|  |  |
| --- | --- |
| SWT – Air Traffic Monitor 2 | |
|  |  |

Handin – Gruppe 10

2018

Daniel Tøttrup – au544366 – 201509520

Mathias Lønborg Friis – au547006 – 201505665

Lasse Lildholdt – au546990 – 201507170

Philip Nygaard Schmidt – ay547068 - 201506381

# Jenkins Projects and Repo URL

Jenkins Unit <http://ci3.ase.au.dk:8080/job/SWT_10_ATM2_UNITTEST/>

Jenkins Integration <http://ci3.ase.au.dk:8080/job/SWT_10_ATM2_INTEGRATION/>

Jenkins Coverage <http://ci3.ase.au.dk:8080/job/SWT_10_ATM2_COVERAGE/>

Jenkins Static Analysis <http://ci3.ase.au.dk:8080/job/SWT_10_ATM2_STATICANALYSIS/>

Github repo <https://github.com/mathiasfriis/SWT_10_ATM_part_2>

Indholdsfortegnelse

[Jenkins Projects and Repo URL 2](#_Toc530577011)

[Refleksioner 4](#_Toc530577012)

[Softwarearkitektur 4](#_Toc530577013)

[Integrationstest og strategien for denne 5](#_Toc530577014)

[Arbejdsfordelingen 6](#_Toc530577015)

[Konklusion 7](#_Toc530577016)

[Appendix 8](#_Toc530577017)

[Appendix 1 – Klassediagram udkast 8](#_Toc530577018)

[Appendix 2 – Sekvensdiagram udkast for handleNewTrackData 9](#_Toc530577019)

[Appendix 3 – Revideret klassediagram 10](#_Toc530577020)

[Appendix 4 – Sekvensdiagram: Ny trackData modtages, og der opstår en ny Seperation Event 11](#_Toc530577021)

[Appendix 5 - Sekvensdiagram: Ny trackData modtages, og Seperation Event eksisterer allerede 12](#_Toc530577022)

[Appendix 6 - Sekvensdiagram: Ny trackData modtages, og Seperation Event fjernes 13](#_Toc530577023)

[Appendix 7 – Part 2 Opdateret klassediagram 14](#_Toc530577024)

[Appendix 8 – Part 2 Dependency Diagram 15](#_Toc530577025)

[Appendix 9 – Part 2 Sekvensdiagram: Ny TrackData modtages, og der oprettes et TrackEnteredEvent 16](#_Toc530577026)

[Appendix 10 – Part 2 Sekvensdiagram: Ny TrackData modtages, og der oprettes et TrackLeftEvent 16](#_Toc530577027)

[Appendix 11 – Part 2 Sekvensdiagram: HandleNewTrackData 16](#_Toc530577028)

# Refleksioner

## Softwarearkitektur

**Resultat af feedback fra Handin-1**  
På baggrund af feedback fra Handin-2 afleveringen af ATM del 1 har gruppen i starten af design-processen af ATM del 2 besluttet at vores ATM klasse har haft behov for at skulle følge Single Responsibility-princippet bedre. Dette vil også medføre et system og en klasse som er nemmere at kunne teste.

Yderligere havde gruppen misforstået kravet om inkludering af speed og course kravet til HandIn-1. Derfor er denne funktionalitet nu implementer og derfor en del af trackdata og hele systemet ved denne handin.

**Render og logger**Yderligere har gruppen valgt at refactor render og logger klasserne således disse nedlægges. I det nye design ligger formateringen af renderingen og logging i trackdata, hvorefter stubberne consoleOutput og fileOutput håndterer selve udskrivning i hhv. konsollen og logging-filen.

**Event-funktionalitet**For at kunne tilgodese den nye ønskede event-funktionalitet i ATM-systemet har gruppen valgt at lave en abstrakt event klasse: Event, samt 3 konkrete event-klasser: SeperationEvent, TrackEnteredEvent og TrackLeftEvent. Det resulterer i at vores kode og solution mere testbar, da vi i systemet forsætter med at implementere ny funktionalitet med lav kobling. Samme tankegang som blev brugt til at implementere Airspace klassen og andre i Handin-1.

**Rendering af consolen**Med et krav om at alle events der sker på nuværende tidspunkt skal renderes på consollen, samt at vi nu har en del event muligheder: SeperatonEvent, TrackEnteredEvent og TrackLeftEvent, er det en nødvendighed at vi får opdateret consollen oftere end når der blot kommer ny trackdata fra transpondereceiveren. Derfor har gruppen implementeret en timer, som sørger for at opdatere renderingen af events hvert 50. millisekund, således vi er sikker på at vi har den nyeste og aktuelle information om de aktive fly i airspacet samt eventuelle events.

Al design-diagrammer findes under Appendix i dette dokument.

## Integrationstest og strategien for denne

Gruppen startede med at lave et dependency-tree, efter at designet for de kommende ændringer til ATM-systemet var på plads.

