CHALMERS

OmniTrace - Spårningssystem för fordon

MATTIAS ANDERSSON
MICHAEL BODIN
JOHAN BÖHLIN
MARCUS HANSSON
LARS-ERIK LIS
MATTIAS NORDQVIST
PATRIK OPACIC

Projektarbete

Civilingenjörsprogrammet för datateknik

CHALMERS TEKNISKA HÖGSKOLA
Institutionen för data- och informationsteknik
Avdelningen för datorteknik
Göteborg 2006

Innehållet i detta häfte är skyddat enligt Lagen om upphovsrätt, 1960:729, och får inte reproduceras eller spridas i någon form utan medgivande av författaren. Förbudet gäller hela verket såväl som delar av verket och inkluderar lagring i elektroniska och magnetiska media, visning på bildskärm samt bandupptagning.

© Mattias Andersson, Michael Bodin, Johan Böhlin, Marcus Hansson, Lars-Erik Lis, Mattias Nordqvist, Patrik Opacic, Göteborg 2006.

Sammanfattning

OmniTrace är ett bevakningssystem för fordon. Systemet gör det möjligt för dig som ägare att ha full kontroll över ditt fordon. Du kan bestämma när fordonet får röra på sig och inom vilka områden. Om dessa restriktioner bryts antas fordonet vara stulet och ett SMS skickas till din mobiltelefon med meddelandet: ”Ditt fordon är stulet!”. Vid stöld kan du via ett användargränssnitt se på en karta exakt var fordonet befinner sig. Denna rapport beskriver i detalj hur spårningstjänsten fungerar, hur systemet är uppbyggt och hur projektgruppen arbetat – från design till färdigt system. Systemet är anpassat för spårning av fordon men dess användningsområden är betydligt fler – vilket framgår av denna rapport.

Abstract

OmniTrace is a sort of “guardduty” service for vehicles. The system makes it possible for you as an owner to have full control over your vehicle. You can decide when the vehicle is allowed to move and which geographical areas it is allowed to be in. If these restrictions are violated the vehicle is assumed to be stolen and an SMS is sent to your mobile phone with the message: “Your vehicle is stolen!”. Upon an event of vehicle theft you can, through a user interface, see the exact position of the vehicle. This report describes in detail how this service works, how the system is built and how the project team has worked – from design to complete system. The system is adapted for tracing vehicles but its usage are many – something that should be clear after reading this report.

Innehåll

1 Inledning....1
1.1 Uppdraget....1
1.2 Syfte och mål....1
1.3 Problemformulering....2
1.4 Rapportens Innehåll....2
1.5 Avgränsningar....3
2 Metodik....4
2.1 Organisation / Arbetsform....4
2.2 Planering....4
3 Systemimplementation....7
3.1 Reflektion....7
3.2 Lösningar....8
3.2.1 Spårkrets....8
3.2.2 Servern....9
3.2.3 Användargränssnittet....10
3.2.4 Administratörsmjukvaran....10
3.2.5 Databasen....11
4 Resultatredovisning....12
4.1 Systemöversikt....12
4.2 Spårkretsen....13
4.2.1 Huvudkortet....14
4.2.2 IR-Fjärrkontrollen....16
4.2.3 GPS-modulen....17
4.2.4 GSM-modulen....18
4.3 Servern....20
4.3.1 OmniServer....20
4.3.2 Användargränssnittet....21
4.3.3 Administratörsmjukvaran....22
5 Verifiering av systemet....23
5.1 Testfall....23
6 Slutsats och diskussion....24
6.1 Produkten....24
6.2 Säkerhetsanalys....25
6.3 Utbyggnadsmöjligheter....25
Appendix A....29
Appendix C....38
Appendix D....39
Appendix E....40
Appendix F....41
Appendix G....42
Appendix H....44

Förord

Detta projektarbete och rapport är ett resultat av en kurs i D-linjens utbildningsprogram som heter *EDA335 - Projektkurs D – DT* och den ingick som ett obligatoriskt moment i utbildningen.

Inför kursstarten fick studenter välja mellan ett antal olika projektförslag. Varav ett, Patrik Opacic och Johan Böhlin lämnade in med titeln *Spårningssystem för bil*, som tilldelades medlemmarna i denna projektgrupp. Projektförslaget har varit utgångspunkten för gruppens arbete och utifrån den formulerades projektuppdraget.

Examinator: Stig-Göran Larsson vid institutionen för Datorteknik på Chalmers Tekniska Högskola.
Handledare: Albert Nummelin

1 Inledning

1.1 Uppdraget

För att minska antalet ouppklarade stölder av fordon kan vi gå till väga på många olika sätt. Man skulle kunna förhindra att stölder av fordon inträffar genom att bygga säkrare låssystem. Det förhindrar dock inte att eventuella tjuvar bogserar bort fordonet. Ett annat alternativ är att bygga ett spårningssystem för fordon så att polisen kan spåra upp fordon som blivit stulna och därmed göra stölder av fordon mindre attraktivt. Ett spårsystem minskar stölder och ökar även andelen uppklarade brott som trots allt uppträder. Nackdelen med ett spårsystem skulle vara att tjuvarna lyckas inaktivera det på något vis, men för att göra det behövs det mer kunskap om systemet vilket väldigt få tjuvar troligtvis har. Vi valde att använda oss av spårsystemförslaget för att lösa problemet.

Tänk dig följande scenario: Din bil är stulen. Du kontaktar polisen och anmäler stölden. Dagar går och du får klara dig utan din bil medan du hoppas att bilen återfinns i gott skick. Problemet är att du inte vet var bilen befinner sig, det tar tid att få tillbaka din bil och värdesaker kan ha försvunnit.

Vår uppgift var att konstruera ett system för att spåra stulna fordon. Tanken är att ägare ska kunna gå in via ett användargränssnitt och se var deras fordon befinner sig. Kravet på systemet illustreras bäst med följande scenario:

- Bilägaren parkerar och larmar sin bil
- Bilen blir stulen
- Bilägaren får ett SMS till sin mobil som talar om att bilen är stulen
- Bilägaren spårar bilens position via ett användargränssnitt

1.2 Syfte och mål

Projektkursen löpte genom hela årskurs 3 och dess mål var att göra studenter vana att arbeta i projekt. Tanken är att vi efter fullgjord kurs ska ha [6]:

- identifierat och formulerat en projektuppgift
- fördelat roller och fastlagt samarbetsprinciper
- sammanställt en tidsplanering
- identifierat kunskapsbehov och sökt information
- tagit fram och värderat lösningsförslag, samt utvärderat lösningar
- genomfört muntliga och skriftliga redovisningar

Målet med detta projektarbete var i första hand att få prova på att arbeta i team och lösa ett gemensamt problem. Det fanns också krav på att projektgruppen följde en viss arbetsmetodik. En projektuppgift skulle formuleras, roller och samarbetsprinciper fastläggas samt tidsplanering och lösningsförslag framställas. All information sammanställdes i en s.k. projektplan. Hela det dokumentet återges inte här, men en del av informationen som är relevant finns återgiven i avsnitten: *2.1 Organisation / Arbetsform* och *2.2 Planering*.

Utöver dessa saker är syftet att bygga ett fungerande spårsystem för fordon som kan installeras på vilket fordon som helst. Ett system som detta att bidra till andelen uppklarade stölder av fordon i samhället och eventuellt minska antalet stölder.

Ett spårsystem för fordon kommer att reducera olösta brott väldigt säkert, då fordon som har systemet kommer att kunna spåras och därmed hittas. Tjuvar kommer att ertappas oftare för utförda stölder av fordon och med detta motiveras till att inte begå fler stölder.

Risken med systemet är att om en säkerhetslucka skulle uppstå så att vem som helst kan se positionen för ett visst fordon. Om en tjuv kunde ta sig in och se att ingen bil är hemma kan denna uppmuntras till att göra intrång i en villa eller annan bostad. Målet kommer då ändå att uppfyllas men vi får samtidigt en möjlig bieffekt av systemet. Men detta kan alltid avhjälpas med hjälp av att kryptera informationen.

1.3 Problemformulering

Problemet i detta uppdrag är att spåra ett stulen fordon. Detta i sig resulterar i en mängd mindre problem:

- Hur delar vi upp arbetet för att få gruppen arbeta effektivt?
- Vilka deadlines ska sättas?
- Hur löser vi konflikter i gruppen?
- Hur många delmoment är klokt att använda sig av?
- Hur erhåller vi fordonets position?
- Hur överför vi informationen om positionen till en säker plats?
- Hur visar vi informationen för rätt person utan att obehöriga får ta del av den?
- Hur stora krav ska vi ha på säkerheten?
- Vilka åtgärder behöver vidtas för att tillfredställa dessa krav?

Dessa är några av de viktigare frågorna att besvara i arbetet för en fungerande produkt i slutändan. Listan skulle kunna göras längre, men vi nöjer oss med detta.

1.4 Rapportens Innehåll

Denna rapport beskriver projektgruppens arbete och spårningssystemets uppbyggnad. Rapporten fungerar som systemspecifikation och kan också användas som en manual. Tanken är att den ska kunna läsas av både användare och utvecklare. Rapporten börjar med att ge en översikt hur systemet är konstruerat. Vidare i rapporten beskrivs de olika delarna i konstruktionen mer ingående. Slutligen presenteras en analys av systemet samt vilka slutsatser som dragits.

