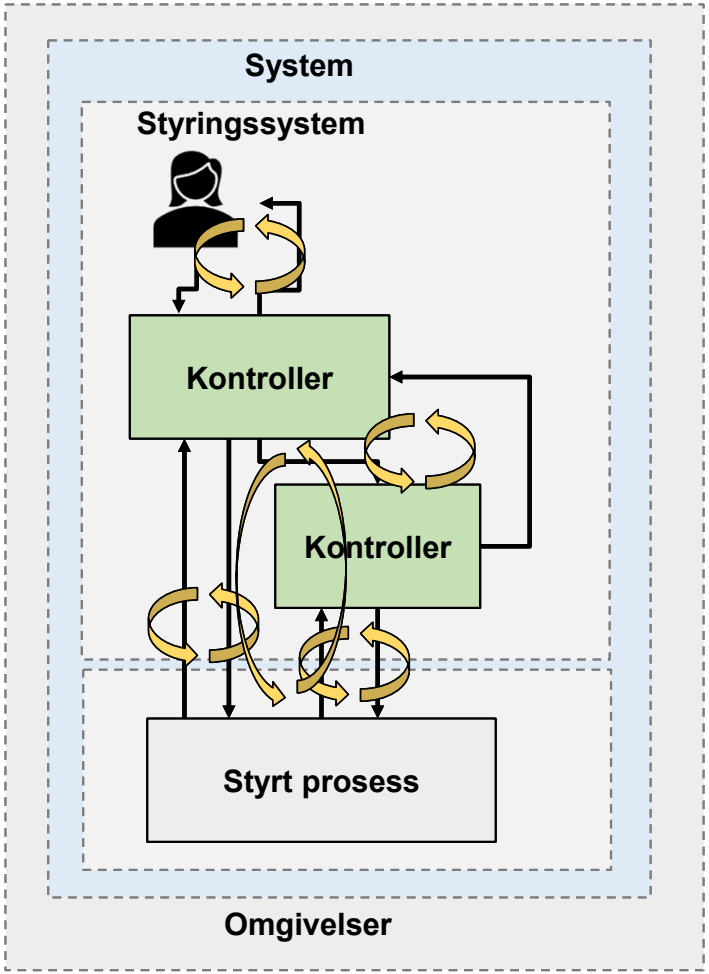
The software in modern vehicles is not only complex but also increasing in size exponentially, making the need for efficient analysis techniques increasingly urgent. One manufacturer reports two to three million lines of software code for average 2005 models [1], six million lines of code for 2007-2008 models [1], and sixteen million lines of code in 2013 [2]. Meanwhile, NHTSA data shows that more and more software-related issues are being discovered too late. About 382,000 U.S. vehicles were indicated in software-related recalls in the year 2000, compared to 13 million in 2013 and 48 million a year later [3]. In fact, some have claimed that up to 50% of car warranty costs are related to the electronics and their embedded software [4].

The traditional techniques used for analysis have not kept pace with the increased complexity of modern software-intensive systems. Although most analysis techniques consider individual failures (e.g. Fault Tree Analysis, Failure Modes and Effects Analysis, etc.), many other safety-critical decisions and assumptions are made during the development process and can easily be overlooked in a component failure-based analysis. Judgments about what behavioral requirements are needed, what interactions could potentially be dangerous, what information and feedback the software decisions should be based on, how to coordinate multiple distributed controllers, and other issues must be determined during development. While these issues are typically addressed using engineering best-judgment, they are often resolved in a compartmentalized manner without a complete understanding of the system-level impact of the proposed solution. As a result, unintended interactions are often introduced and may be overlooked by standard failure-based analyses.

STPA i korte trekk

- **Kvalitativ** sikkerhetsanalyse for komplekse (sammensatte) systemer med ***emergente*** egenskaper

- Tar som utgangspunkt at **tap av sikkerhet** kan skyldes et **styringsproblem**:
  - Fokus på å avdekke hvordan kommandoer fra kontrollere/logikk kan medføre farlige situasjoner og feil i samvirke med andre.
  - Analyse av årsaker
  - (utfra dette) formulere nye/endrede krav til styringsalgoritmer og teknisk løsning (sensorer,…)



Emergente egenskaper

Emergens (fra latin *emergere*, 'dukke opp', «frambryting») er et vitenskapsteorisk begrep for prosesser der et komplekst mønster dannes ***ut fra samspill*** mellom enkle strukturer eller atferder (Wiki)

"The whole is greater than its parts"

Helheten er større en summen av delene

Helheten er noe annet enn summen av delene

Emergent egenskap

Oppførsel som ikke lar seg forklare av å bare forstå enkeltkomponentenes oppførsel

Aristoteles, 384-322 fvt

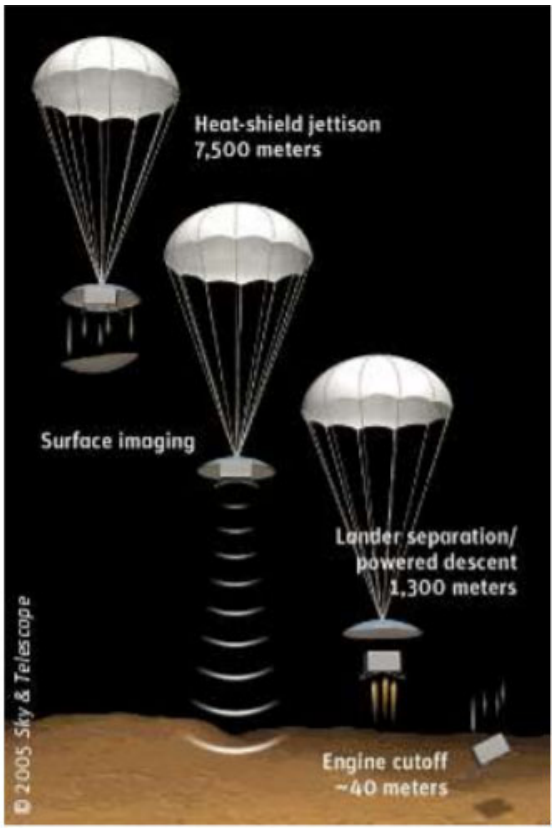
Emergente egenskaper

Kan være:

- **Ønskede (spesifiserte)**
- Uønskede
- Uventede
- Uforutsigbare

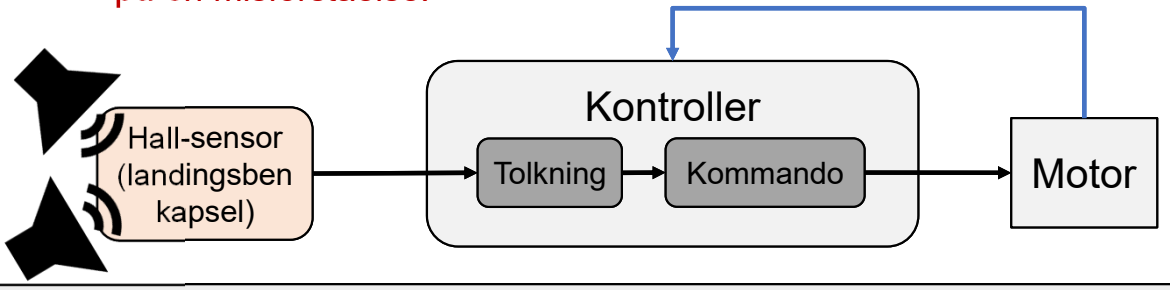
Tesla autopilot

Eksempel: Mars Polar lander (1999)



https://en.wikipedia.org/wiki/Mars_Polar_Lander

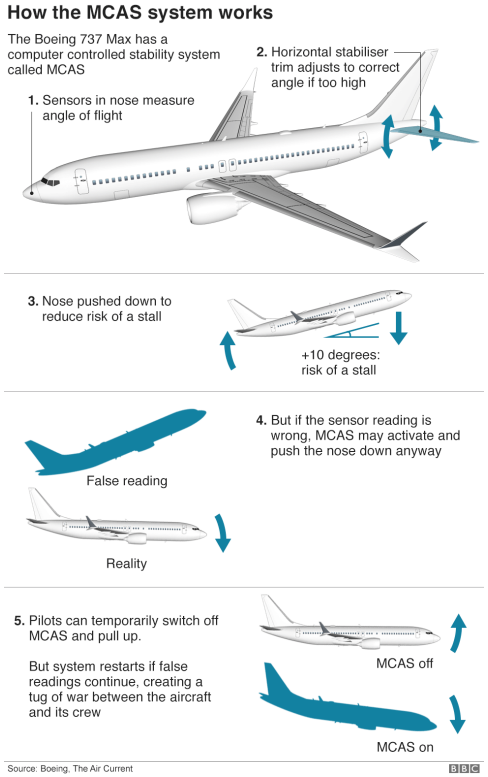
- **Ønsket oppførsel**: Fartøy skulle slå av motoren *etter* landing
- **Hva skjedde:**
  - **Støy** generert av sensor på landingsben ble feiltolket (av kontroller) som om fartøyet var allerede landet og motoren ble avslått 40 meter over bakken.
  - Dette var en type feil på sensor som man ikke avdekket og derofr ikke tok høyde for.
- **Grunn til uønsket oppførsel:** Kontrollsystemet tok beslutning basert på en misforståelse.



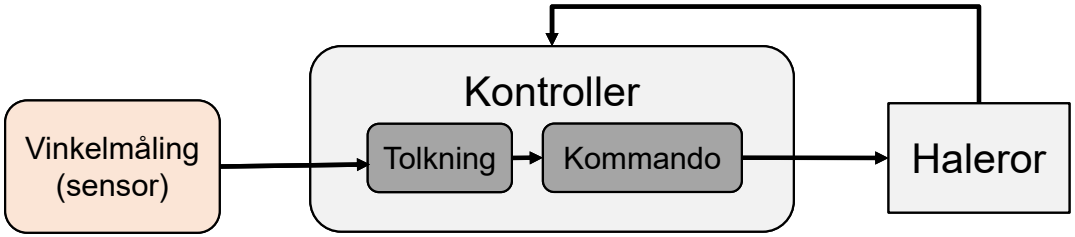
Modell (for tolkning): Sensor indikerer 0 meter = på bakken
Algoritme (kommando): Hvis på bakken → Slå av motor

Hvordan kunne kontrollsystemet ha fanget opp misforståelsen og håndtert situasjonen?

Eksempel: Boeing 737 max (2019, 2020)



- **Ønsket oppførsel:** Ved stall (vinkel i forhold til luftstrøm der løfteevne mistes), skal flyet automatisk styre fly-nesen ned slik at løftet gjenopprettes.
- **Hva skjedde:**
  - Feil i sensor som måler vinkel bidro til at stabilitetsstyringssystemet (MCAS) tok over og presset fly-nesen ned, *uten* at flyet var i stall-situasjon.
  - Til tross for flyverens forsøk på å overstyre, tok MCAS tok hele tiden tilbake kontrollen og fortsatte å presse fly-nesen ned.
- **Grunn til uønsket oppførsel:**
  - **Kontrollsystemet oppdaget ikke at sensorene ga feil informasjon**
  - **Kontrollsystemet ga ikke flyverne tilstrekkelig mulighet til å deaktivere MCAS når flyverne vurderer det som nødvendig.**