Undervejs i implementeringen var der dog flere forskellige ting som var nødvendige at ændre undervejs, som vi kender det fra iterative processer. Dette medførte i at dependency-tree’et også skulle kigges på igen, således at denne stemmer overens med vores nuværende design af ATM-systemet.

Herefter har gruppen skulle revurdere hvorvidt vi ønskede at bruge en Bottom-Up-Plan, Top-Down-Plan, Big Bang og Sandwich.

Normalvis vil Sandwich umiddelbart være et godt valg til at lægge plan over integrationstest. Dog skal vi tage i mente at størrelsen af ATM-systemet og en eventuelt integrationsplan for sandwich-modellen, ikke ville give resultatet som normalt ønsker ved en sandwich-model. Det ønskede resultat af sådan en model ville være at man får fordelene fra de andre modeller. Dette er dog ikke aktuelt i denne sammenhæng, da vores ATM-system er for småt i denne kontekst.

Eftersom sandwich-modellen ikke passer i denne kontekst var vi i gruppen nødsaget til at vælge en af de resterende planer for integrations-test. Valget faldt på bottom-up-plan. Dette skyldes at vi i gruppen synes det gav god mening, da vi dermed kunne teste bottom-klasserne inden at vores ATM klasse havde implementeret nye funktioner og refactor’ed gammel kode.

Som resultat af bottom-up kunne vi i gruppen nyde af at arbejdsfordelingen blev mere naturlig, da nogle kunne unit-teste, imens at der forsat blev ændret og implementeret ny funktionalitet i ATM-klassen.

Dependency-tree og et tilhørende table findes under Appendix i dette dokument.

## Arbejdsfordelingen

Vi har i gruppen bestræbt os efter at udarbejde arbejdsfordelingen således at enhver ny funktion der er blevet implementeret, som følge af den nye udleverede kravspecifikation, er blevet implementereret og testet af samme person. På denne måde, sikrer det at alle i gruppen får erfaring med hele forløbet af et en test driven udviklingsproces.

Denne tankegang er fuldt videre over i vores brug af Jenkins som på samme måde er blevet vedligeholdt af alle i gruppen, hvor den hovedansvarlige for at byggeserveren var aktiv hele tiden skiftede således alle opnåede en ønsket erfaring hermed.

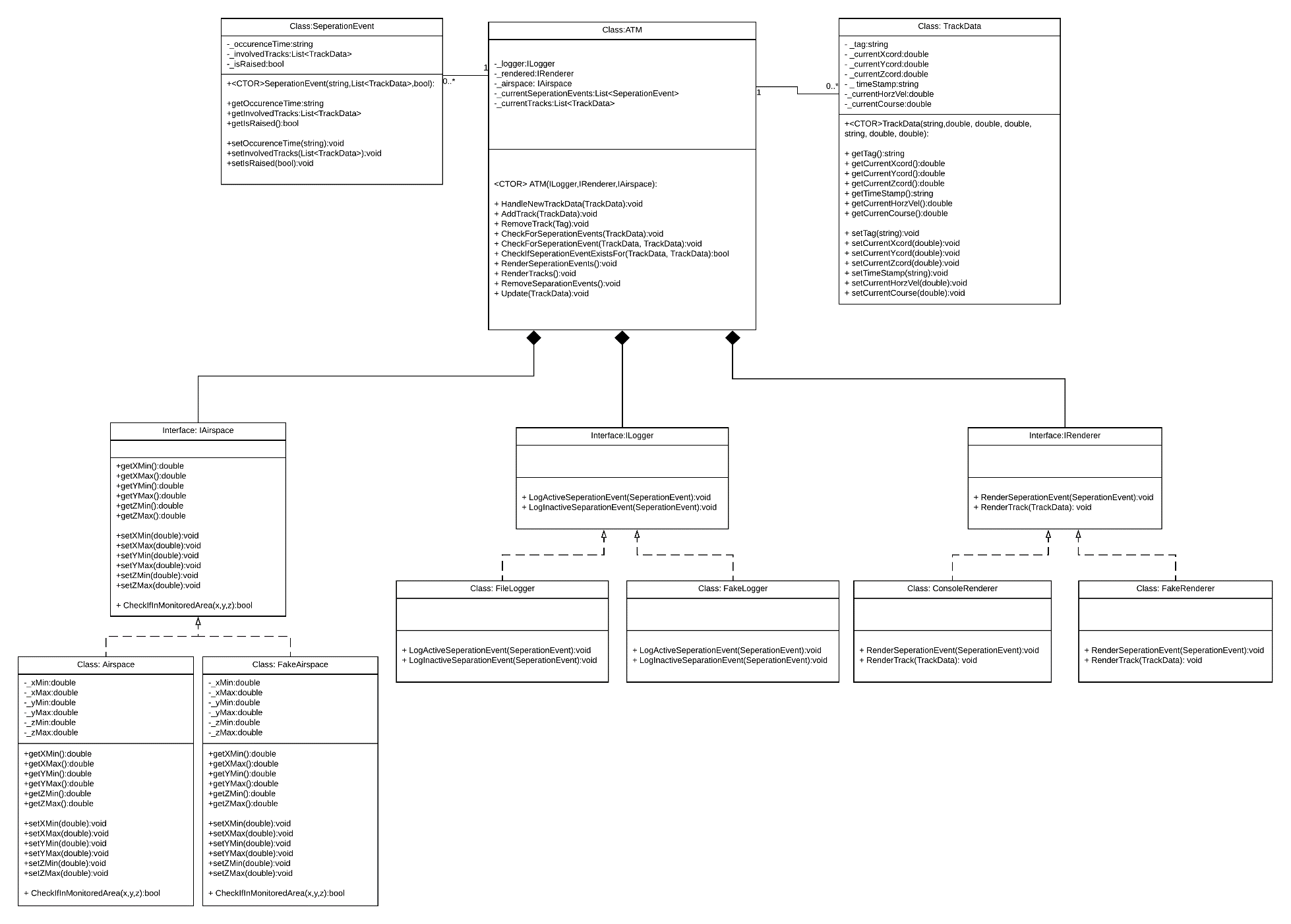
Integrationstestene har i vores arbejdsproces været under revidering flere gange, grundet overvejelser om hvorledes afhængighederne i systemet udartede sig. Dette har resulteret i at flere af de skrevne unittest har kunne gøre det ud for integrationstesten på et givent niveau. Dette har gjort at arbejdsfordelingen fra unittest processen er ført videre med over til integrationstestene.

