

Raven

H5 – Projekt opgave

Rasmus W. Knudsen & Niklas W. Micheelsen

Indholdsfortegnelse

[Forord 2](#_Toc485481428)

[Indledning 2](#_Toc485481429)

[Projektstyring 3](#_Toc485481430)

[Problemstilling 3](#_Toc485481431)

[Problemformulering 4](#_Toc485481432)

[Rigt billede 4](#_Toc485481433)

[Kravspecifikation 4](#_Toc485481434)

[Udviklingsmiljøer 5](#_Toc485481435)

[Raven-Desktop 5](#_Toc485481436)

[Raven-Arduino 5](#_Toc485481437)

[Raven-Android 5](#_Toc485481438)

[Værktøjer 5](#_Toc485481439)

[Klasse Diagram 6](#_Toc485481440)

[Raven-Desktop 6](#_Toc485481441)

[Kilde Kode 6](#_Toc485481442)

[MainWindow.xaml.cs 6](#_Toc485481443)

# Forord

Denne rapport skrevet ud fra vores projekt under H5, den er skrevet af Niklas W. Micheelsen & Rasmus W. Knudsen. Rapporten omhandler en prototype af vores ’Vogn monitorering og analyse’ softwarepakke.

Vores projekt er tænkt som at være en prototype af en såkaldt ”black box” som kan sælges som en del af en serviceaftale til større transport firmaer, såsom DHL, UPS, 3x34 osv.

Vores Development prototype er delt op i tre dele – OBD-II connector, Arduino og en Android mobil. Når vores ”proof of concept” er fuldt fungerende og Raven-Desktop er færdigudviklet kan vi begynde på fase 2 af Raven.

I fase 2 vil vi arbejde på at minimere prototypen til en lille kasse som man vil kunne efterlade tilsluttet i bilens OBD-II stik. Med en indbygget GPS og et 3G GSM modem vil den kunne opdatere og sende data til Raven servers.

Navnet Raven har vi valgt som en reference til Odins to ravne, Hugin og Munin, som overvåger landskabet fra luften.

# Indledning

Raven GPS er en suite af software designet for at give transportselvskaber muligheden for at tracke og analysere sine chauffører, vogne og deres ruter.

Dette bliver udført ved hjælp af bilens OBD-II stik, der er en ny standard inden for Auto industrien. OBD står for ”On-Board Diagnostics” og giver adgang til bilens CAN Bus som giver adgang til ’hjernen’ af bilen dvs. data så som hastighed, RPM, Motor stress og meget mere. Disse informationer bliver sendt til chaufførens smartphone over bluetooth, der derefter vil blive sendt til Raven GPS’ database.

Et par eksempler på i hvilken situation dette vil være brugbart er f.eks. En arbejdsgiver får en klage over en medarbejders vanvidskørsel. Der vil arbejdsgiver kunne åbne Raven programmet, skrive medarbejderens nummerplade, stelnummer eller potentielt andre identificerende detaljer og se alle ruter en specifik medarbejder har foretaget. Arbejdsgiver vil så kunne trykke på en given rute og verificere at der er hold i klagen og tage aktion.

Eller at en af firmaets vogne brød sammen eller nedkøling af læsset ikke fungerer ordentligt så kan firmaet finde ud af hvor den er ved at slå den op og tilkalde den nærmeste vogn som er tilgængelig til dens position.

## Projektstyring

Til projektstyring har vi brugt Kanbanboard og servicen ”Trello”, Trello giver os mulighed for hurtigt at kunne oprette nye opgraver med detaljeret beskrivelser. Trello virker på den måde at man kan organisere opgaver som digitale post-it notes og tilføje dem til forskellige kolonner.

Når en ny opgave skal oprettes starter den ved at blive tilføjet til en af vores ’To-do’ kolonner for vores forskellige projekt dele. Hvis denne opgave angiver en opgave for Raven-Arduino vil den blive markeret som en Arduino opgave og tilføjes til vores ’To-Do’ kolonne, når opgaven bliver begyndt på bliver opgaven flyttet til ’In-progress’ og eventuelle under opgaver vil derefter blive tjekket af på listen en for en. Når alle under opgaver er fuldført vil opgaven bliver flyttet en sidste gang, til vores ’Done’ kolonne der over tid vil blive længere og længere.