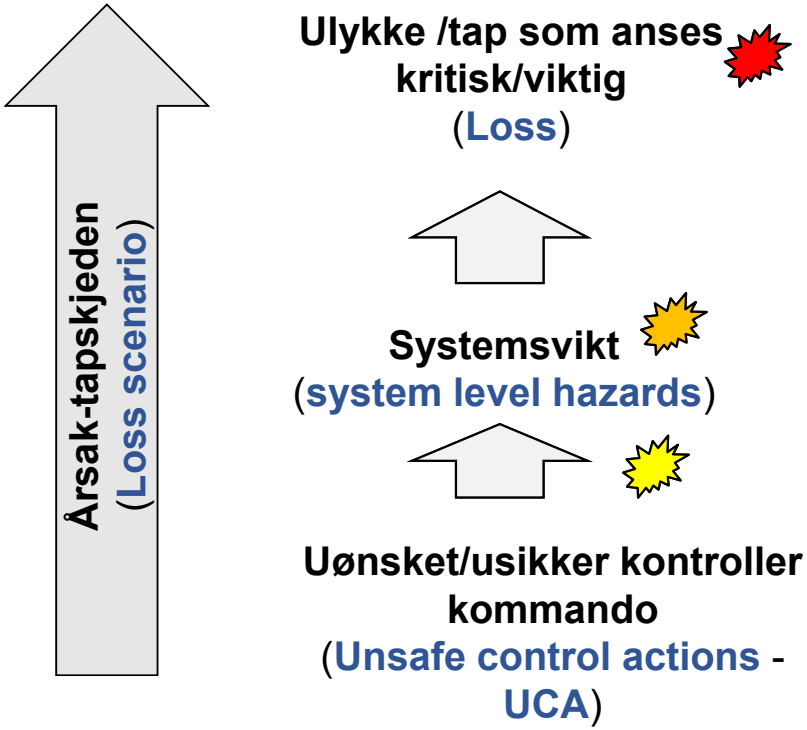


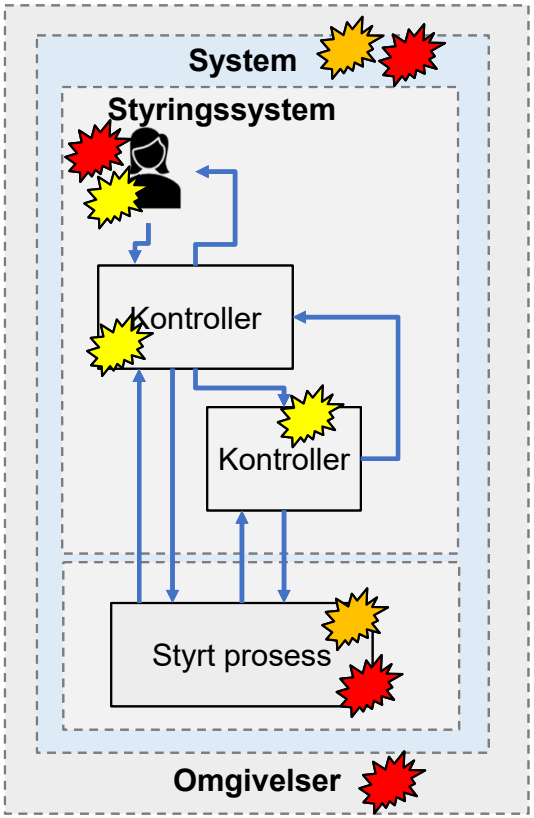
Modell: Målt verdi (sensor) gir vinkel i forhold til luftstrøm
Kommando: Hvis vinkel > Verdi, aktiver haleror for å senke nesen

Hvordan kunne kontrollsystemet oppdaget feilinformasjonen? Hvordan kunne MCAS systemet tillatt at flyverne fikk ta over i en kritisk situasjon?

https://en.wikipedia.org/wiki/Boeing_737_MAX_groundings

Sentrale begrep i STPA





Sentrale begrep i STPA

Eksempler

- **Loss / tap:**
  - Konsekvens i forbindelse med systemet som interessenter (eier, samfunn, andre) vil unngå
  - Kan angå mennesker, miljø eller finansielle tap

  Fullstendig tap av Mars Polar lander

- **System level hazards/ systemsvikt:**
  - Årsaker knyttet til systemets oppførsel eler egenskaper som leder til tapet

  Fartøyet ikke i stand til å lande kontrollert

- **Unsafe control action /uønsket kontrollkommando:**
  - Årsaker til systemsvikt som kan tilegnes aksjoner gjort av kontrollere

  Kontroller slår av motor etter feilaktig tolkning av signal fra landingsben

Sentrale begrep i STPA

Eksempler

- **System constraints:/systembegrensninger:**
  - Måter å forhindre eller unngå en systemsvikt
  - Mulig for designer/operatør å kunne påvirke
  - Negasjon av hver systemsvikt

Fartøyet må opprettholde evne til å lande kontrollert

- **Controller constraints /kontroller (kommando) restriksjoner:**
  - Måter å unngå usikker kontroller aksjoner/kommandoer (UCAer)

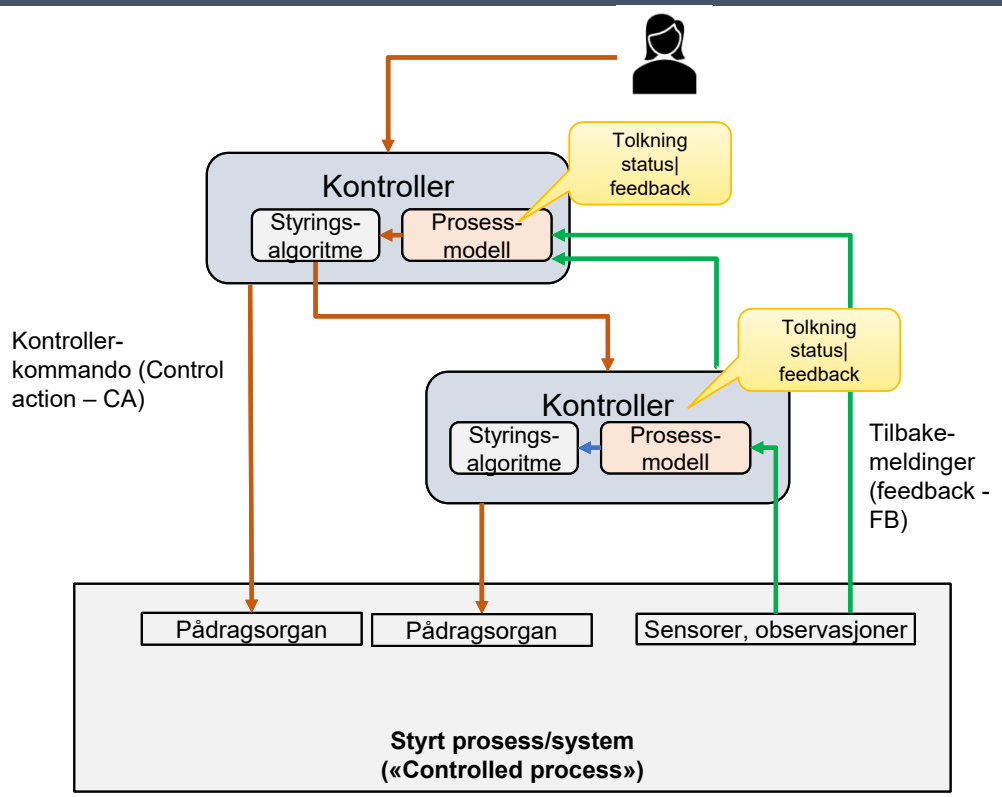
Kontroller må ikke slå av motor dersom sensorsignal er påvirket av støy fra landingsben

Sentrale begrep i STPA

Hierarkisk kontrollstruktur:

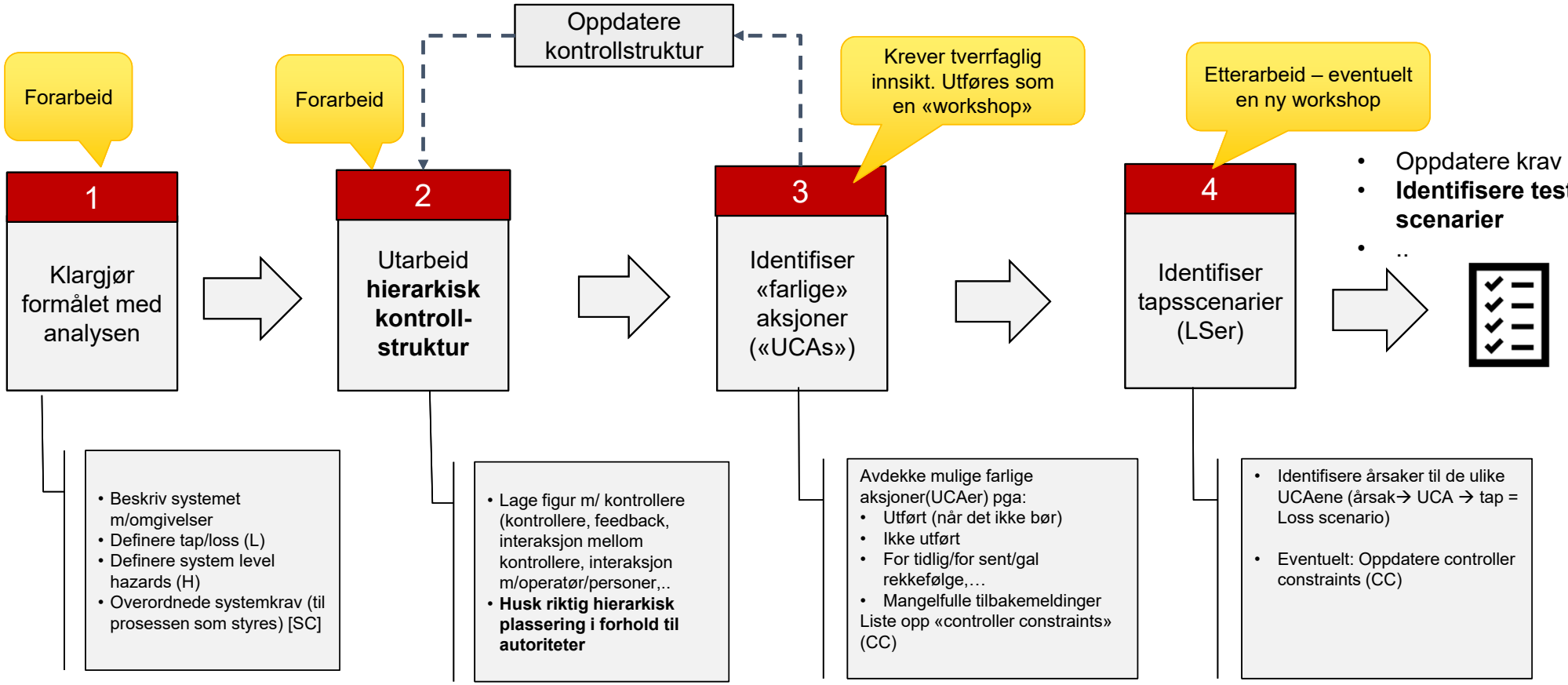
- **Kontrollstruktur:** Identifiserer kontrollere (fysiske, menneskelige), interaksjoner og tilbakemeldinger.
  - Prosessmodell
  - Styringsalgoritme
  - Kontroller kommandoer
  - Feedback

- **Hierarkisk:** Plassering vertikalt sier noe om **autoritet** i beslutningene.



Hierarkisk kontrollstruktur

STPA - steg for steg



Case-eksempel: Bil med elektronisk gir-velger

Bilde:
https://www.kongsbergautomotive.com/products-services/passenger-cars/driveline/shifters/sbw-in-knob/

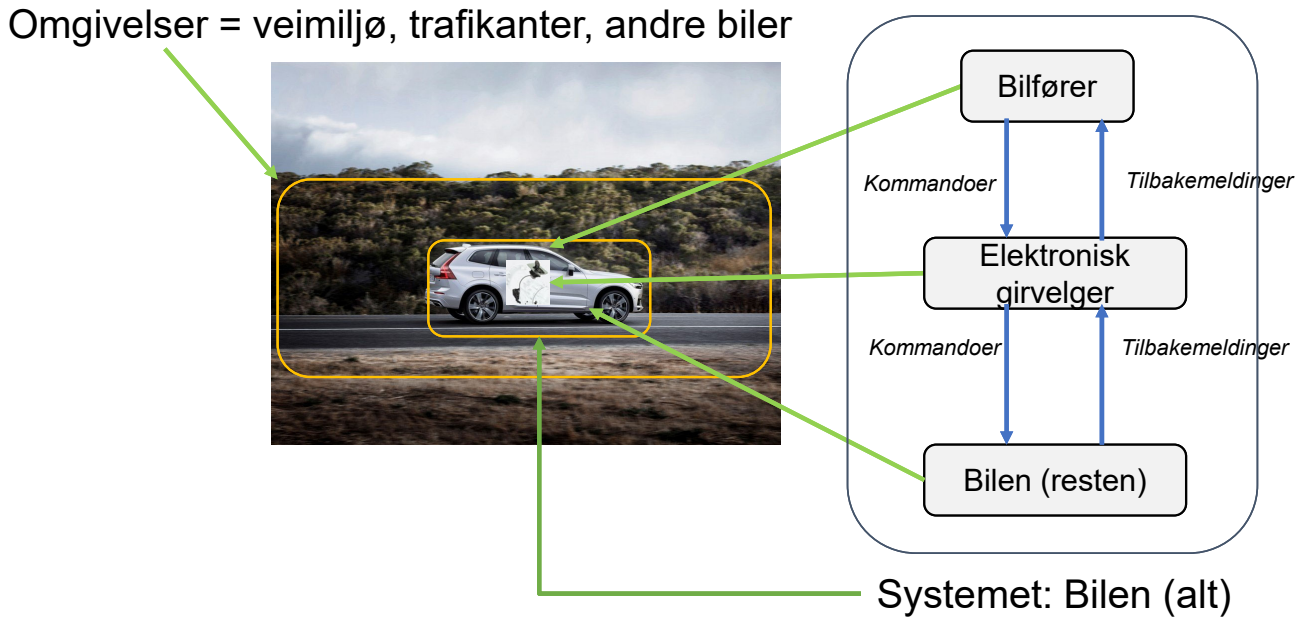
Artikkel: http://sunnyday.mit.edu/STAMP/SAE-STPA.pdf
(Merk at denne artikkelen bruker noen andre begrep fordi STPA kom i en ny versjon i 2018)