Af vores repository vil denne arbejdsfordeling ligeledes være tydelig i vores commit tree, der viser et jævnt flow af commits, fra alle gruppens medlemmer.

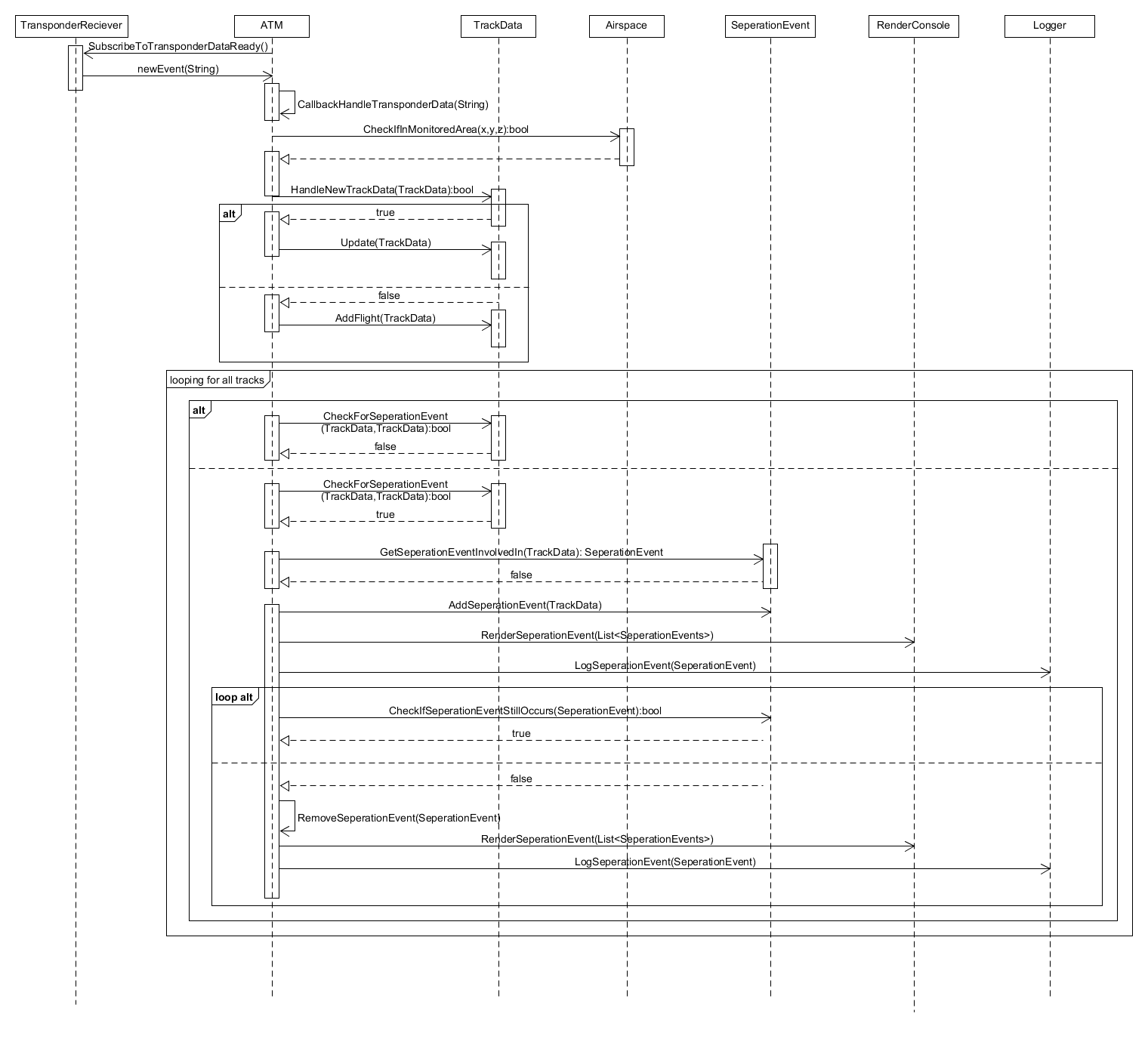