# Problemstilling

**Hvordan kan det være, at i har valgt at lave dette produkt?**

Det er hovedsageligt fordi at vi har stor interesse inden for biler, men også fordi at det ville være spændende at lave et projekt som både var afhængelig af hardware og som benyttede flere platforme, noget som ligner et realistisk produkt og som inkluderede et område vi ikke havde særligt meget kendskab til.

**Hvor kom ideen fra?**

Ideen kom fra at en af vores fædre som havde lavet noget lignede for mange år siden. Noget som han forslog kunne erstattes hvis nogle tog konceptet og forbedrede det. Realistisk set så er hvad vi har lavet en prototype da for at det kan erstatte den gamle version mangler der meget funktionalitet. Men det er hovedsageligt der inspirationen kom fra.

**Hvad ville i gerne lære igennem arbejdet?**

Vi ville gerne lære mere om RTOS da det er et ekstremt brugbart værktøj i den virkelige verden, det har været udfordrende at arbejde med Arduino.

# Problemformulering

Problemer, bekymringer eller ting der bør overvejes inden for denne type af produkt.

**Hvordan vil i overføre data fra Arduino til Android?**

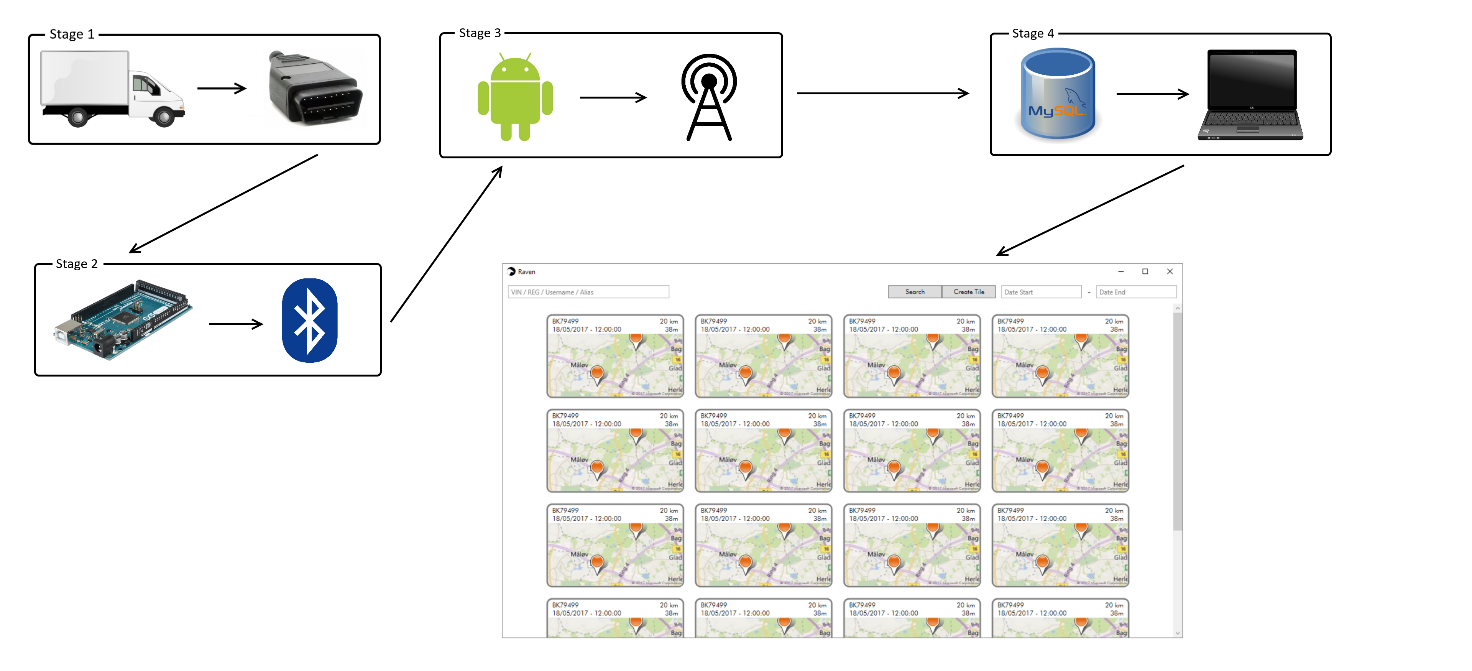
Til vores prototype version valgte vi at bruge Bluetooth imellem Arduino og Android da et Bluetooth modul til Arduino ikke koster mange penge.