Steg 1

1

Klargjør formålet med analysen

- **Beskriv systemet m/omgivelser**
- Definere tap/loss (L)
- Definere system level hazards (H)
- Overordnede systemkrav (til prosessen som styres) [SC]



Steg 1

1

Klargjør formålet med analysen

- Beskriv systemet m/omgivelser
- **Definere tap/loss (L)**
- Definere system level hazards (H)
- Overordnede systemkrav (til prosessen som styres) [SC]

Listen bør ikke overstige 3-5 stykker

Tap kan være tap av menneskelig, kritisk operasjon, miljøskade og tap av rykte

L-1: **Personskade/tap av menneskeliv** når bilen kolliderer med annen bil

L-2: **Personskade/tap av menneskeliv** når bilen treffer terrenget eller fysiske hindringer i veibane

L-3: **Personskade/tap av menneskeliv** av andre årsaker enn kollisjon

*****

(L-4: **Tap av markedsandel** på grunn av alvorlige feil med bilens styresystem)

….

Fra diskusjonen i seminaret: Generell prioritet av tap vil være menneskeliv, skade på miljø, og deretter verdier. Å ha med flere «dimensjoner» er allikevel med på å øke bevisstheten av hva som er å anse som verdier man må og ønsker å beskytte.

Steg 1

1

Klargjør formålet med analysen

- Beskriv systemet m/omgivelser
- Definere tap/loss (L)
- **Definere system level hazards (H)**
- Overordnede systemkrav (til prosessen som styres) [SC]

Listen bør ikke overstige 3-6 stykker

Årsakene til systemsvikt skal ikke beskrives

No	Fare/uønsket oppførsel på systemnivå	Kobling til tap
H-1	Bilen har ikke sikker avstand til andre kjøretøy	L-1
H-2	Bilen opprettholder ikke sikker avstand til terrenget og hindringene langs veien	L-2
H-3	Bilen går i en tilstand som den ikke lar seg kontrollere	L-1, L-2, L-3
H-4	Forhold i bilen gjør at fører og passasjerer eksponeres for fare	L-3

Steg 1

1

Klargjør formålet med analysen

- Beskriv systemet m/omgivelser
- Definere tap/loss (L)
- Definere system level hazards (H)
- **Overordnede systemkrav (til prosessen som styres) [SC]**

Listen bør ikke overstige 3-6 stykker

Systemkrav er **negasjon** av farlig systemoppførsel

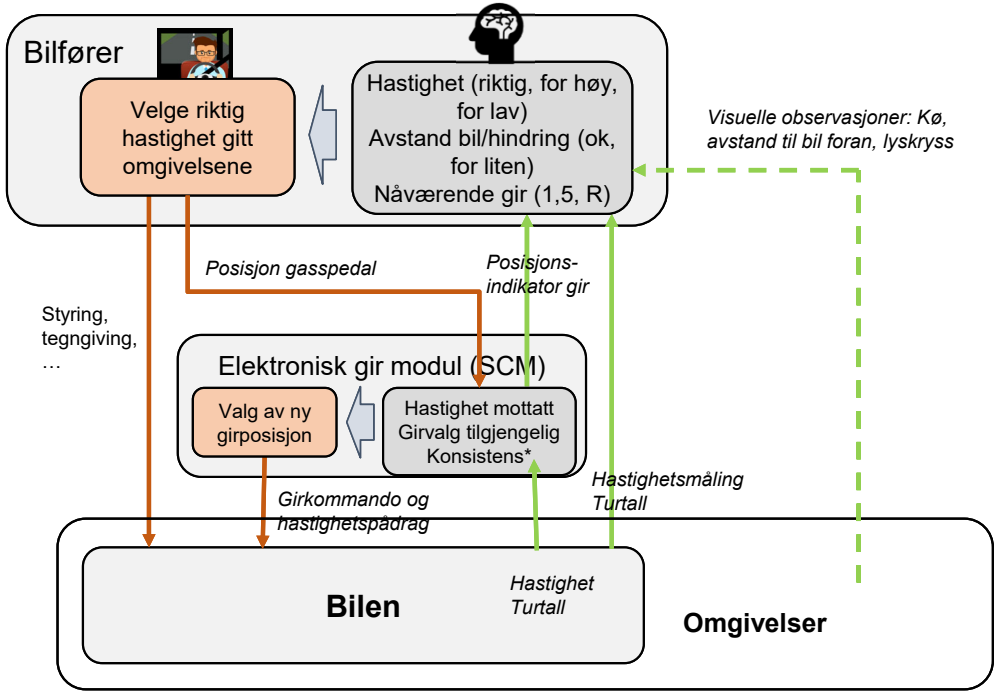
No	Systemkrav	Kobling til uønsket system-oppførsel
SC-1	Bilen opprettholder alltid tilstrekkelig avstand til bilen foran	H-1
SC-2	Bilen opprettholder alltid tilstrekkelig avstand til andre fysiske hindringer i og ved veibanen	H-2
SC-3	Bilen holder seg innenfor veibanen	H-2
SC-4	Bilen må kunne nå en definert sikker tilstand ved tap av kontroll med bilen	H-3
SC-5	Bilen må ha systemer for begrense skadeomfang ved hendelser som brann, deler som løsner,..	H-4

Steg 2

2

Utarbeid **hierarkisk kontroll-struktur**

- Lage figur m/ kontrollere (kontrollere, feedback, interaksjon mellom kontrollere, interaksjon m/operatør/personer,..
- Husk riktig hierarkisk plassering



*Konsistens i forhold til girvalg (nåværende, ønsket) og hastighet og turtall

Steg 3

3

Identifiser «farlige» aksjoner («UCAs»)

Avdekke mulige farlige aksjoner pga:
- **Utført (når det ikke bør)**
- **Ikke utført**
- **For tidlig/for sent/gal rekkefølge,…**
- **Mangelfulle tilbakemeldinger**

Liste opp «controller constraitns» (CC)

I **hvilke** situasjoner kan hver av disse være kritisk?

