TrackABus

BACHELORPROJEKT

Procesrapport for TrackABus

Author:

Gruppe 13038

Supervisor:

Michael Alrøe

14. december 2013

Versionshistorie:

Ver.	Dato	Initialer	Beskrivelse
1.0	12-12-2013	13038	Start på rapportskrivning

Godkendelsesformular:

Forfatter(e):	Christoffer Lousdahl Werge (CW) Lasse Sørensen (LS)
Godkendes af:	Michael Alrøe.
Projektnr.:	bachelorprojekt 13038.
Filnavn:	Procesrapport.pdf
Antal sider:	14
Kunde:	Michael Alrøe (MA).

Sted og dato: _____

10832 Christoffer Lousdahl Werge

MA Michael Alrøe

09421 Lasse Lindsted Sørensen

1 Resumé

I forbindelse med bachhelor projektet, er der blevet udarbejdet et system til at indlæse positionen for en bus, vise denne på et kort, samt underette brugeren om, hvor lang tid der er til, at en bus er ved et givent stoppested. Desuden er der også mulighed for, at ruter kan favoriseres, og hermed gemmes lokalt. Ud over bruger funktionaliterne, understøter systemet også muligheden for, at en adminstrator kan oprette, vedligeholde og slette busser, busruter og stoppesteder.

Persisteringen af data sker i form af to relationelle databaser; En lokal og en distribueret. Den lokale giver brugeren mulighed for at gemme ruter der bruges ofte, og den distribuerede indeholder alt information om ruterne, samt bussernes nuværende og tidligere position. Den distribuerede database indeholder også funktionalitet til at udregne tiden for en bus til et givet stoppested.

Den distruberede database er opbygget ved hjælp af MySQL, og den lokale er opbygget ved hjælp af SQLite. Brugeren tilgår systemet igennem en Android platform, hvorigennem samtlige bruger-funktionaliteter kan tilgås. Koodesproget brugt til dette er Java. Administrator hjemmesiden er opbygget ved hjælp af HTML, CSS og JavaScript, men da den er bygget på baggrund af ASP.NET er C# det primære kodesprog i denne sammenhæng. Til systemet er der blevet designet en simulator, som står for at simulere en bus, i alle henseende. Den er blevet udarbejdet i WPF ved brug af .NET framworket, med C# som kodesprog.

Til projektstyring er der blevet brugt dele af Scrum, og herunder er V-modellen fundet yderst brugbar.

Det færdige produkt er et system, hvori en bruger kan holdes opdateret omkring busser placering, og hvori en administrator nemt kan udføre vedligeholdelse. In relation to the bachelor project, a system has been designed, to retrieve the position of a bus, draw it on a map, and notify the user of how much time there is, until the closest bus, is at the chosen stop. The user system also implements the functionality, that a route can be favoured, and saved locally. Besides this, the system also supports the possibility, that a administrator can create, maintain and delete buses, routes and stops.

The persistence of data, is done by two relational databases; One locally and one distributed. The local database makes it possible for the user to save a route that is used

often, and the distributed contains all the information regard the routes, as well as the previous and current positions of the buses. The distributed database also contains the functionality to calculate the time until a bus reaches a specified stop.
The distributed database is created with MySQL and the local is created with SQLite. The user accesses the system through an Android platform, where all user-functionalities can be access through. The language used for this is Java. The administrator homepage, is created with HTML, CSS and JavaScript, but since it is created as an ASP.NET application, C# is the primary language used. A simulator has been designed for the system, with the purpose of simulating the complete functionality of a bus. This has been created as a WPF application, with the .NET framework, and with C# as a language.
For the purpose of project management, has parts of scrum been used, and through this the V-model was found to be very useful.
The final product is a system, where the user can be kept updates in regard the the position of a bus, and where an administrator easily can perform management.

Indhold

2 Forord

Dette projekt er udarbejdet af to studerende på Ingeniørhøjskolen i Aarhus. Projetet udgør, i sammenhæng med dette dokument og dokumenterne TrackABus Kravspecifikation, TrackABus Accepttest specifikation og TrackABus Systemarkitektur, bachelorprojektet på syvende semester for IKT-Linjen. Dokumentopsætningen er udgjort på baggrund af den projektform, der er blevet arbejdet med i første til fjerde semester.
Bachelorprojektet er lavet uafhængtigt af et firma, og er derfor udarbejdet på egen hånd, af projektgruppens medlemmer.