Vi havde overvejet et GSM modul (data forbindelse via SIM kort), så var der ikke behov for Android, men vi syntes at et GSM modul ville være for dyrt i første omgang.

**Hvordan vil i lagre data?**

Vi valgte at lagre vores data i Json format på en SQL server, det endte med at blive en LONGTEXT da det største antal tegn tilladt. Der er sikkert hunderede vis af bedre muligheder, men da det er relativt hurtigt at lave indsæt eller udtræk og at vi indtil videre ikke har ramt grænsen for vores logs.

# Rigt billede



# Kravspecifikation

Desktop:

* Søgefunktion hvor brugere kan filtrere ruter efter registrerings numre, førerens brugernavn og andre parametre.
* Vise en boks for hver tur kørt, som indeholder generel information og et kort over ruten. Information som, registrerings nummer, distance, varighed, dato og tid på starten af turen.
* Detaljeret tur visning som viser den fulde rute, punkter som er blevet registreret i løbet af turen som kan vise dato, tid, hastighed, omdrejninger i minuttet, breddegrad og længdegrad på det tidspunkt hvis brugeren klikker på et af punkterne.

Android:

* Modtag data fra bluetooth modulet på Arduino’en.
* Filtrere modtaget Json linjer og formatere til gyldig og læseligt Json logs.
* Send filtreret data til SQL via GSM/Data forbindelse på Android.

Arduino:

* Læs fra OBD-II.
* Efter en læse cyklus start sending.
* Send data til Android i Json format over Bluetooth til Android.

Database:

* Test

# Udviklingsmiljøer

## Raven-Desktop

Miljø: Visual Studio 2017

Raven-Desktop er skrevet i WPF (C#, XAML) ved brug af Bing Maps SDK

## Raven-Arduino

Miljø: Arduino IDE

Raven-Arduino er skrevet i C.

## Raven-Android

Miljø: Android Studio

Raven-Android er skrevet i java.

# Værktøjer

Balsamiq Mockups 3

Visual Studio

Android Studio

Arduino IDE

MYSQL

GitHub

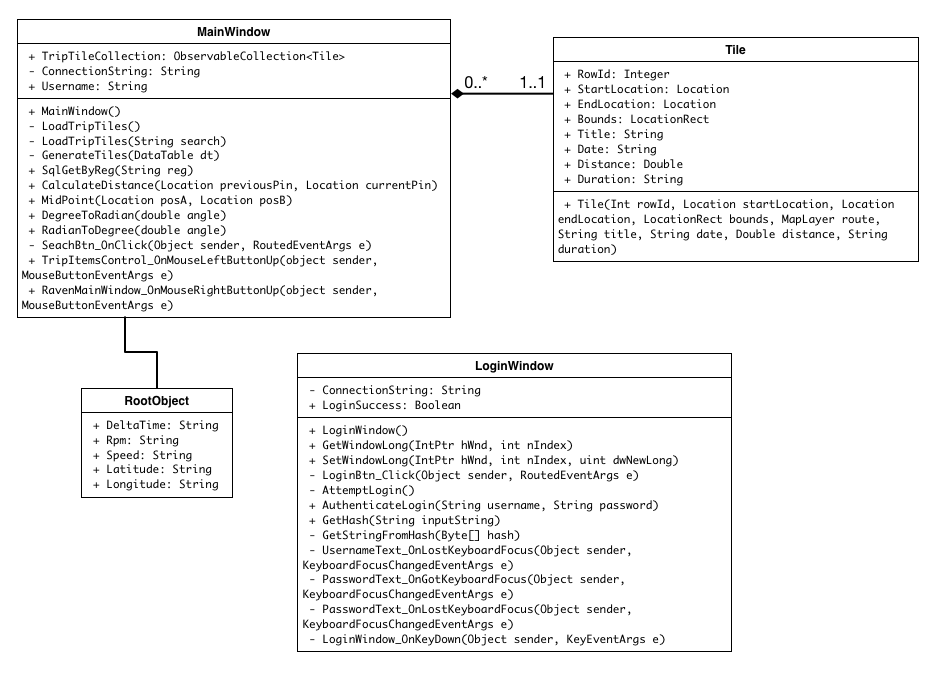
Trello

# Flowcharts

* TripTiles are clickable boxes in our TileViewer that show general information about a trip, along with a map of where it was driven.

# Klasse Diagram

## Raven-Desktop



## Kildekode

## Desktop

### MainWindow.xaml.cs