# Konklusion

# Appendix

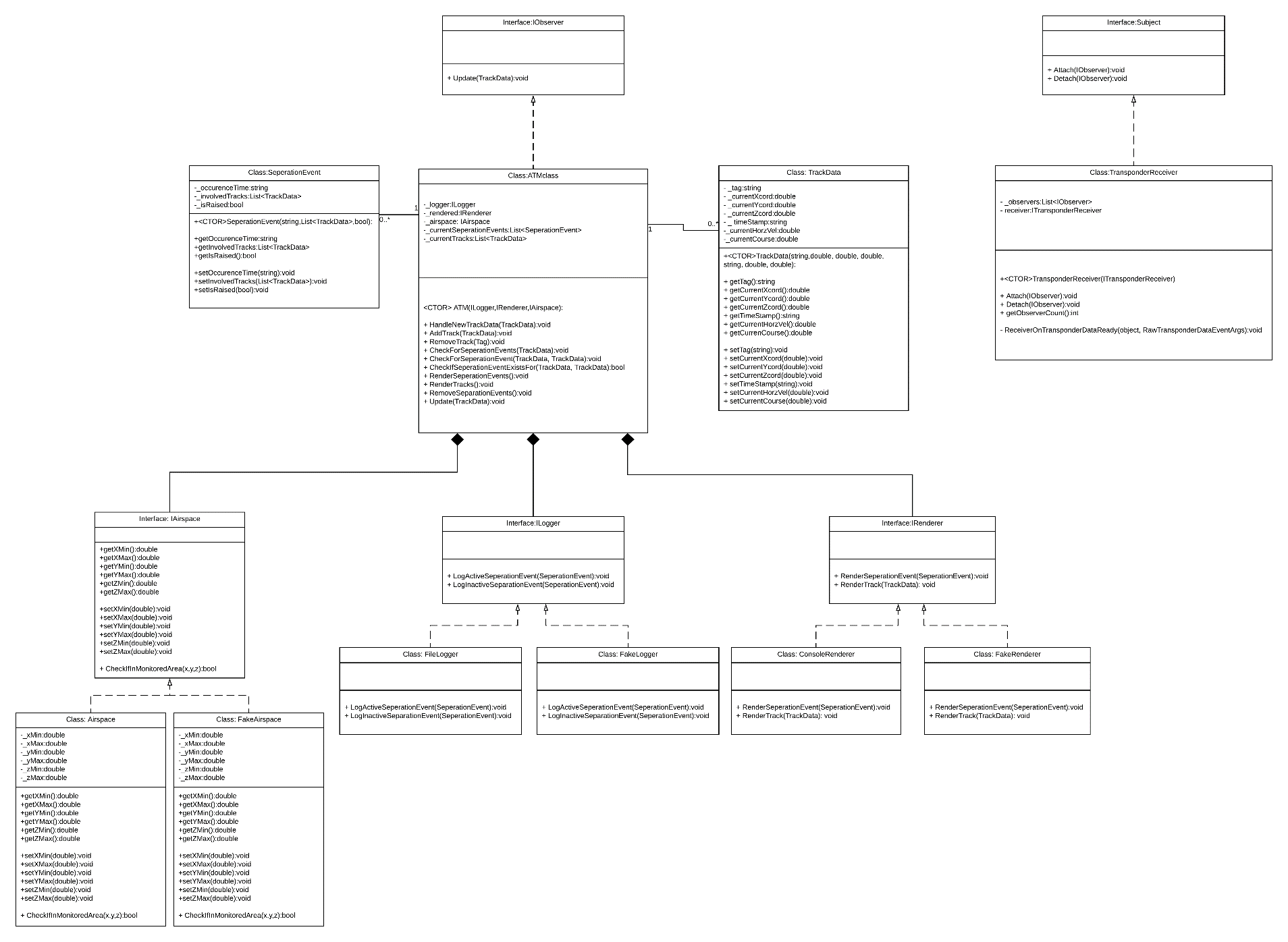
## Appendix 1 – Klassediagram udkast



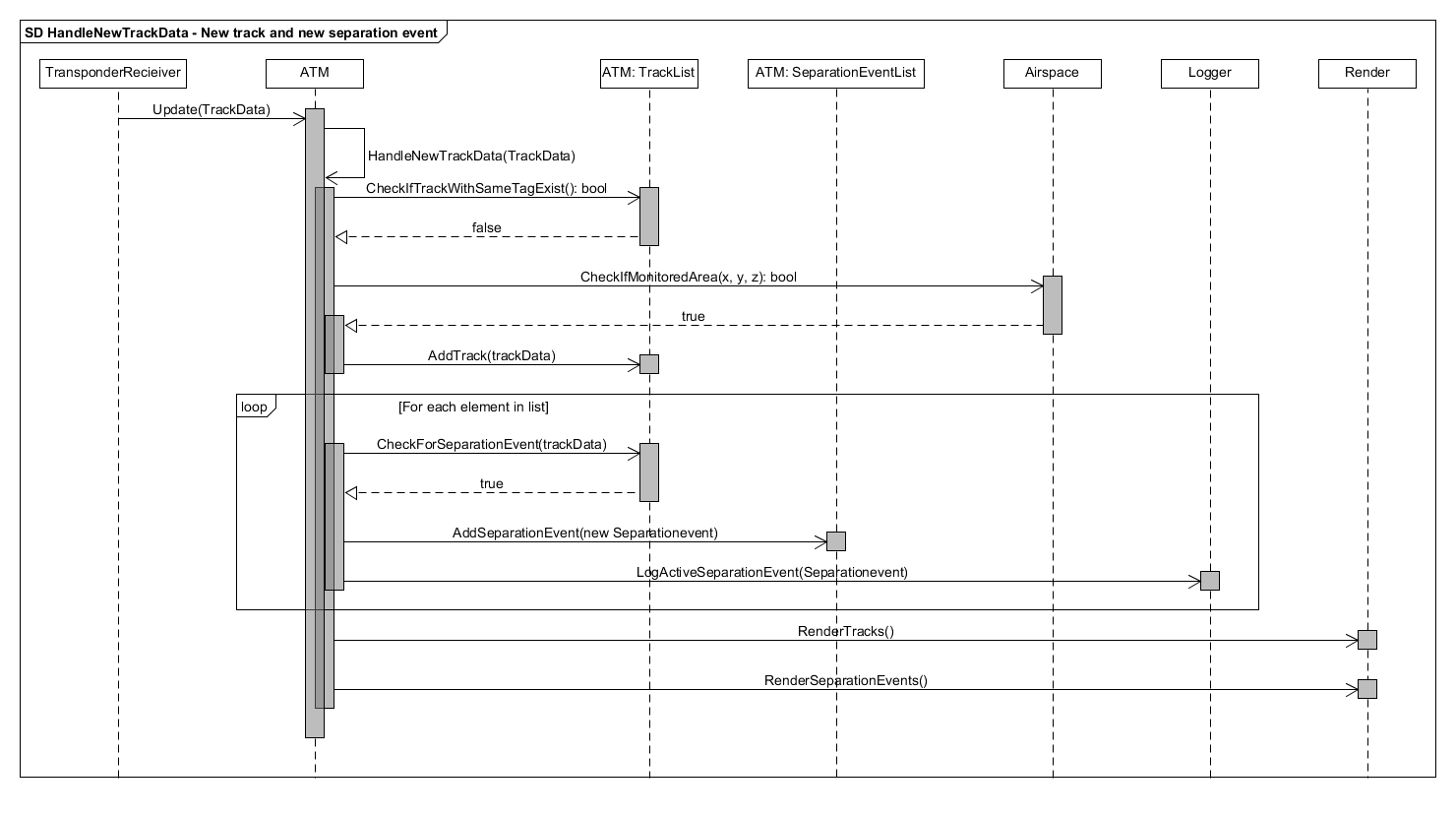
## Appendix 2 – Sekvensdiagram udkast for handleNewTrackData