3 Indledning

En bus følger en ruteplan, men det er ikke altid at bussen er ved et givent stoppested, præcis på det tidspunkt det forekommer i ruteplanen. Det vil derfor være gavnligt at kunne vide, præcis hvor en bus er, og hvor lang tid der er, til den nærmeste er ved et givet stoppested. Denne viden vil, for det første, give brugeren en større chance for at nå sin bus og, for det andet, med sikkerhed vide, om en bus er kørt fra et givet stoppested. Dette dokument beskriver udviklingen af en mobilapplikation der kan vise rute, stoppesteder og busser på en rute, samt vise tiden til et stoppested for en bus. Desuden beskriver dokumentet også oprettelsen af et administrator værktøj, hvori busser, ruter og stoppesteder kan vedligeholdes. Mobilapplikationen er designet til android, men kan nemt genskabes på en anden platform, da samtlige funktionaliteter ligger på en server. Serveren er dog ikke klargjort til et distribueret system, men kan nemt skiftes ud med et der er, hvis det skulle være nødvendigt.

Kravene er blevet udarbejdet iterativt, da projeketet ikke har ekstern kunde, og således ikke repræsenterer kundens krav til systemet, men derimod udviklernes. En kunde er blevet simuleret, i form af en vejleder, således at alle krav-ændringer og accepttest udførsel, blev gemmemgået med hjælp fra en ekstern kilde.

TrackABus mobilapplikation og administrator hjemmeside, er blevet udhviklet til at håndtere system kravene. Systemet fungerer således, at en bruger kan vælge en rute, sådan

samtligt persisteret information omkring ruten kan blive vist. Dette inkluderer busser, stoppesteder og, selvfølgelig ruten. Brugeren kan herefter tilgå tids-funktionalitet ved at trykke på et stoppested, hvorefter tiden til ankomst for den nærmeste bus i begge retninger, vil blive vist. De komponenter brugeren kan tilgå, skal først oprettes igennem administrations hjemmesiden. Denne del af systemet er defor den eneste, der kan ændre distribuerede rute komponenter. I sammenhæng med persitering bruges der to relationelle databaser; En distribueret, og en lokal til hver mobilapplikation. Den distribuerede håndtere samtlige information om de forskellige komponenter, hvor den lokale bruges i sammenhæng med favorisering af ruter. Mobilapplikationen er baseret på et eksamensprojekt i ITSMAP, lavet af gruppens to medlemmer, og betegnes derfor som legacy code. Hele systemet kan derfor ses som en videreudvikling af dette eksamensprojekt. Der eksisterer ingen yderligere krav, en dem projektgruppen selv har fastsat.

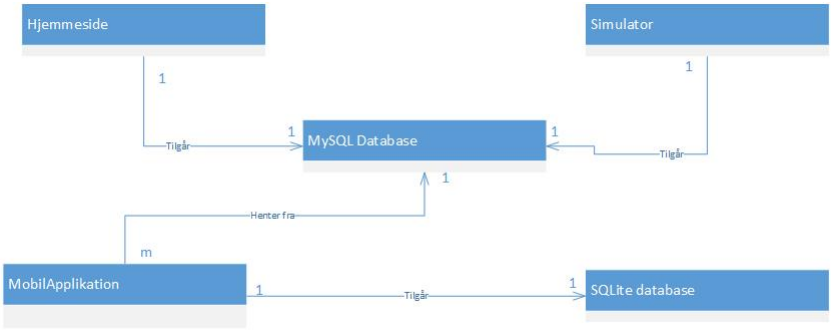
Det har ikke været muligt at tilgå reelle data for busser og ruter. Derfor har det været nødvendigt at bruge en del af arbejdsresourcerne på, at designe og implementere et system til at kunne at kunne håndtere disse data. Dette blev i sidste ende, til administrator hjemmesiden og simulatoren.
I arbejdsprocess øjemed, er der blevet holdt daglige møder i gruppen, hvor dagens arbejde blev diskuteret, samt det ugentlige mål revuderet. Desuden blev der holdt ugentlige møder med ad-hoc kunden, hvori projektes fremskriden blev forklaret og diskuteret.