Avsnittet 3 *Systemimplementation*, beskriver ingående hur vi gick tillväga när vi konstruerade systemet och hur vi tänkte när vi arbetade med att ta fram vår produkt. Vi nämner vilka problem som uppstått i arbetet, oklarheter som är värda att klargöra och vilka diskussioner av lösningar vi haft. Vi valde att dela upp detta avsnitt för varje del i projektarbetet.

Nästa del i rapporten är 4 *Resultatredovisning,* som beskriver resultatet av vårt arbete. Här beskrivs ingående hur systemet fungerar och tar upp en del tekniska detaljer. Liksom avsnittet *Systemimplementation* är detta uppdelat för varje del i projektet.

Slutligen kommer vi i avsnitten 5 *Verifiering av systemet* och 6 *Slutsats och diskussion,* diskutera hur arbetet gått och vilka problem som uppstått. Vi presenterar vilka testfall vi tagit fram och hur det gick med testerna av projektet .

1.5 Avgränsningar

I detta projekt kommer vi anta vissa avgränsningar för att begränsa projektets storlek. Vi har t.ex. valt att inte använda oss av krypterad dataöverföring. Vi har inte heller försökt integrera delar i fordonet eller använda t.ex. en bils batteri som strömförsörjning. Vidare så kan man inte fjärrstyra ett fordon på något sätt om det stjäls (t.ex. begränsa en bils hastighet). En annan begränsning vi har är att vissa delar inte är särskilt konfigurerbara eller utbyggbara. Att göra systemet helt utbyggbart skulle vara för tidskrävande.

2 Metodik

2.1 Organisation / Arbetsform

När projektgruppen ursprungligen bildades bestod den av sju teknologer: Mattias Andersson, Michael Bodin, Johan Böhlin, Heidi Fisk, Marcus Hansson, Mattias Nordqvist och Patrik Opacic. Heidi Fisk hoppade i tidigt skede av och Lars-Erik Lis tillträde. Gruppen samlades varje vecka för att diskutera och följa upp arbetet. Under mötena bestämdes bl.a. ansvarsfördelning och vilken utvecklingsmodell som skulle användas.

Ansvarsfördelningen såg ut på följande vis: Michael Bodin utsågs till projektledare och Marcus Hansson till vice ordförande. Projektgruppen delades in i en hårdvaru- och mjukvarugrupp. Dessa grupper ansvarade för olika delar av projektet. I mjukvarugruppen ingick Mattias Andersson, Lars-Erik Lis och Mattias Nordqvist. I hårdvarugruppen ingick Johan Böhlin, Marcus Hansson och Patrik Opacic.

Utvecklingsmodellen vi valde definierade i stora drag hur vår arbetsform kom att se ut. Den modell vi valde kallas för spiralmodellen. Den fungerar som så att man designar och konstruerar en enkel prototyp. Efter man konstruerat en prototyp testar man så att den uppfyller de delkrav man ställt innan man fortsätter. Sedan bygger man på prototypen med fler funktioner tills man får en ny prototyp – som man sedan konstruerar och testar. Så håller man på tills man får en färdig produkt som uppfyller alla krav man har. Att jobba utefter denna modell har varit lämpligt av den anledningen att vi haft två delar i vårt projekt som behövde synkroniseras. Svårigheter som uppstår då en del av projektet legat lite före i prototyputvecklingen har vi kunnat lösa genom att ha kvar äldre prototyper som eftersläpande delar kunnat testa mot. I vårt fall blev det just att en del av projektet blev klart mycket tidigare än en annan.

Kommunikationen i gruppen har fungerat bra. Projektmedlemmarna var tilldelade olika ansvarsområden och i de fall man behövde inblanda sig i någon annans område fick man lov att fråga denne person om det gick bra. Vi tror att tack vare detta har många konflikter och problem förhindrats och arbetet har löpt på i stadig takt.

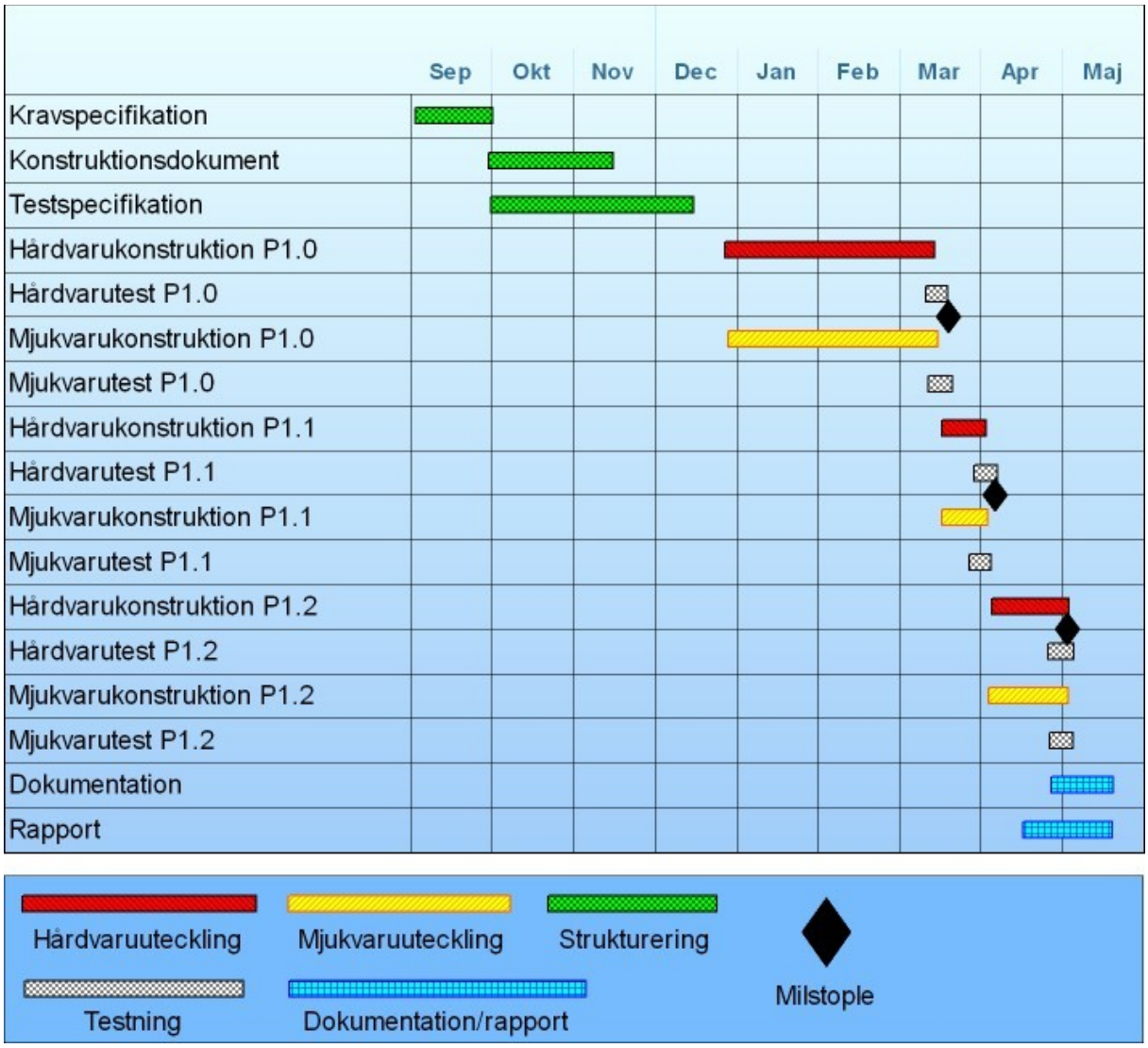
2.2 Planering

Under första halvan av läsåret skedde planeringen av projektet. En projektplan framställdes med följande innehåll:

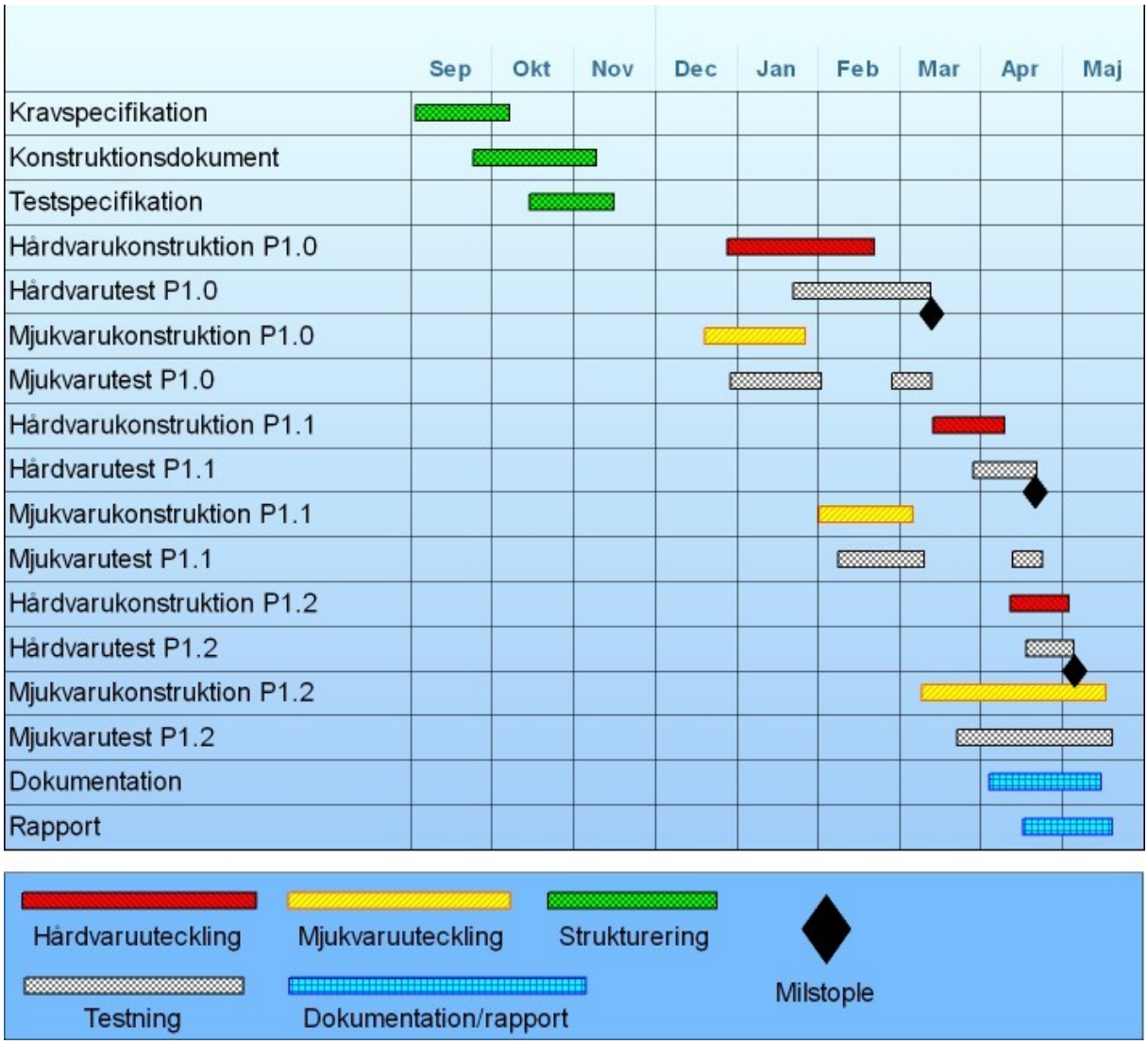
- *Kravspecifikation* – Definition av projektets åtagande och vad vi ville åstadkomma. Här definierade vi också kraven på de enskilda prototyperna.
- *Utvecklingsplan* – Här beskrevs projektets organisation och arbetsform.
- *Konstruktionsdokument* – En grov planering eller ”planritning” över projektet och hur de olika delarna i projektet skulle utformas.
- *Verifieringsplan* – Här stod hur vi tänkt testa om vi uppfyller de krav vi ställt i kravspecifikationen.

Vi delade också upp våra mantimmar så att de skulle matcha den poängindelning vi hade på respektive läsperiod i läsåret. Denna planering visade sig senare vara orealistisk - framförallt

under andra läsperioden, då övriga kurser blev mycket tidskrävande. Vårt första s.k. Ganttschema, som beskriver projektets tidfördelning såg ut såhär:



I början på läsperiod tre gjordes en revidering av Ganttschemat för att få den att passa in i arbetets gång så långt som vi hunnit i projektet. Även den planeringen fick vi lov att revidera och ytterligare ett Ganttschema togs fram i slutet av projektet. Det slutgiltiga schemat ser ut såhär:



När vi jämför de båda Ganttscheman kan vi se hur utdraget allt blir fram till de sista veckorna då allt sker med en gång. Precis som några projektledare från Ericsson förutspådde att det skulle bli.

3 Systemimplementation

3.1 Reflektion

I problemformuleringen ställde vi frågan: Hur bygger vi ett spårningssystem? Vi kom fram till att det fanns behov av att dela upp problemet i flera delar. Det första vi behövde veta var fordonets position då det blev stulet. Vidare behöver vi förmedla denna information till ägaren på något sätt.

För att få ut fordonets position skulle vi kunna använda oss av ett loggningssystem som lagrar information om hur fordonet rört sig och hålla reda på vilken plats det befinner sig på. I en bil skulle vi kunna mäta hur hjulen snurrar samt hur bilen styr och beräknar positionen med hjälp av informationen. Systemet är bra på det sätt att det inte behöver ha kontakt med någon satellit eller annat positionsmedium för att fungera. Däremot, om bilen bogserats bort så kommer inte systemet märka av någon förflyttning. En annan stor nackdel är att bilen måste kalibreras ofta för att systemet ska stämma. Om bilen skulle tappa väggreppet eller sladda på vägbanan innebär detta att systemet måste kalibreras igen. Man kan eventuellt använda sensorer som känner av hur bilen rör sig, genom att kolla på underlaget och beräkna en rörelse och position. Detta är dock lätt för en tjuv att kringgå genom att snabbt och enkelt täcka över sensorn om denne vet vart den sitter. Ett tredje alternativ är att använda GPS systemet som är bra på det sättet att det aldrig behöver kalibreras. Systemet är tillräckligt noggrant för att få ut en hyfsad riktig position på vart fordonet befinner sig. Nackdelen är att det systemet behöver kontakt med satelliter för att fungera, vilket innebär att det slutar fungera om fordonet åker under radiovågsisolerande material.

Det andra problemet var att ta emot och att säkert lagra koordinaterna. Det lagringsutrymme som fanns på spårkretsen var ganska begränsat om vi inte byggde ut det. Det skulle dock bli väldigt krångligt att lagra allt på spårkretsarna i systemet. Nästa steg i problemet, att hämta koordinater, blir också mer komplicerat eftersom vi måste ansluta till spårkretsen hela tiden. Detta skulle innebära stor datortrafik till och från varje GPRS-anslutning, vilket skulle bli kostsamt i längden. Varje spårkrets skulle också kosta mer för varje minnesmodul som vi skulle bli tvungna att montera på spårkretsarna. Vi skulle kunna göra så att alla sändare skickar iväg sina koordinater direkt och lagra dessa på en central enhet i, till exempel, en databas. Detta skulle minska trafiken till och från spårkretsarna, då vi slipper ansluta till dem för att hämta koordinater. Man får eventuellt en snabbare anslutning p.g.a. att bandbredden troligtvis är bättre mellan databasen och slutanvändaren. Hur vi än löser lagringsproblemet kommer vi dock att behöva något som tar emot samt skickar kommandon. Här finns det egentligen inte så mycket alternativ när vi väl valt en lösning på tidigare problem. Det mest naturliga skulle vara att ha en server till detta.

Det tredje och sista momentet i problemet blev att göra dessa koordinater tillgängliga för fordonsägaren. Detta problem vill vi ha en flexibel lösning på, eftersom OmniTrace kan användas till många olika syften. För att uppfylla detta önskemål skulle vi kunna utveckla ett användargränssnitt som kan hämta koordinater och skicka olika kommandon till spårkretsen för att eventuellt kunna larma av eller på. Problemet då är att flexibiliteten kommer att styras av det användargränssnitt vi har gjort till systemet. För att utöka flexibiliteten skulle vi kunna göra ett protokoll för kommunikation, där olika kommandon finns. De som använder systemet får bygga egna lösningar utefter systemets syfte.

Vi har ett överliggande problem i samtliga delar av projektet som rör integritet och säkerhet. Detta diskuteras vidare i avsnittet *6.2 Säkerhetsanalys*.

3.2 Lösningar

För att få ut koordinaterna till ett medium som lagrar dessa valde vi att använda oss av ett vanligt förekommande system som heter GPRS. Detta system används bl.a. av mobiltelefoner för att skicka och ta emot information trådlöst. Systemet har bra täckning och kapaciteten var tillräcklig för vårt syfte och det gjorde överföringar av koordinater smidig. Det finns andra vanliga system i samhället, exempelvis 3G-nätet, som är snabbare men inte alls har lika bra täckning. I detta system prioriterades täckning framför bandbredd p.g.a. att kunna hitta ett fordon var det än befinner sig. Ett annat system är Mobitek, men detta system används mest av uttryckningsfordon såsom ambulanser, poliser, brandkårer och andra enheter inom civilförsvaret. Att använda detta sätt att skicka koordinater på skulle eventuellt belasta mobiteksystemet en hel del om många fordon använde OmniTrace.

För att sedan lagra alla koordinater använde vi oss av en databas kopplad till en server med tillhörande XML-protokoll.

Det sista momentet i problemet bestod av att göra dessa koordinater tillgängliga ägaren. Vi utvecklade därför ett användargränssnitt och ett administratörsgränssnitt. Mer om hur vi designade dessa i rubrikerna nedan.