Control action	Not providing causes hazard	Providing causes hazard	Provided too early, too late wrong order	Provision stopped too soon, applied too long

Hentes fra hierarkisk kontrollstruktur

Steg 3

3

Identifiser «farlige» aksjoner («UCAs»)

Avdekke mulige farlige aksjoner (UCAer) pga:
- **Utført (når det ikke bør)**
- **Ikke utført**
- **For tidlig/for sent/gal rekkefølge,…**
- **Mangelfulle tilbakemeldinger**

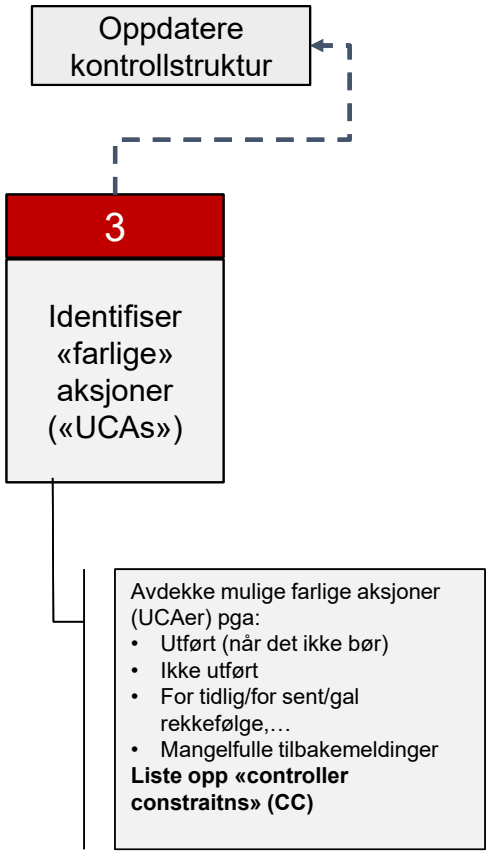
Liste opp «controller constraitns» (CC)

Kontroller: Elektronisk girvelger/modul (SCM)

Control action	Not providing causes hazard	Providing causes hazard	Provided too early, too late wrong order	Provision stopped too soon, applied too long
Girkommando og hastighetspådrag (SCM)	UCA-1: SCM unnlater å gi girvalg og hastighetspådrag som samsvarer med gasspedalpådrag sjøfør [H-1, H-2, H-3, H-4] UCA-2: SCM unnlater å velge et alternativt gir dersom girvalg (plutselig) blir utilgjengelig* [H-1, H-2, H-3, H-4]	UCA-3: SCM gir kommando om nytt girvalg og hastighet uten at fører har endret pedalpådrag hastighet [H-1, H-2, H-3, H-4] UCA-4. SCM gir kommando om et gir som ikke er tilgjengelig [H-3, H-4] ~~UCA-5: SCM gir en girkommando som er farlig (skade på motor, , plutselig fartsendring,…) [H-1, H-2, H-3,H-4]~~	UCA-5: SCM gir girkommando og hastighetspådrag for sent i forhold til sjøførens forventning [H-1, H-2, H-3, H-4] UCA-6: SCM gir kommando om girskifte og hastighetsendring for raskt i forhold til hva sjøfør forventer [H-1, H-2, H-3, H-4]	UCA-7: SCM gir kommando om girskifte og hastighetsendring, men holder på endringen for lenge i forhold til sjaåførens pådrag. [H-1, H-2, H-3, H-4]

**mer enn ett gir kan kanskje brukes for ulike hastigheter…*

Steg 3



Kontroller: Elektronisk gir (SCM)

Vurder hva som mangler for at CC skal oppfylles!

CC	Negasjon av	Beskrivelse	Forhindrer flg system level hazard
CC-1	UCA-1	SCM må gi angi et gir og hastighetspådrag som samsvarer med sjåførens gasspedalpådrag	[H-1, H-2, H-3]
CC-2	UCA-2	SCM må selv angi et tilgjengelig/trygt alternativt gir som ivaretar hastighetspådraget om et gir blir utilgjengelig	[H-1, H-2, H-3]
CC-3	UCA-3	SCM må ikke gi ny girkommando og hastighetspådrag uten at sjåføren har endret pådrag	[H-1, H-2, H-3, H-4]
CC-4	UCA-4	SCM må ikke gi en girkommando for et gir som har status som utilgjengelig	[H-3, H-4]
CC-5	UCA-5,6	SCM må gi girkommando og hastighetspådrag innen X ms etter fører har valgt nytt gir	[H-1, H-2, H-3]
CC-7	UCA-7	SCM må opprettholde valgt gir og hastighetspådrag til sjåfør gir nytt pådrag	[H-1, H-2, H-3]

Mangler tilbakemelding om aktivt gir, hastighet og turtall fra bilen

Steg 3

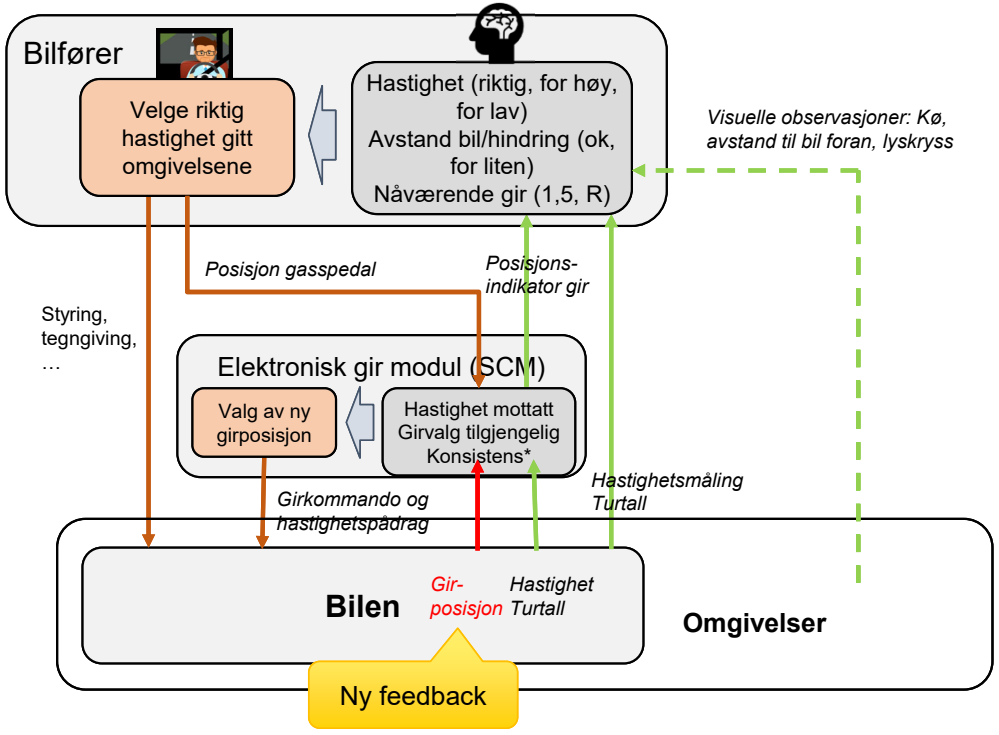
Oppdatere kontrollstruktur

3

Identifiser «farlige» aksjoner («UCAs»)