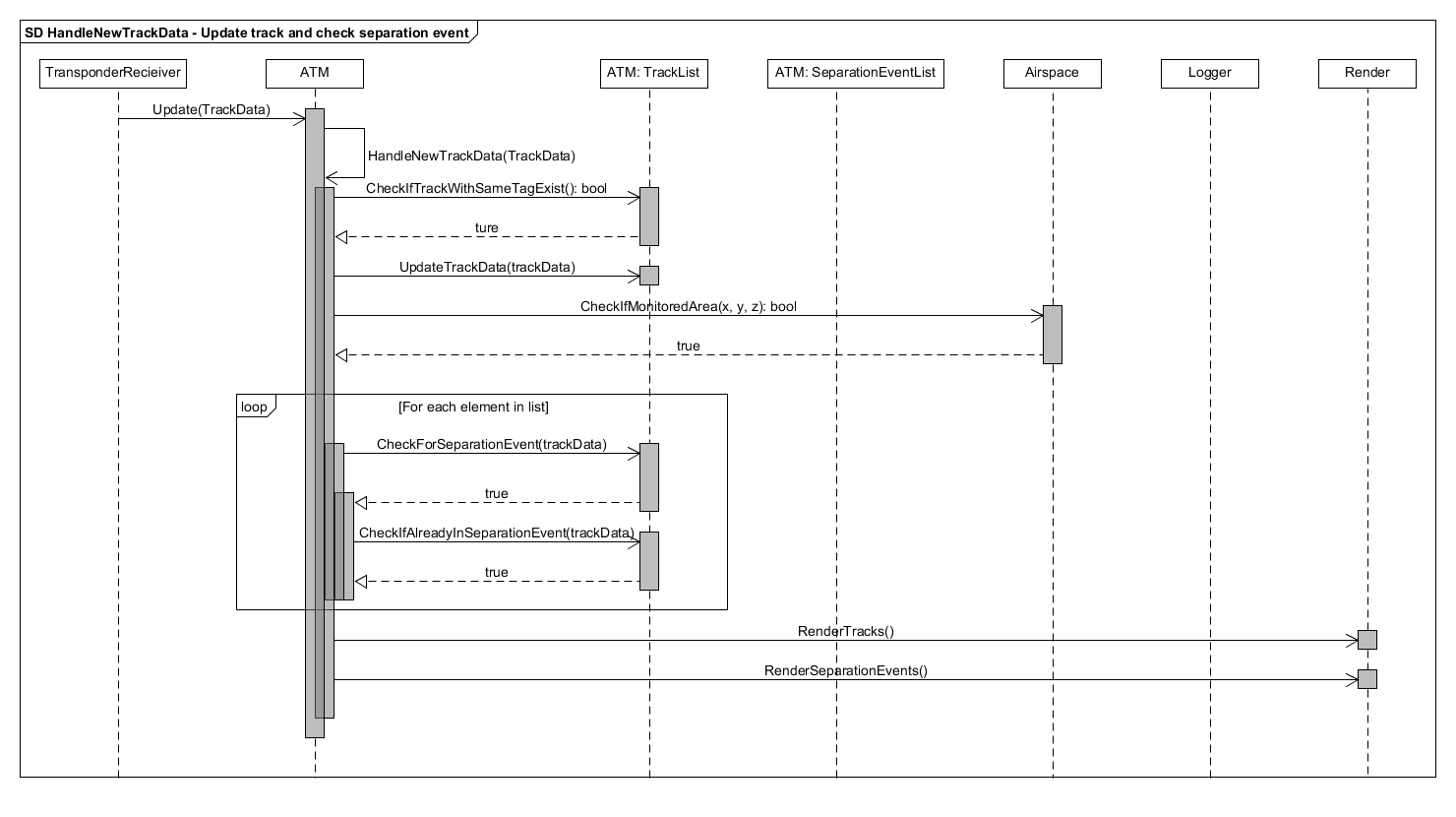
## Appendix 3 – Revideret klassediagram



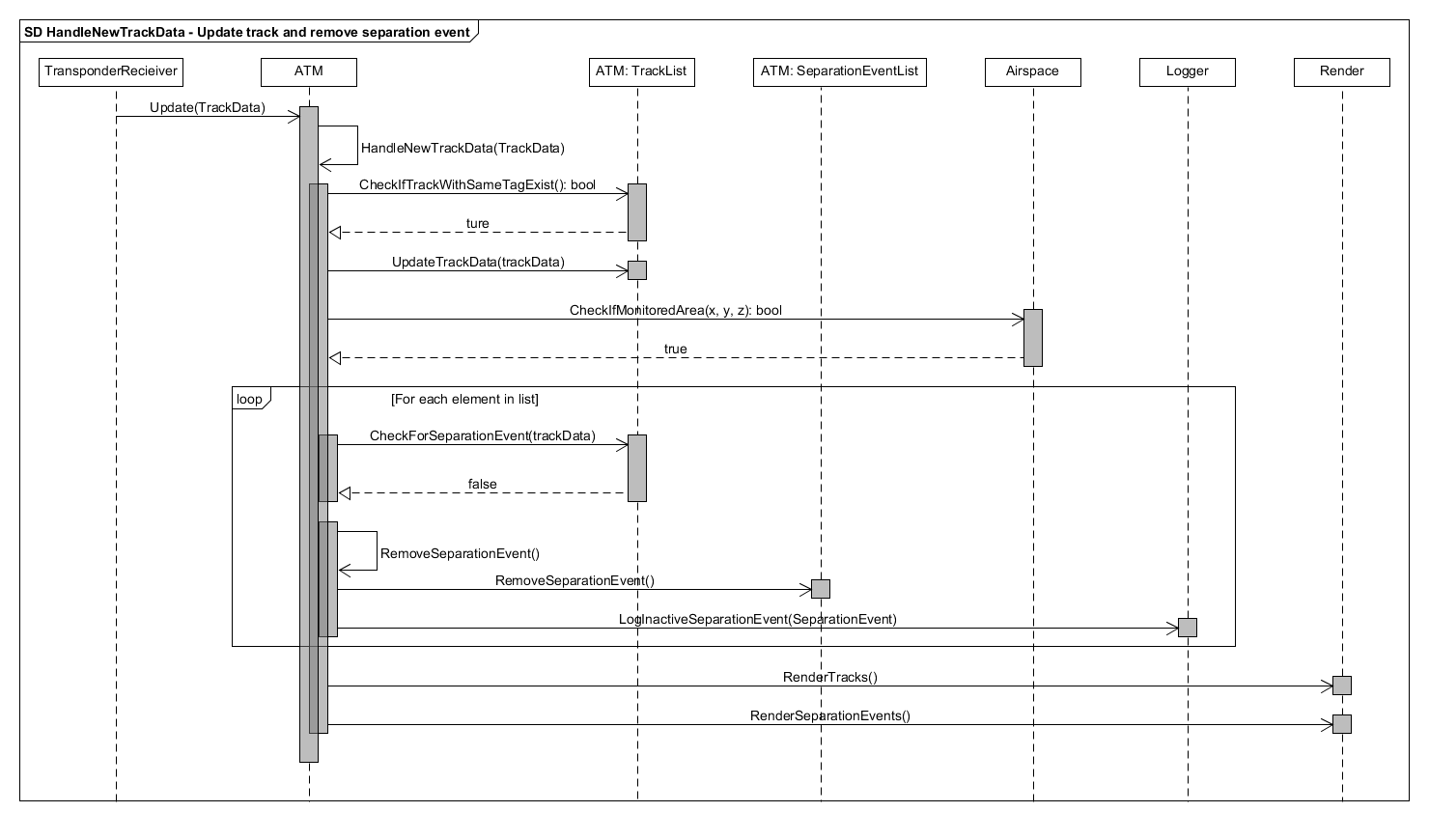
## Appendix 4 – Sekvensdiagram: Ny trackData modtages, og der opstår en ny Seperation Event