3.2.1 Spårkrets

Spårkretsens uppgift är att ta fram koordinater för en plats och skicka det till ett ställe där det kan lagras och hämtas av behöriga personer. För att erhålla koordinater använde vi oss av en GPS-modul och för att sedan skicka dem använde vi en GSM-modul. Vi behövde därför ett huvudkort som kommunicerar mellan dessa moduler. Eftersom både GPS- och GSM-modulerna har egna kretskort med anslutningar så konstruerade vi ett eget huvudkort som vi sedan kunde ansluta dem till. På styrmodulen kom vi att behöva någon typ av kontroller som kan ta in information från GPS-modulen, lagra den temporärt och sedan skicka den vidare till GSM-modulen som i sin tur skickar vidare till servern.

För att lösa detta använde vi en mikrokontroller eftersom de är billiga, strömsnåla och de har de har den kapacitet vi behöver. Det finns många olika mikrokontroller på marknaden. När vi valde den vi skulle ha till styrmodulen kollade vi på hur många in-/ut- anslutningar den kunde hantera, hur stort internminne den hade och om vi hade erfarenhet av just den typen av mikrokontroller i projektgruppen.

Vi valde en mikrokontroller från Microchip som heter PIC18F252. Den har en kapacitet som sträcker sig lite utöver det vi behöver. När vi valde den hade vi två krävande funktioner i åtanke: geografiska zoner (förbjudna/tillåta zoner för fordonet) och schemaläggningstider (när fordonet får röra sig eller inte). Dessa funktioner beskrivs mer utförligt i *4 Resultatredovisning*. Vi valde en mikrokontroller som med marginal klarade våra kapacitetskrav.

Vi konstruerade ett huvudkort där vi kunde sätta vår mikrokontroller och andra komponenter på. Vi applicerade tre stycken anslutningar på huvudkortet. En anslutning som vi kunde använda för att koppla in GPS-modulens kretskort, ett för GSM-modulens kretskort och ytterligare en för att kunna omprogrammera mikrokontrollern, t.ex. vid en hårdvaru-uppdatering. Mellan GPS- och programmeringsanslutningen anslöts en switch som bestämmer vilken av dessa två anslutningar som ska användas. Detta gjorde vi för att båda anslutningarna använder samma inport på mikrokontrollern. Därutav kan vi inte programmera om mikrokontrollern samtidigt som vi skickar koordinater till den. För att underlätta felsökningen av mikrokontrollen anslöt vi även en diod som blinkar när datatrafik överförs.

Vi hade även en diod som indikerar om matningsspänningen är på. Samtliga anslutningar använder en annan seriell nivå än vad mikrokontrollen gör. För att den och andra moduler (inklusive datorn) ska kunna kommunicera med mikrokontrollen behövdes det en krets för att översätta dessa logiska nivåer till de nivåer som mikrokontrollen jobbar på och vice versa. Till hjälp hade vi två kretsar: ett till GPS-/datoranslutningen (eftersom de har samma inport på mikrokontrollen) samt ett till GSM-anslutningen. För att ge mikrokontrollen en jämn strömnivå fäste vi på ett par spänningsregulatorer och till dessa ett strömanslutningsdon. En kristall användes för att klocka mikrokontrollen i 20 MHz. Vi kommer att ta upp dessa komponenter mer detaljerat i *6. Resultatredovisning* senare i rapporten

Vi arbetade på detta projekt genom att utveckla prototyper. Spårkretsens utveckling genom prototyperna har oftast varit före utvecklingen av servern. Detta har gjort att vi kunnat testa servern och användargränssnittet mot spårkretsen när det behövts. Första prototypen av spårkretsen var att få den att kunna ansluta till servern samt att kunna skicka koordinater till den. Efter första prototypen var målet att bygga på fler funktioner. I andra versionen av spårkretsen implementerade vi larmaktivering via SMS och ett telefonsamtal. Vi ordnade också så att spårkretsen ansluter till servern och börjar skicka koordinater om den är larmad och samtidigt förflyttas. I tredje versionen utökades den med egen strömförsörjning för att lättare kunna testas på fältet. Vi fick även tid över för att konstruera en fjärrkontroll som kan aktivera larmet genom IR.

3.2.2 Servern

Serverns uppgift är att ta emot de koordinater som mikrokontrollern skickar via GSM-modulen på spårkretsen, spara dem i en databas och göra dessa tillgängliga för behörig person.

För att utveckla den använde vi Java som är ett relativt lätt programmeringsspråk som alla i gruppen behärskar. Det är också bra för att vi inte behöver porta koden till olika system som vi behövt om vi hade programmerat servern i t.ex. C++. Detta gör Java till det smidigaste och den minst avancerade lösningen på serverproblemet.

Vi använder även en databas för att lagra alla koordinater som skickas till servern och i den lagras också information om fordon, registreringsnummer, etc. För mer detaljerad information om vad databasen innehåller, kolla i *Appendix G*.

För att göra all information tillgänglig för behöriga personer har vi implementerat ett XML-protokoll. Med hjälp av detta kan program som är anslutna till servern hämta den information

de efterfrågar utan att olovligt modifiera eller förstöra innehållet. Protokollet finns beskrivet detaljerat i *6. Resultatredovisning*.
I mjukvarugruppen har vi utsett en man per mjukvarukomponent. En har implementerat mest på servern, en har arbetat mer på användargränssnittet och en på databasen samt administratörsmjukvaran. Eftersom det är en del kommunikation mellan komponenterna så behövdes det möten och samarbete för att få arbetet att fungera på ett tillfredställande sätt. Detta komplicerade arbetet och det tog längre tid i många fall.

Serverns prototyper var upplagda så att de skulle matcha spårkretsens prototyper. T.ex. i första versionen då spårkretsen bara skulle ansluta och skicka koordinater och servern ta emot dem samt skriva ut dem på skärmen eller skriva dem till en loggfil. I version två skulle servern lagra alla koordinater i en databas samt via ett XML-protokoll kommunicera med ett eller flera användargränssnitt samtidigt. I tredje versionen implementerades funktioner för hantering av tidsbegränsningar och geografiska zoner.

3.2.3 Användargränssnittet

I OmniTrace-systemet ingår det inte något gränssnitt för klienter. Det kommer beställaren själv att få stå för. Detta på grund av alla olika syften OmniTrace kan användas till. Vi gjorde ett eget exempel på ett användargränssnitt, främst för demonstrationssyfte men även för att verkligen se till att kommunikationen mellan servern och gränssnittet fungerar som den ska. Det som begränsar det användargränssnittet som utvecklas av en viss beställare, är XML-protokollet. I XML-protokollet som beskrivs mer utförligt i *6. Resultatredovisnin*g, bestäms alla typer av kommandon som användargränssnittet kan beordra servern att utföra. Det är därmed det sättet servern och användargränssnittet kommunicerar med varandra.

Eftersom olika personer jobbade på server och användargränssnittet innebar det en risk för problem om kommunikationen inte skulle fungera som den var tänkt.

Prototyperna för användargränssnittet har varit av enklare form. Första versionen var att få systemet att fungera mot servern. Systemet skulle visa på en karta var spårkretsen befinner sig. Andra versionen är att lägga in fler funktioner, t.ex. att kunna lägga in förbjudna eller tillåtna zoner samt att schemalägga de tider fordonet får röra på sig.

3.2.4 Administratörsmjukvaran

Det här gränssnittet är, som namnet antyder, byggt för operatörer. Till skillnad från användargränssnittet så arbetar detta gränssnitt direkt mot databasen, vilket gör att man kommer åt fler funktioner än användargränssnittet. Detta gränssnitt implementerades i PHP. Den som implementerade administratörsmjukvaran strukturerade också databasen, vilket var en bra idé då vi kunde förhindra missförstånd samt effektivisera arbetet lite.

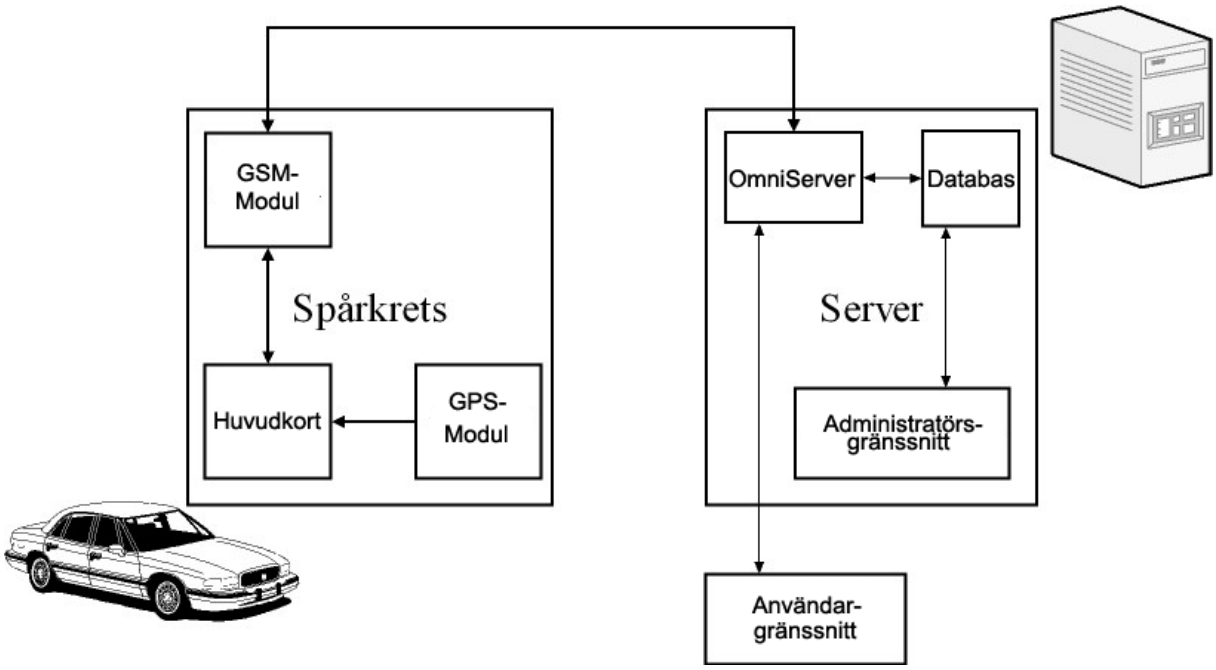
I början av utvecklingen av administratörsmjukvaran så var vi först tvungna att sätta upp en utvecklingswebbserver i form av en PC som kör FreeBSD, Apache 2 och PHP 4.
Sedan implementerades ett ramverk av funktioner som beskrivs nedan. Alla delar av gränssnittet utvecklades utifrån dessa funktioner. Detta gav en konsekvent källkod där man, efter att ha tittat på en del av gränssnittet, snabbt förstår hur andra delar fungerar. Mer detaljerat om detta i resultatredovisningen.