Avdekke mulige farlige aksjoner (UCAer) pga:
- Utført (når det ikke bør)
- Ikke utført
- For tidlig/for sent/gal rekkefølge,…
- Mangelfulle tilbakemeldinger

Liste opp «controller constraitns» (CC)



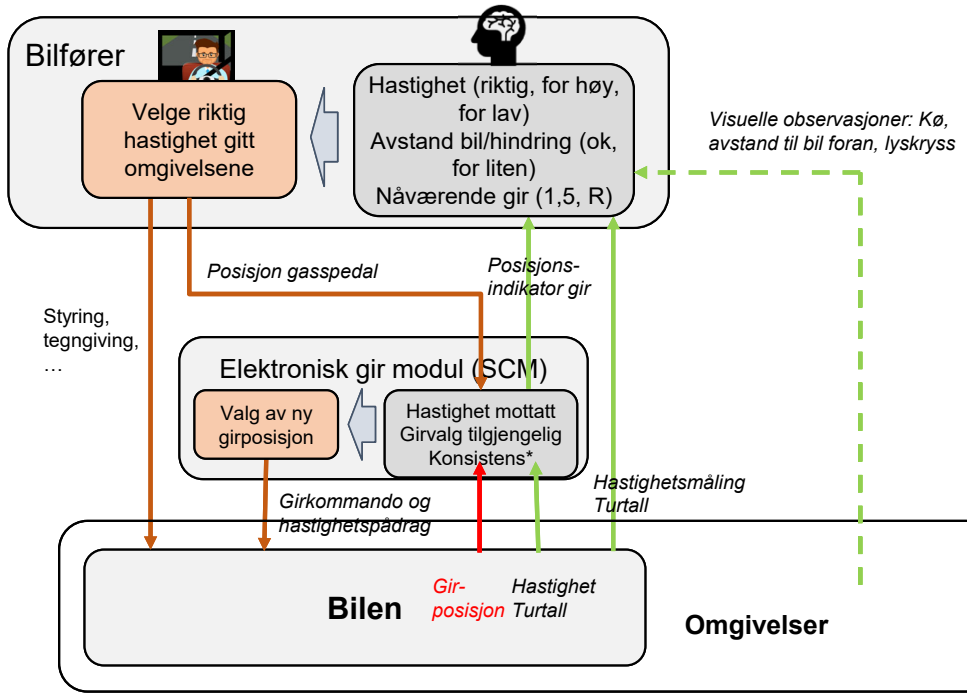
Steg 4

4

Identifiser tapsscenarier (LCer)

- Identifisere årsaker til de ulike UCAene (årsak→ UCA → tap = Loss scenario)
- Oppdatere system constraints (SC)

Følg UCA gjennom HELE sløyfen og se etter årsaker til at den inntreffer



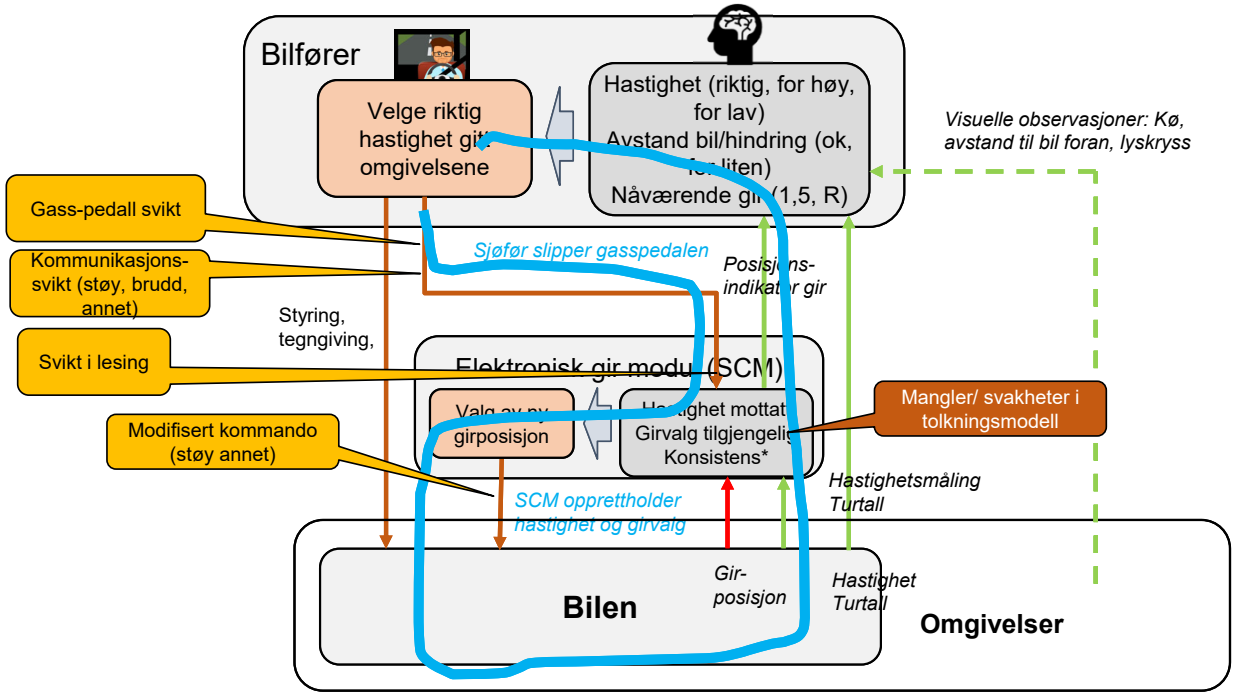
Steg 4

4

Identifiser tapsscenarier (LCer)

- **Identifisere årsaker til de ulike UCAene (årsak→ UCA → tap = Loss scenarios)**
- Oppdatere controller constraints (CC)

Eksempel: UCA-1: SCM unnlater å gi girvalg og hastighetspådrag som samsvarer med gasspedalpådrag sjøfør



Steg 4

4

Identifiser tapsscenarier (LCer)

- Identifisere årsaker til de ulike UCAene (årsak→ UCA → tap = Loss scenario)
- **Oppdatere controller constraints (CC)**

CC	Negasjon av	Beskrivelse	Forhindrer flg system level hazard
CC-1	UCA-1	SCM må gi en gir og hastighetspådrag som samsvarer med sjåførens gasspedalpådrag	[H-1, H-2, H-3]