4 Projektafgrænsning

Da projektet har været udviklet selvstændigt, uden et projektoplæg eller noget fastsatte krav fra en kunde, har der ikke været mange afgrænsninger. En afgrænsning der hurtig opstod, var muligheden for at få GPS-koordinator for busserne, fra midttrafik. Dette gjorde det endnu vigtigere, at hurtigt få lavet en bus simulator der kunne bruges som alternativ.

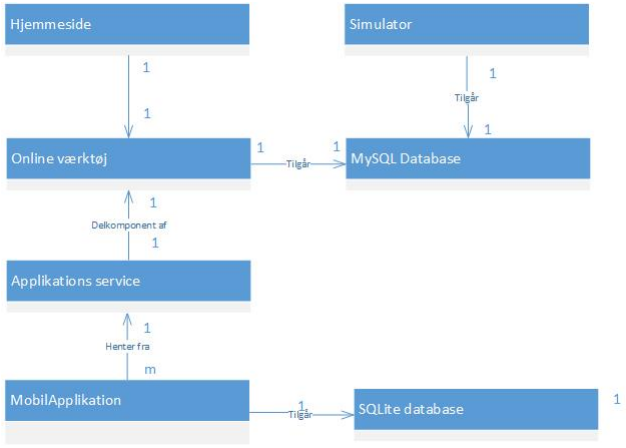
5 Specifikations- og analysearbejdet

Ved påbegyndelse af projektet blev der undersøgt, hvilke domæner projektet skulle bestå af. Dette var i særdeleshed vigtigt, da sytemet består af undersystemer, som udelukkende interagerer over en distribueret database. På figur 1, kan det første udkast til modellen ses.



Figur 1: Første udkast af domænemodellen

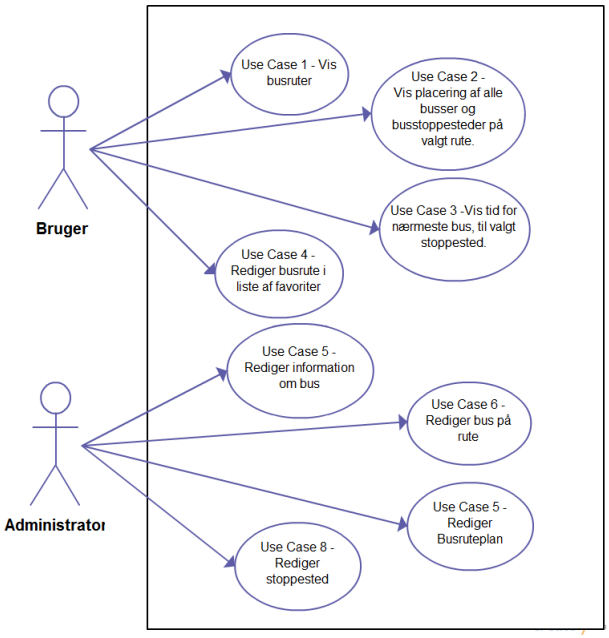
Under udviklingen af systemet, udviklede domæne modellen sig til en ny version. De fleste komponenter forblev de samme, men applikationens tilgang til den distribueret database, blev ændret til at være indirekte. Det vil sige den tilgår en online service, som igen tilgår databasen. Dette sikrer en lav kobling mellem databasen og applikationen, samt en høj samhørighed, da database tilgangen bliver samlet i en komponent ikke direkte knyttet til applikationen. På figur 2 kan den færdige domænemodel ses.



Figur 2: Færdig domænemodellen

Efter udviklingen af domænemodellen, blev der udtænkt de brugssituationer systemet kunne blive udsat for. Simulatoren blev ikke set som en komponent i denne sammen-

hæng, da dette blot var et værktøj til udviklingen. Det var udelukkende hjemmesiden og mobilapplikationen, der blev set som komponenter i en brugssituations sammenhæng. På figur 3, kan det Use Case diagram, systemet er bygget op omkring, følges.