3.2.5 Databasen

Vi använde MySQL som databas eftersom den är snabb och gratis. När vi designade databasen utgick vi från några grundläggande saker vi ville att den skulle innehålla och byggde på därifrån. Den var inte uppdelad i olika prototyper utan utvecklades utefter de krav som vi hade på informationslagring. I början var den väldigt enkel då den bara lagrade koordinater, tid och fordonsinformation. Längre fram i projektet fick den även lagra diverse flaggor samt olika kommandon eller åtgärder som vi ville utföra när de olika flaggorna sattes. Dessa ändringar uppkom då vi började utveckla administratörsmjukvaran. Eftersom administratörsmjukvaran endast arbetar gentemot databasen behövde den kunna ge kommandon för att t.ex. larma på och av spårkretsen.

4 Resultatredovisning

4.1 Systemöversikt



Figur 1 – Systemöversikt

Spårningssystemet består av två huvuddelar (Figur 1): spårkretsen och servern. Spårkretsen sitter i fordonet och har som uppgift att skicka aktuell tid, hastighet och position till servern när fordonet blivit stulen. Servern har som uppgift att administrera över spårkretsarna, logga de spåruppgifter som spårkretsen skickar och presentera dessa för ägaren via ett användargränssnitt.

De tekniska krav som ställdes på systemet var att systemet skulle fungera felfritt under normala väderförhållanden och klara av att göra automatisk omstart vid mjukvarufel. Systemet skulle vara modulbaserat, utformat så att avgränsningarna är tydliga och vara platformsoberoende i den mån det gick. Ur integritets- och säkerhetssynpunkt så var det viktigt att endast behöriga personer fick tillgång till spårningsinformationen. Vidare så skulle ägaren på olika sätt kunna bestämma när och hur spårningen aktiveras. Aktiveringen skulle kunna ske på följande sätt:

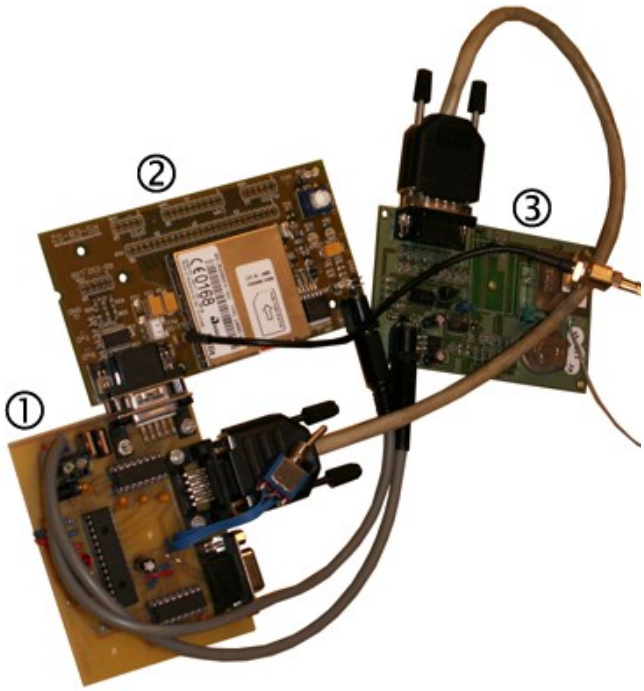
- Fordonet lämnar en angiven geografisk zon
- Ägaren skickar ett SMS till ett visst nummer
- Fordonet rör på sig medan den är larmad
- Fordonet rör på sig under en viss tidsperiod

Om vi utgår från scenariot i inledningen kan vi stegvis förklara hur systemet fungerar. Det börjar med att *ägaren parkerar och larmar sin bil*. Larmningen sker i detta fall via en IR-fjärrkontroll som sitter på bilägarens nyckelring. Spårkretsen ändrar sin status till larmad. Nästa händelse är att *bilen blir stulen*. Spårkretsen tar hela tiden emot positioneringsdata via GPS och märker att bilen rör på sig. Spårkretsen skickar GPS-data till servern tillsammans med en indikation som talar om för servern att spårkretsen är stulen. Servern loggar GPS-datan den tar emot i en databas och ser till att *ägaren får ett SMS till sin mobil som talar om*

att bilen är stulen. Ägaren kan via ett användargränssnitt ta fram uppgifter om bilens position ur databasen och se på en karta bilens exakta position.

Det finns ytterligare några scenarion som visar hur systemet är tänkt att fungera. Ett sådant scenario är t.ex. att ägaren glömmer att larma sin bil eller att tjuven kommer över bilnycklarna och avlarmar bilen. I så fall kan spårkretsen larmas genom att ett SMS skickas till den. Vidare stödjer systemet s.k. geografiska zoner - om bilen åker utanför eller in i en viss zon anses den stulen. Det finns också möjlighet att förbjuda bilen att röra sig under vissa definierade tidsperioder.

4.2 Spårkretsen

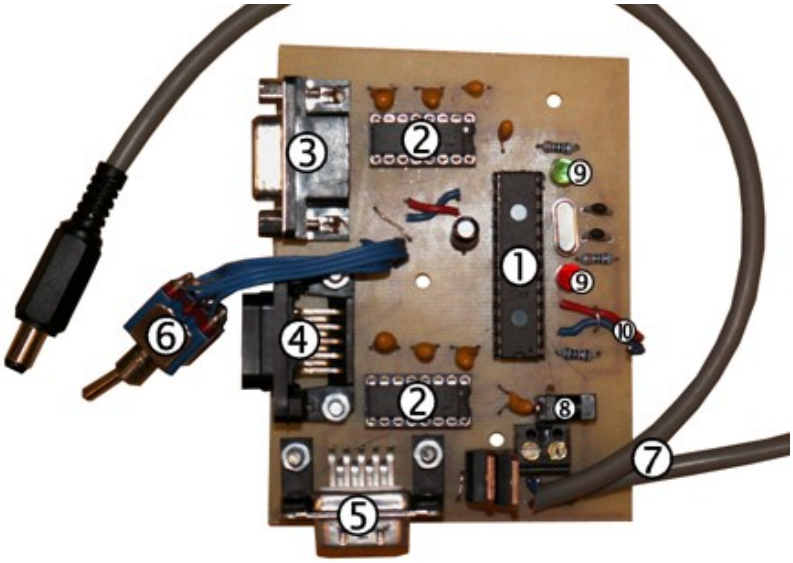


① Huvudkort
② GSM-modul
③ GPS-modul

Figur 2 – Spårkretsens tre moduler

Spårkretsen (Figur 2) består av tre olika delar: GPS-modulen, GSM-modulen och huvudkortet. Sladdarna på bilden är serie-/strömkablar och antenner. GPS-modulens funktion är att tillgodose huvudkortet med information om tid/datum, fordonets position och hastighet. GSM-modulen används för trådlös kommunikation med servern och ägarens mobiltelefon. Den har möjlighet att skicka och ta emot data via GSM nätet samt upprätta en TCP-anslutning med servern. Huvudkortet är hjärtat i spårkretsen och sköter allt från initiering och konfigurering av GPS- och GSM-modulen till dataöverföring av spårningsinformation till servern.