CC	Beskrivelse	Loss scenario	*Konkretisering av* (CC)
CC-1	SCM må gi en gir-kommando og hastighetspådrag i henhold til fører gasspedalpådrag fra sjåfør	SCM gir ikke girkommando og hastighetspådrag i henhold til sjåførens pådrag fordi …	
		LC-1.1 … gasspedalen har sviktet	CC-1.1 Alarmer sjåfør* hvis gasspedal har svikt (om det er mulig å innføre en sensor for det)
		LC-1.2… det er brudd i kommunikasjonslinken eller støy på signalet.	CC-1.2 Alarmer sjåfør* hvis kommunikasjonssvikt (antar mulighet for diagnose)
		LC-1.3 …SCM har en svikt i lesing av signalet	CC-1.3: Overvåke kommunikasjon fra førerpanel og SCM og gi alarm med brudd/feil.
		LC 1.4 .. SCM har mangel i tolkning av lest signal	CC-1.4: Oppdater logikk med ….
		LC 1.5: … SCM tror at bilen allerede har hastighet og girvalg i henhold til sjåførens pådrag	CC-1.5: Vurdere to ulike måter å måle hastighet på og som leses av SCM

*Antar at det ikke er mulig å la SCM bestemme aksjon i disse tilfellene. Ikke gitt hvilke valg sjåføren har.

Avslutningsvis

Når er analysen ferdig?

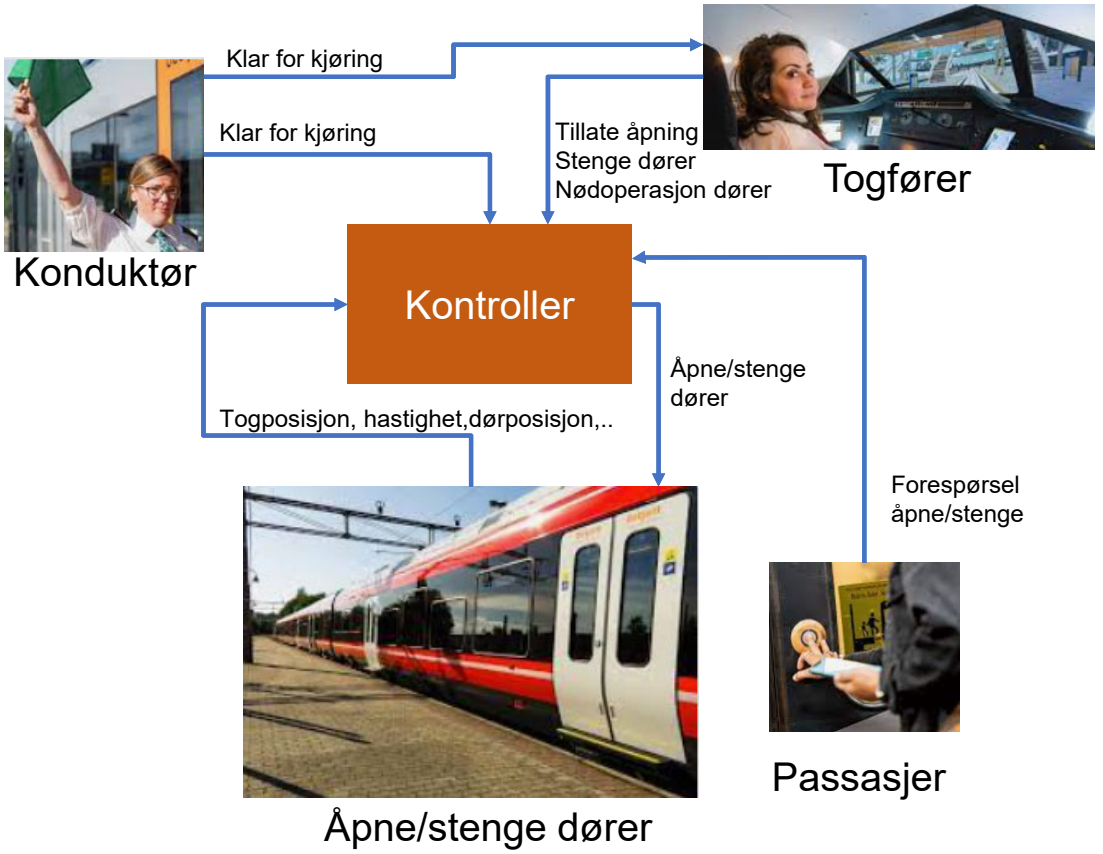
- Det er vanskelig å avgjøre om en STPA analyse er uttømmende (altså at ingenting er oversett)
- Kvaliteten på analysen er sterkt påvirket av **forberedelsene** til analysen og **hvem som deltar i gjennomgangen (i workshopen)**
- Alle nye feil (og dermed reviderte & nye krav) som avdekkes igjennom analysen er seg selv positivt
- Selv hevder STPA-entusiastene at STPA overgår alle andre analyser. Jeg mener at STPA er nyttig supplement.

Forberedelser:
Er underlagsdokumenter og informasjon tilstrekkelig

Hvem som deltar i workshopen
Dekkes alle relevante disipliner (fag, drift,…)
Ledes workshopen på en strukturert måte
Er underlaget som brukes i workshopen egnet for å støtte diskusjonene?

Hvis vi rekker…

Diskuter steg for steg i STPA metoden for dette systemet



Har vi belyst målsetningen?

Målsetning: Å belyse

I komplekse systemer der mye funksjonalitet er i programvare er tap av sikkerhet ofte et «**kontrollproblem***» og ikke nødvendigvis en konsekvens av fysisk svikt.

Systems theoretic process analysis (**STPA**) påstås å være en god metode for å avdekke slike kontrollproblemer og stille nye og bedre krav til styresystemet.

*Feil, avvik eller rett og slett uforutsett effekt av styringsalgoritmer og tilhørende kommunikasjon m sensorer og aktivert utstyr

Gruppeoppgave 2 (TTK2)

Gruppeoppgave 2 TTK2 (mellom seminar 2 og 3)

- Gi eksempler på argumenter for hvorfor STPA egner seg for å analysere systemer med emergente egenskaper.

- Gjennomfør en STPA analyse for systemet* dere har valgt (eventuelt nytt om det er å foretrekk) med fokus på å gi eksempler på hva man gjør i hvert steg.

Dokumenter resultatene i maks 10 slides. Lag illustrative figurer.

*Ikke velg akkurat det eksempelet som er valgt for STPA analysen i forelesningen

Case eksempel (TTK2)