Figur 3: Færdig domænemodellen

Da Use Case diagrammet udelukkende afspejler mobilapplikationen og hjemmesiden, blev det valgt, at to primære aktører ville være repræsentanter for hele systemet; Brugeren, som tilgår mobilapplikationen, og administratoren, som tilgår hjemmesiden. Databasen og simulatoren kunne ses som sekundære system aktører, men dette er dog blevet fravalgt, da i en brugssituation vil det være brugeren fuldstændigt irrelevant hvorfra dataen kommer. Ud fra dette diagram blev der udviklet en kravspecifikation, som beskriver de funktionelle krav, systemet skal opfylde. Dette dokument er blevet set som agilt, da der ikke var nogen ekstern kunde, som kunne fastsætte nogen krav, og derved var det op til gruppen, hvordan systemet skulle udhvikles. Dette har haft den betydning for systemet, at det under processen har været nødvendigt, at foretage visse ændringer i dokumentet, dog ikke uden, at det er blevet grundigt diskuteret i gruppen. Disse udviklinger blev samtidigt afspejlet i Use Case diagrammet, som blev ændret i sammenhæng med kravændringer. Dette har medført, at Use Case diagrammet og kravspecifikationen er blevet samlet til et dokument, der følger systemets udvikling meget godt.

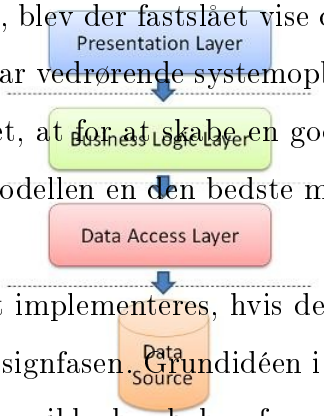
Til kravene sat i kravspecifikations dokumenter, er der blevet udviklet en acceptestsspeci-

fikation, som beskriver, hvordan en eventuel kunde kan teste, at de stillede krav er opfyldt. Dette dokument har også været agilt, i og med det har ændret sig sammen med kravspecifikationen. Ved fuldent implementering af systemet, blev testene specificeret i dette dokument fulgt af gruppens ad-hoc kunde.
Under implementeringen blev det fundet meget brugbart at bruge simple sekvensdiagrammer, til at analysere kodens forløb. Disse har dog kun været midlertidige, da de er blevet tegnet på en tavle. Dette har givet det gruppemedlem, som ikke implementerede denne del af systemet, mulighed for at følge det andet gruppemedlems tankeforløb. Til de mere komplekse dele af systemet er der dog blevet udhviklet mere detaljerede sekvensdiagrammer. [1]

Analyse- og specifikations arbejdet har fungeret rigtigt godt, da den konstante kommunikation mellem gruppens medlemmer har sikret, at der var et bibeholdt overblik over processen. Dette sikrede desuden også, at funktionaliteter aldrig blev overset.
Opdelingen af systemet i domænemodellen, samt underopdelingen i Use Cases, gjorde det nemt for gruppens medlemmer at vælge en opgave og fuldføre den. I denne sammenhæng er der aldrig blevet gjort redudant arbejde, da grænsefladen mellem domænerne var klare.

6 Designprocessen

Da først udkast til kravspecifikationen var skrevet, blev der fastslået vise designregler der skulle opretholdes igennem systemet. Den første var vedrørende systemopbygningen af de forskellige komponenter. Det blev hurtigt vedtaget, at for at skabe en god kodestandard og en overskuelige implementering, var tre-lags modellen en den bedste metode. (se figur 4)



Figur 4: 3-Tier model

Tre-lags modellen er simpel og solid, og kan nemt implementeres, hvis der arbejdes efter det, hvilket er tre punkter, der vægtede tungt i designfasen. Grundidéen i tre-lags modellen er, at opdele systemet i ufhængige moduler, som ikke har behov for, at kommunikere kompleks mellem sig. Dette vil skabe høj samhørighed, samt lav afhængihed, hvilket er bestræbelser der forekommer i et hvert IT-system.[2]

[1] Sekvensdiagrammer kan ses i kontekst i systemarkitektur dokumentet under *5: Logisk view, 6: Process/Task view og 8: Implementerings view*

[2] En detaljeret model af systemerne kan følges i systemarkitekturen, under *5: Logisk View og 8: Implementerings View*

Også i sammenhæng med høj samhørighed og lav afhængihed, blev det vedtaget, at mobilapplikationen aldrig måtte kommunikere direkte med databasen, samt at alle udregninger skulle ske server-side. Det resulterede i en applikation som ikke udfører noget datamanipulations arbejde, men kun henter, sætter og gemmer data. Desuden kan applikationen nemt skiftes ud, hvis den, for eksempel, skal supporteres på en anden platform, hvilket betyder lav afhængighed mellem system komponenterne.
For sikkerhedsmæssige årsager, blev det også vedtaget, at forbindelsenbeskrivelsen til databasen skulle gemmes væk.