4.2.1 Huvudkortet



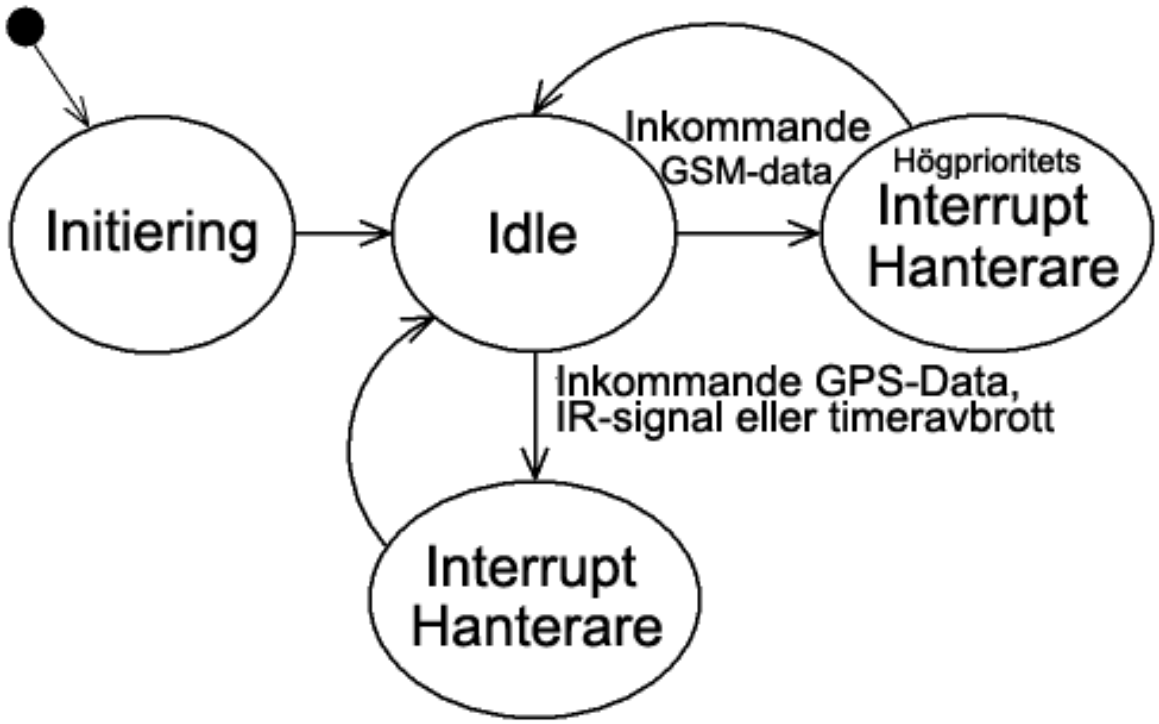
① Mikrokontroller (PIC18F252)
② RS-232 ↔ TTL omvandlare (MAX232)
③ Anslutningskontakt för dator
④ Anslutningskontakt för GPS-modulen
⑤ Anslutningskontakt för GSM-modulen
⑥ Switch för att växla mellan anslutningskontakt ③ och ④.
⑦ Matningsspänningssladdar till GPS- och GSM-modulen.
⑧ Anslutning för matningsspänning
⑨ Operations- och larmindikator
⑩ Resetkontakt

Figur 3 – Huvudkortets komponenter

Huvudkortet (Figur 3) utgör hjärtat av spårkretsen och består av ett antal olika komponenter. Den viktigaste och mest centrala komponenten är mikrokontrollern ① - ett integrerat chip som innehåller CPU, RAM, I/O-portar och timers. Mikrokontrollern är programmerad för att styra kretsen för att få den funktionalitet vi önskar. Huvudkortet ansluts till GPS- och GSM-modulen via anslutningskontakter ④ och ⑤. Komponent ② används för att omvandla mellan kontrollerns respektive modulernas/datorns logiska nivåer. Matningsspänning ansluts till kontakt ⑧ och matningsspänningssladdarna ⑦ kopplas till GPS- respektive GSM-modulen. Vidare så används anslutningskontakt ③ för att ladda ny mjukvara i kontrollern. Se kopplingsschemat i *Appendix D* för en detaljerad beskrivning av huvudkortet. Resten av avsnittet ägnas åt att beskriva den mest intressanta delen av huvudkortet – kontrollern där vi visar hur vi programmerat den.

Den mikrokontroll vi använder heter PIC18F252 och tillverkas av Microship™. Den klockas i 20 MHz och har en inbyggd hårdvaru-USART som sköter seriekommunikation över en serieport [2]. Kontrollern innehåller också mycket minne relaterat till våra behov. Se avsnitt *Bibliografi* för länk till mikrokontrollerns datablad.

Det finns flera sätt att programmera kontrollern: BASIC, C, Assembler språk, osv. Vi valde att använda en variant av C som kallas C18. Som utvecklingsmiljö använde vi MPLAB – ett windowsbaserat gratisprogram från Microchip. Nedanstående tillståndsdiagram representerar mikrokontrollerns dynamiska beteende så som det är definierat av vårt program.



Figur 4 – Mikrokontrollerns tillståndsdiagram

Under initieringen skickar mikrokontrollern initieringskommandon till GSM-modulen (se *Tabell 5* under avsnitt 4.*2.4 GSM-modulen*). Hög- och lågprioritets interrupthanterare sätts. Huvudtimern sätts. Mikrokontrollerns status sätts till OLARMAD (indikerar om fordonet är larmat), EJ STULEN (indikerar om fordonet är stulet), ORÖRLIG (indikerar om fordonet är i rörelse) och GEOZONECHECK (indikerar om geozoner finns definerade). Kontrollern går sedan in i ett idle-tillstånd där den helt enkelt väntar på att någon av hög- eller lågprioritets interrupten ska avfyras.

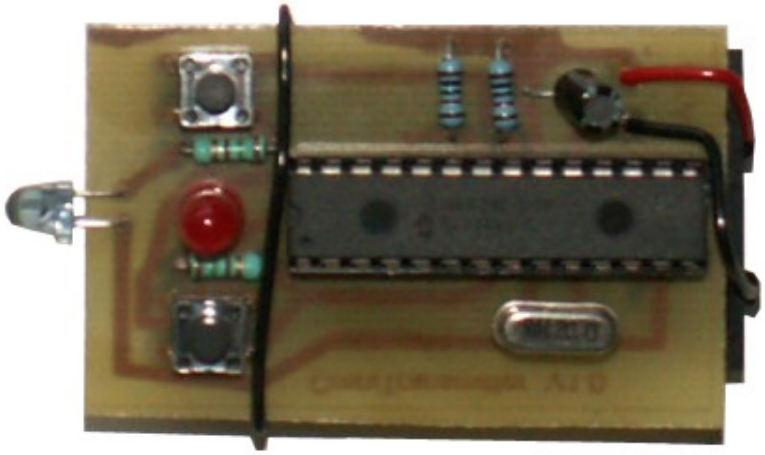
Högprioritetsinterrupten avfyras vid inkommande data från GSM-modulen. Kontrollern går då in i hanteringstillstånd som läser data från GSM-modulen och sätter en lågprioritetsinterrupt-timer som avfyras efter en viss tid då vi kan vara säkra på att all data från modulen har lästs in. Programmet återgår till idle-tillståndet.

Lågprioritets-interrupten avfyras vid inkommande GPS-data, IR-data eller timeravbrott. Mikrokontrollern går då in i ett hanteringstillstånd som kollar vilken typ av interrupt som inträffat och hanterar det:

- GPS-data – kontrollern läser data från GPS-modulen och om ett helt paket mottagits så kollar vi om fordonet rör på sig och sätter i så fall kontrollerns status till RÖRLIG. Programmet återgår sedan till idle-tillstånd.
- IR-data – data läses från IR-mottagaren och om korrekt data mottagits så sätts kontrollerns status till LARMAD. Programmet återgår till idle-tillstånd.
- Timer – kontrollern kollar om programmet är OK annars resettas det och går in i initiering-tillstånd. Om det är dags att skicka spårdata, ett nytt GPS-paket finns redo att skickas och kontrollern har status STULEN (eller om GEOZONECHECK är satt) så skickas GPS-paketet till servern tillsammans med kontrollerns status. Programmet återgår till idle-tillstånd.

- GSM-timer - kontrollern kollar vilken data som mottagits från GSM-modulen och hanterar det på lämpligt sätt.

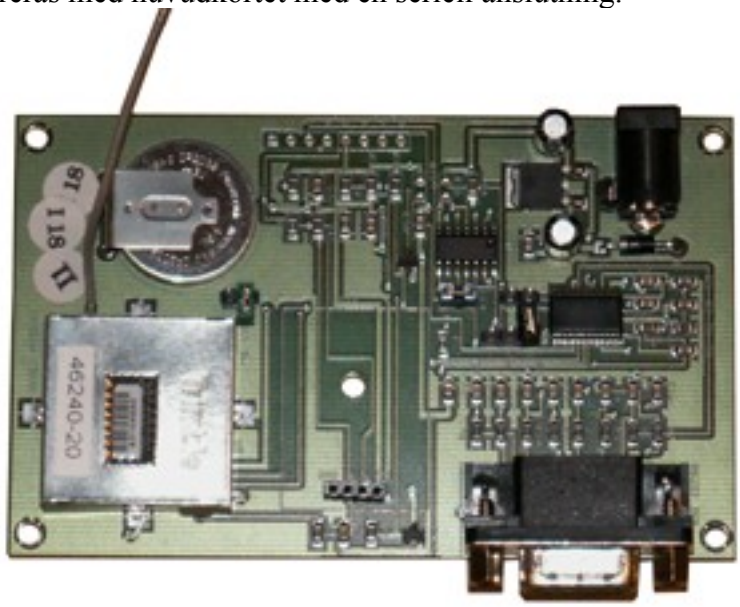
4.2.2 IR-Fjärrkontrollen



IR-Fjärrkontrollen används helt enkelt för att larma/avlarma spårkretsen. När en knapp trycks ner på dosan skickas en larma/avlarma-signal till en mottagare på huvudkortet. Varje IR-kontroll har en unik kod för att skilja dem åt.