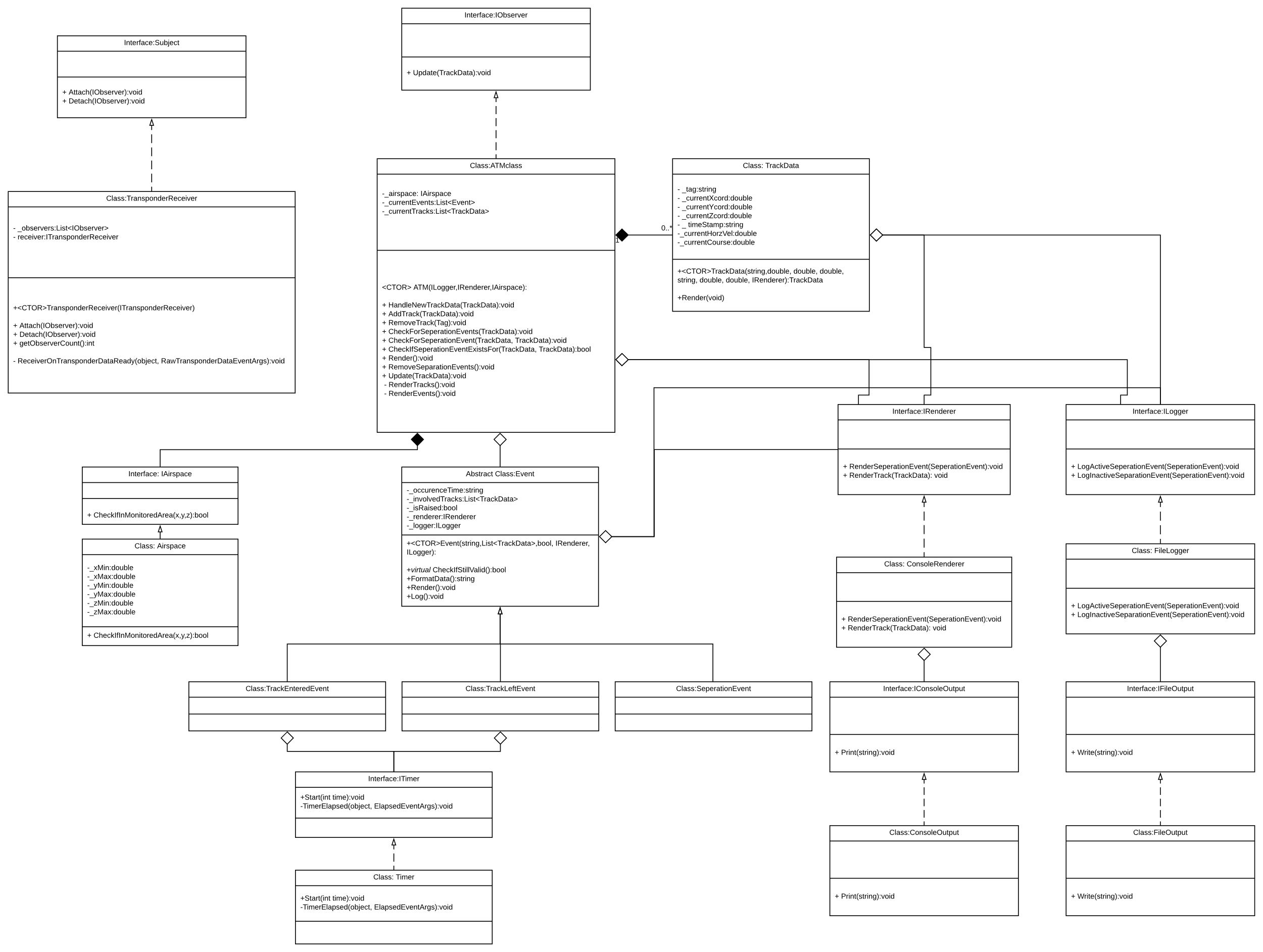
## Appendix 5 - Sekvensdiagram: Ny trackData modtages, og Seperation Event eksisterer allerede