I datamæssigt sammenhæng, blev det først undersøgt, hvorvidt det var muligt at tilgå reelle informationer om busser, det blev dog hurtigt etableret, at dette ikke var en mulighed. Dette var grundstenen til, at simulatoren blev udviklet. Simulatoren gav gruppen mulighed for at teste de dele af systemet, hvor informationer om en bus var en nødvendighed, uden en reel bus skulle tilgås. Dette gjorde altså projektet uafhængig af hvilken kilde, informationen kommer fra, og derfor også med til at skabe lav kobling.

Rutemæssigt blev der gjort en del overvejelser. Projektet blev i først omgang specificeret til, at samtlige stoppesteder på en ruten (dvs. stoppesteder i begge retninger), skulle kunne vælges, således at brugeren kunne tilgå den enkelte retnings stoppested. Dette blev hurtigt genovervejet, da det ville resultere i en uoverskuelig rute, med to gange så mange stoppesteder som der, reelt set, var behov for. Derfor blev det vedtaget, at der kun skulle vælges et stoppested, og tiden for den nærmeste bus, i begge retninger, ville blive vist.
I samme sammenhæng blev det også overvejet, at en rute sagtens kunne have mere end en endestation, og det var derfor nødvendigt også at tage højde for disse typer ruter.[3]

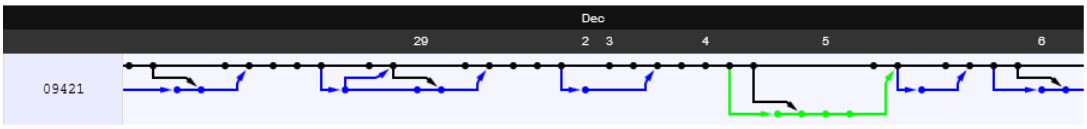
Under hele processen er designløsninger og -beslutninger blevet diskuteret imellem medlemmerne af gruppen, hvilket har resulteret i et yderst fleksibelt produkt.

[3]Se afsnit *9: Data View* i systemarkitekturen, for mere information om dette.

7 Udviklingsværktøjer

Under udførslen af projektet er der blevet gjort brug af dele af Scrum, men da gruppens medlemsantal kun er to, blev det ikke set som nødvendigt at tage et scrum værktøj i brug. Der har været konstant kommunikation mellem gruppemedlemmerne, da der udelukkende er blevet arbejdet sammen. Under skriveprocessen af kravspecifikationen, acceptte-sten, system arkitekturen og dette dokument, er samtlige underafsnit blevet skrevet på et scrum-board, hvorpå samtlige gruppemedlemmer kan melde sig på en opgave.
Under udviklingen af systemet, blev der gjort brug af ansvarsområder, hvori der var klare linjer for, hvem der skulle udvikle hvilke dele.

Til versionsstyring er GitHub taget i brug, som er en versioncontroller, der implementerer Git. Dette er et standard versionsstyrings system, hvori medlemmer kan tilføje, fjerne eller ændre filer, hvorefter andre medlemmer kan tilgå ændringerne. Når et system af denne størrelse imlementeres, genereres der ofte en del unødvendige filer. Git implementerer muligheden for, at kunne ignorere disse filer igennem en ignore protocol. Dette har specielt været nyttigt under udviklingen af mobilapplikationen, samt ved skrivningen af projektes dokumenter, da begge dele auto-genererer en del unødvendige filer, ved bygning. GitHub, og dermed versionstyrings redskaberne, tilgås igennem et commandline værktøj, kaldet Git Shell, som bygger ovenpå Windows Powershell. Heri kan der navigeres til den mappe, hvor repositoriet er oprettet og herfra addes, commites, deletes og merges, præcis som et normalt versionsstyrings værktøj. Repositoriet kan også tilgås online, hvori versionstyrings historikken kan tilgås. Dette har været nyttigt i den sammenhæng, at et tidligere commit nemt kunne hentes ned, hvis to versioner skulle holdes op mod hinanden. Commit og merge træet kan også følges. På figur **??**, kan en del af dette træ ses. Den sorte linje repræsenterer masteren, eller trunken. De blå og grønne linjer repræsenterer en lokal branch, hvori mere end et commit er blevet tilføjet. Når den sorte linje peger ned på en af de blå eller grønne, betyder det at et merge er blevet udført. Prikkeren på de forskellige linjer repræsenterer et commit.