4.2.3 GPS-modulen

GPS-modulens funktion är att tillhandahålla huvudkortet med information om aktuell tid/datum samt spårkretsens position och hastighet. Vi valde att använda *Lassen™ iQ GPS Receiver Starter Kit* som uppfyller våra olika driftkrav och som dessutom har fördelen att lätt integreras med huvudkortet med en seriell anslutning.



Starter-kittet gör det enkelt att testa, konfigurera och integrera GPS-mottagaren tillsammans med huvudkortet. Kittet innehåller bl.a. ett 3.6V batteri som förser mottagaren med backupström och en RS232 seriell anslutning. Det ingår dessutom strömkablar, antenner, en adapter och mjukvara för att testköra GPS-mottagaren.

Själva mottagaren heter *Lassen IQ GPS Receiver* är en fullfjädrad GPS-mottagare med alla de egenskaper man kan önska sig. Den är liten (26x26x6 mm) och drar lite ström (mindre än 89 mW) [3]. Under normal operation så sänder modulen ut komplett position, hastighet och tid i vad som kallas *Trimble TSIP Binary Protocol*.

TSIP datapaketen har följande format: <DLE><ID><DATA><DLE><ETX>, där:

- <DLE> är byte 0x10
- <ETX> är byte 0x03
- <ID> är en byte som identifierar typ av paket som skickas
- <DATA> är tillhörande data som skickas med paketet

Datan skickas med mest signifikant byte först. Nedanstående tabeller visar de relevanta paket som GPS-mottagaren skickar och vilket dataformat de har. Notera att de datafält i paketen som inte varit intressanta för oss har utelämnats ur tabellerna. Datatyperna Single (float eller 4 bytes REAL) samt INT16 (16 bitars integer) följer *ANSI/IEEE Std. 754* standarden.

Tabell 1 - GPS Time (Report Packet 0x41) – Skickas varje 5:e sekund

Byte	Beskrivning	Datatyp	Enhet
0-3	GPS tid i veckan	Single	Sekunder
4-5	Extended GPS veckonummer	INT16	Veckor
6-9	GPS UTC offset	Single	Sekunder

Tabell 2 – Position (Report Packet 0x4A) – Skickas varje sekund

Byte	Beskrivning	Datatyp	Enhet
0-3	Latitud	Single	Radianer; + för norr, - för syd
4-7	Longitud	Single	Radianer; + for öst, - for väst
8-11	Altitud	Single	Meter

Tabell 3 – Velocity (Report Packet 0x56) – Skickas varje sekund

Byte	Beskrivning	Datatyp	Enhet
0-3	Östlig Hastighet	Single	m/s; + för öst, - för väst
4-7	Nordlig Hastighet	Single	m/s; + för norr, - för syd
8-11	Hastighet Uppåt	Single	m/s; + för upp, - för ner

4.2.4 GSM-modulen

GSM-modulen har som uppgift att möjliggöra trådlös kommunikation mellan spårkretsen och servern eller ägarens mobiltelefon via GSM-nätet. Vi valde att använda *Telit GM862 Family EVK Quick Start* – ett kitt som möjliggör enkel integrering mellan huvudkortet och valfri *Telit GM852 Family* GSM-komponent.

Kittet gör det enkelt att komma åt alla funktioner som vi behöver. Kittet innehåller en RS232 seriell anslutning, strömkablar, antenner, en adapter och mjukvara för att testköra GSM-komponenten [1].

Själva GSM-komponenten vi valt heter *Telit GM862-GPRS module* och möjliggör trådlös dataöverföring med GPRS (General Packet Radio Service). Komponenten används för att ta emot SMS från ägaren och upprätta TCP-anslutning mellan spårkretsen och servern. GSM-komponenten styrs via serieporten med s.k. AT-kommandon [5] som består av en sekvens ASCII-tecken följt av <CR> (carriage return). Nedanstående tabeller visar de relevanta AT-kommandon som används i vårt system när komponenten konfigureras och initieras, då en trådlös anslutning ska upprättas och när SMS hämtas.

Tabell 4 – Konfigurering

Kommando	Beskrivning
AT#SKTSET=0,<port>,<host>	Ställer in port- och host-parameter som används vid TCP-anslutning till fjärrdator.
AT#USERID=<användarnamn>	Ställer in användarnamn för GPRS-anslutningen mot operatören.
AT#PASSW=<lösenord>	Sätter lösenord för GPRS-anslutningen mot operatören,
AT+CGDCONT=1, ”IP”, <host>, “0.0.0.0”, 0, 0	Ställer in operatörens GPRS-server.
AT#SKTTO=<tid>	Sätter socket inaktivitet timeout (angivet i sekunder)
AT#SKTSAV	Sparar ovanstående inställningar permanent i modulen.

Tabell 5 – Initiering

Kommando	Beskrivning
+++	Kopplar ner tidigare TCP-anslutning ifall sådan finns.
AT+CMGF=1	Ställer in att SMS levereras i ASCII-format.
AT+CNMI=2,0,0,0	Talar om för GSM-komponenten att notifiera när nytt SMS anlänt.
AT+CLIP=1	Talar om för GSM-komponenten att skicka ut telefonnumret från inkommande samtal.

Tabell 6 – Svar från GSM-komponenten

Kommando	Beskrivning
OK	Talar om att mottaget kommando är OK
RING	Skickas vid varje signal vid inkommande samtal
+CLIP: ”<nummer>”, <typ>, 128, ””, 0	Skickas efter varje signal och indikerar vilket nummer det inkommande samtalet är ifrån.
+CMTI: ”SM”, <plats>	Indikerar att ett nytt SMS är mottaget och vilken minnesplats det är sparat på
+CMGR: ”<status>”, ”<nummer>”, ”<datum>”, <NL><meddelande>	Svar när ett SMS efterfrågas. Visar om SMS:et är oläst eller inte, vilket nummer det är ifrån, datum då SMS:et mottogs följt av innehållet i SMS:et
CONNECT	Svar när en GPRS-anslutning blivit upprättad.
ERROR	Svar när felaktigt kommando tagits emot.
NO CARRIER	Svar när GPRS-anslutning avbrutits.

Tabell 7 – SMS och Anslutning

Kommando	Beskrivning
AT#SKTOP	Skapar en GPRS-anslutning mot servern.
AT+CMGD=<plats>	Tar bort SMS på angiven minnesplats.
AT+CMGR=<plats>	Hämta SMS på angiven minnesplats.
+++	Koppla ner GPRS-anslutning.

4.3 Servern

Servern har flera olika uppgifter varav några av dessa är att logga spårinformation som skickas från spårkretsen, larma spårkretsen när vissa villkor är uppfyllda och att administrera spårkretsarna och användarkonton. Servern består av tre delar: OmniServern, databasen och administratörsgränssnittet. Ett användargränssnitt som presenterar spårinformationen för ägaren har också utvecklats men ingår inte i OmniTrace-systemet.

Vi använde oss dessutom av SMS-tjänsten Pixie. Tjänsten fungerar som så att man kopplar upp mot en server och skickar SMS genom kommandon via XML.

4.3.1 OmniServer

OmniServer är ett javaprogram utvecklat så att den stödjer alla funktioner som spårkretsen och användargränssnittet kräver. Eftersom den är skriven i java kan den köras på vilken plattform som helst med Java Runtime Environment (JRE1.5) installerat.

Programmet är uppbyggt av trådar och vid start körs två olika trådar igång som lyssnar på var sin port. Den ena lyssnar på porten där anslutningar från spårkretsar kommer in och den andra där klienter ansluter. En ny tråd skapas då en anslutning upprättas vare sig det är från en spårkrets eller från en klient.

Servern kan startas med två olika parametrar:

- -f <filnamn> Loggning till fil.
- -d Debugläge, mer information skrivs ut på standard output.

När en ny anslutning ska upprättas kollas IP-numret först upp. Om IP-adressen inte är tillåten så nekas anslutningen och händelsen loggas, annars startas den nya spårkrets eller klienttråden.

När en ny tråd skapats för en spårkretsansluting ligger den bara och väntar på inkommande kommandon. Kommandon består av teckenlösa bytes och listas i tabellerna nedan. I och med att Java inte stödjer teckenlösa bytes har vi använt oss av *short*-variabler. Genom att skicka kommandon och svar från båda håll kommunicerar server och spårkrets på ett smidigt sätt.