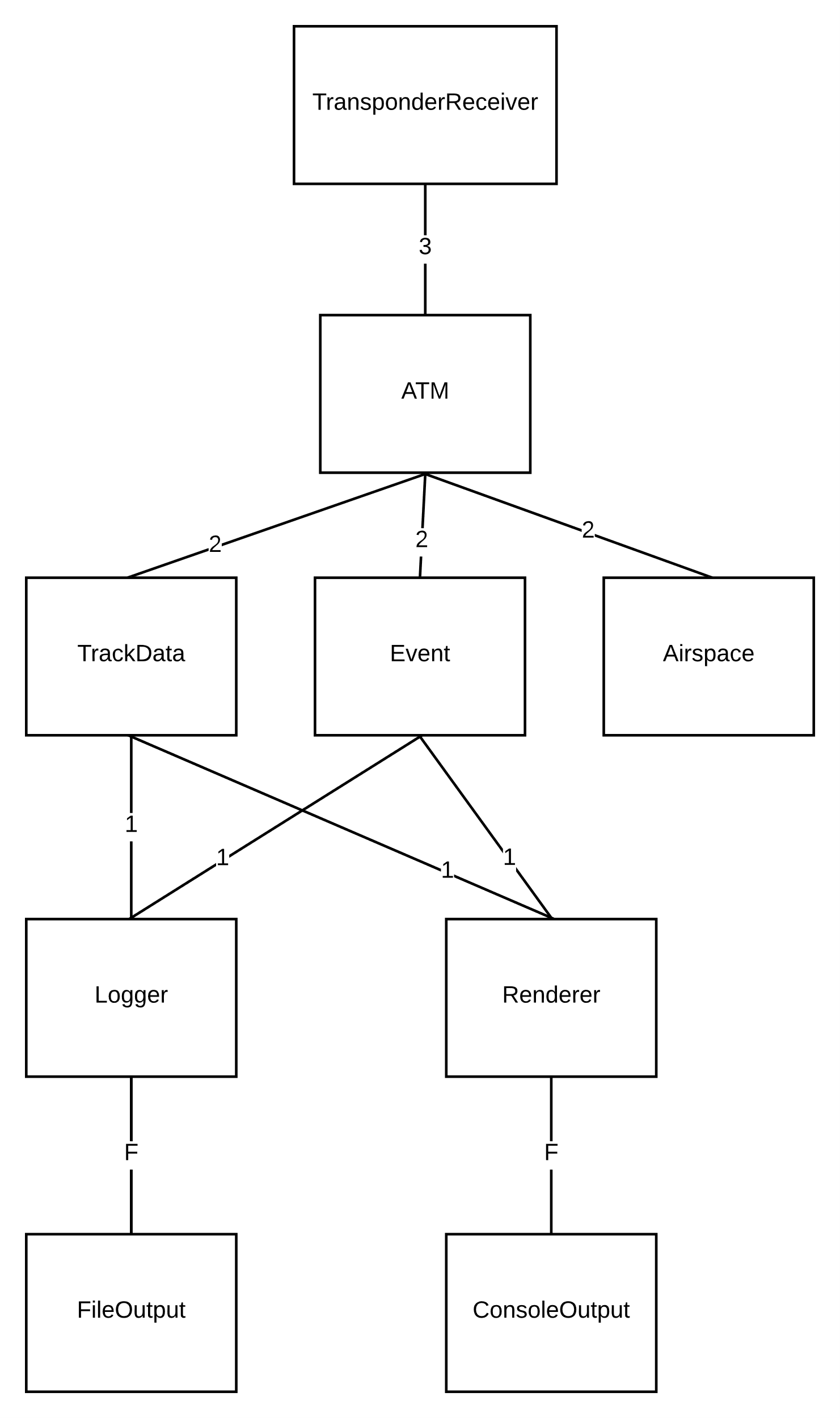
## Appendix 6 - Sekvensdiagram: Ny trackData modtages, og Seperation Event fjernes



## Appendix 7 – Part 2 Opdateret klassediagram



## Appendix 8 – Part 2 Dependency Diagram



## Appendix 9 – Part 2 Sekvensdiagram: Ny TrackData modtages, og der oprettes et TrackEnteredEvent

## Appendix 10 – Part 2 Sekvensdiagram: Ny TrackData modtages, og der oprettes et TrackLeftEvent

## Appendix 11 – Part 2 Sekvensdiagram: HandleNewTrackData