Figur 5: Git commit historik

Alt tekstredigering er foregået i *LaTeX*, hvilket har været meget brugbart, da større dokumenter kunne opdeles i forskellige filer, som kunne arbejdes på individuelt. Dette har tilladt, at begge gruppemedlemmer kunne redigere samme dokument uden der opstod konflikter. Desuden sikres, at de nyeste billeder og lignende altid er opdateret, da *LaTeX* linker til disse i stedet for at indsætte dem direkte.

Alle diagrammer er blevet udviklet i Microsoft Visio 2013, da dette havde indbygget værktøjer til alle de diagrammer, der blev set som nødvendige. Den eneste undtagelse er Use Case diagrammet, som blev udhviklet på hjemmesiden `www.creately.com`. Visio understøtter "stencils", hvori forskellige komponenter relevant for det givne diagram, kan tilføjes. Dette gjorde det yderst nemt at arbejde med.

8 Resultater

Nedenfor er de mest væsentlige resultater listet, hvorefter de kort beskrives. Der refereres til accepttestspecifikationen, hvis det ønskes at få et større overblik over de udførte tests.

Overordnede resultater

- Få en præcis tid til bus ankommer til stoppested.
- En måde at administrere busruter.
- relationel database, til persistering af data.
- Simulator af bus.

?

?

?

Simulator

Det er muligt at simulere en eller flere busser der køre på en busrute.

9 Projektets fortræffeligheder

Nedenfor er en række funktionaliteter og arkitekturer i projektet, som gruppen er specielt stolt af, listet og beskrevet.

Mobil applikation

Da mobil applikationen var den vigtigste del af projektet, blev der sat stort fokus på at få bygget den op med godt design. Der blev også sat fokus på at få gjort applikationen brugervenlig og intuitiv for brugeren. Det blev opnåede ved at brugeren skal fortage så få klik som muligt for at få vist det information han ønsker, samt ved at gøre det tydligt at de forskellige knapper bruges til.

WebService

Der blev hurtigt fastslået at der ikke skulle ske meget arbejde på mobil telefonen, dette blev løst ved at der blev udviklet en webserive. Dette har gjort det muligt at udføre tunge udregnigner på en server i stedet for på telefonen. Desuden gør den det muligt at nemmere at kunne udvikle lignende applikationer til andre mobil styresystemet så som IOS og Windows Phone.

10 Forslag til forbedringer af projektet eller produktet

Under udarbejdelsen af et softwareprojekt er det næsten altid plads til forbedringer, både mht. til arbejdsprocessen og selve produktet. Det gælder også for dette projekt. Nedenfor er en række elementer i projektet, som kunne have været optimeret, listet og beskrevet.

GUI - Administrations hjemmeside

Brugergrænsefladen på administrations hjemmesiden kunne være bedre, vis mindsker vinduet på hjemmesiden vil de forskellige elementer ikke længere passe i deres respektive felter. Det kan nemt blive løst ved at lave et nyt .css stylesheet, der vil gøre hjemmesiden til en mere behagelig brugeroplevelse.

Server

Serveren der hoster både administrations hjemmeside, webservicen og MySQL databasen, er en billige, ikke særlig kraftig server og kan derfor, sandsynligvis, ikke håndtere en særlig stort antal samtidige brugere af mobil aooliaktionen. Dog er det nemt at flytte det hele over på en ny server, når det bliver nødvendigt

Administrator login

Det skal være nødvendigt at logge ind på administratorens hjemmesiden før den kan bruges, eller vil det være muligt for hvemsomhelst at tilgå administrations værktøjerne.