Tabell 8 – Kommandon som skickas av spårkretsen

Byte-värde	Kommando
u	Logga in
f	Koordinatpaket
l	Logga ut
]	OK
h	Alarmändring OK
y	Ändring av telefonnummer OK
p	Ändring av lösenord OK
x	Ändring av intervall mellan sändningar OK
j	Ändring av stulenstatus OK

Tabell 9 – Kommandon som skickas till spårkretsen

Byte-värde	Kommando
]	OK

b	Inloggning misslyckades
[	Ogiltigt kommando
n	Inte inloggad
k	Inloggning lyckades
a	Redan inloggad
h	Begäran om ändring av alarmstatus
y	Begäran om ändring av telefonnummer
p	Begäran om att ändra lösenord
x	Begäran att ändra intervall mellan sändningar
j	Begäran att ändra stulenstatus

När alarmet i spårkretsen är på och fordonet rör på sig antas den vara stulen och spårkretsen meddelar servern om detta. Spåkretsen skickar koordinatpaket som innehåller information om tid/datum samt spårkretsens position och hastighet. Servern lagrar denna information i en databas. Vid stöld tar servern också olika åtgärder beroende på vad för alarm som är aktuellt. OmniTrace-systemets standardalarm är att skicka ett SMS till de telefonnummer som finns registrerade för respektive användare. När de sedan blivit uppmärksammade på att fordonet blivit stulet kan de kolla var det befinner sig via ett användargränssnitt. Om användaren via användargränssnittet vill larma sitt fordon och om ingen TCP-anslutning finns till spårkretsen så använder sig servern av SMS-tjänsten. Vi skickar då ett SMS från server till spårkrets och aktiverar alarmet på det viset.

OmniServern kommunicerar också med användargränssnittet men det sker inte riktigt på samma sätt som med spårkretsarna. Den fungerar på samma sätt i den mening att den väntar på inkommande kommandon och därefter svarar. Däremot skickas kommandon till och från server som XML-dokument, vilket medför både för- och nackdelar. Fördelarna är att det är väldigt välstrukturerat och lätt att konvertera till en struktur i java. En nackdel, i kontrast till hur vi skickar data mellan server och spårkrets, är att vi skickar mycket mer data i form av strukturdata som består av taggar. Oftast är dock bandbredden mycket större mellan användargränssnitt och server än mellan spårkrets och server, så i detta sammanhang spelar det mindre roll.

4.3.2 Användargränssnittet

Vi har utvecklat ett användargränssnitt i form av ett klientprogram (se *Appendix B*). Detta klientprogram är, som nämnts tidigare, inte en del av OmniTrace-systemet men vi har gjort en strikt DTD (Document Type Definition), som bestämmer hur de XML-dokument som skickas till server ska se ut. Man kan därför utforma användargränssnittet efter eget tycke så länge som man följer det implementerade DTD-dokumentet. I servern valideras varje inkommande XML-dokument mot denna DTD (se *Appendix A* för beskrivning). Om valideringen misslyckas skickas ett felmeddelande tillbaks till användargränssnittet som talar om vad felet är. Tyvärr kan man inte med DTD specificera att ett element (kommando) ska ha ett visst innehåll beroende på t.ex. ett attributvärde (typ av kommando). Detta hade varit mycket användbart då vi använder oss av attribut för att servern ska veta vad innehållet är. Alltså kan ett ”felaktigt” XML-meddelande som skickas till servern ändå gå igenom valideringen. Dessa fel kollas ändå manuellt i servern och ett felmeddelande skickas.

4.3.3 Administratörsmjukvaran

Medan användargränssnittet används för att spåra fordon, ge statistik, m.m., så behövdes också ett medföljande system för att kunna styra fundamentala funktioner hos servern. Dessa funktioner är exempelvis accessrättigheter i servern, databastillgång, fordonsregister, operatörsregister, etc. Detta har förvandlats till OmniTrace Administrator, en mjukvara som kompletterar server och spårkrets och erbjuder full kontroll över systemets alla inställningar.

För att inte bara göra mjukvaran plattformsoberoende, utan även nåbar från vilken Internetuppkopplad dator som helst, så bestämdes det att administratörsmjukvaran skulle vara helt webbaserad.

Det finns en uppsjö av alternativ i form av programmeringsspråk, bibliotek, webbservrar, osv. för utveckling av webbaserade applikationer. Den kombination som vi tyckte gav bäst avvägning mellan prestanda, säkerhet, stabilitet och utvecklingstid var Apache och PHP, som vi kom att använda vid implementering av mjukvaran.

5 Verifiering av systemet

5.1 Testfall

Efter att vi var klara med spårkretsen och programmerat det mesta på server och gränssnitt så var det dags att testa OmniTrace på fältet. Vi gjorde först ett testprotokoll som täckte de fall vi ville verifiera och tog med det under testningen. Testet började med de mest grundläggande fallen som t.ex. att kretsen sänder och att servern lagrar koordinaterna på ett tillfredsställande sätt. Detta gjordes genom att två personer kontrollerade inkommande koordinater vid en dator medan övriga förflyttade spårkretsen med hjälp av ett fordon. Kommunikationen mellan dessa två grupper sköttes via mobiltelefon. (se *Appendix H* för mer information om testfallen)

Vi fortsatte med att testa mer avancerade fall, såsom att spårkretsen skickar ut koordinater då den rör på sig samtidigt som den är larmad. Dessutom kollade vi om detta gällde även då den blev larmad via andra alternativ än enbart IR-fjärrkontrollen. Efter detta började vi testa servern mer. Vi testade olika funktioner som t.ex. förbjudna och tillåtna zoner och schemaläggning. Vi hade även ett test för att kolla prestationen på systemet, till exempel hur stor felmarginal det har i position och hastighet.

Efter att vi testat systemet kunde vi verifiera att alla testfall fungerade bra. Vi upptäckte ett nytt fall som vi inte räknat med. Spårkretsen kunde inte larmas när den stod stilla, vilket gav upphov till ett problem: Om ägaren glömmer att larma fordonet, det blir stulet och parkeras, finns det ingen möjlighet att aktivera spårkretsen och spåra fordonets position. Detta åtgärdades strax därefter.

När ett användargränssnitt implementerats testades geografiska zoner. Vi simulerade att ett fordon åkte in i en förbjuden geografisk zon, kollade därefter om den rapporterades som stulen. Vi följde då bilen via en karta och vi verifierade att det stämde.

6 Slutsats och diskussion

6.1 Produkten

I det här projektarbetet konstruerade vi ett spårsystem för fordon. Systemet består av två delar – spårkretsen och servern. Meningen med systemet är att kunna spåra stölder av fordon och indirekt förhindra dessa. Spårkretsen i systemet kan i sin tur delas in i tre huvudkomponenter, en GPS-modul, en GSM-modul och ett huvudkort. Huvudkortet styr kommunikationen mellan de andra två modulerna i spårkretsen och kan styra modulerna till att verkställa olika kommandon som kommer från server, mobiltelefon eller IR-fjärrkontroll. För att sköta dessa uppgifter används en mikrokontroller som sitter på huvudkortet.

Servern är utvecklad i Java och upprätthåller anslutningar till spårkretsar och användargränssnitt. Den tar emot och lagrar information om fordonstyp, tid, registreringsnummer koordinater, mm. All information lagras i en databas. För att få ut den relevanta informationen till användargränssnittet används ett XML-protokoll som tillhandahåller funktioner för att ansluta ett användargränssnitt till servern. XML-protokollet används för att göra systemet mer lättanvänt och strukturerat. Man kan fråga sig om dessa funktioner är tillräckliga. Att ha i åtanke är att belastningen på servern. Är för många komplicerade funktioner implementerade belastas server för hårt då många spårkretsar är anslutna till den.

Användargränssnittet, som vi har utvecklat i demonstrationssyfte, är i skrivet i Java. Det andra gränssnittet för operatörer är skrivet i PHP och arbetar mot databasen istället för servern. Att använda PHP är mindre effektivt eftersom kartbilden inte uppdateras i realtid och kommer därmed att bli ineffektivare om vi följer ett fordon som är i rörelse. Java-gränssnittet däremot kommer att kunna visa fordonets rörelse mer i realtid, för varje ny koordinat användargränssnittet får in ändras bara pekarens position på kartan, när pekaren enbart behöver byta plats får vi mycket snabbare uppdateringstid och gör det lättare att följa ett fordon i rörelse. Säkerhet följer dock av att använda ett webbaserat gränssnitt i PHP. Ett webbaserat gränssnitt ligger på ett fast IP-nummer (eftersom det ligger på en webbserver). Detta innebär att vi använda oss av restriktioner av IP-addresser för att förhindra att vem som helst kan ansluta sig till databasen.

Användargränssnittet är inget som ska ingå i den färdiga produkten utan det är något som beställaren själv får skräddarsy efter egna behov med hjälp av XML-protokollet. Vi valde att göra så pga. alla beställares olika krav på användargränssnittet. Detta gör att det är svårt att implementera ett användargränssnitt som passar i alla situationer. Vissa beställare kanske bara ska övervaka sina fordon t.ex. hemtjänsten för att kolla vilken bil som är närmast den behövande patienten. Därför behövs inte alltid alla de olika funktionerna. Om en beställare t.ex. vill använda OmniTrace för att erbjuda sina kunder anti-stöldlösningar och integrera OmniTrace i deras redan befintliga system är det bara att implementera XML-protokollet i deras applikation.

Vi valde att lägga alla avancerade funktioner (geografiska zoner och schemaläggning) i servern istället för i spårkretsarna, eftersom det både kändes mer naturligt och lättare att ha dessa funktioner i servern. Nackdelen med detta är att ett centralt system belastas med väldigt många beräkningar (befinner jag mig i en zon eller inte) pga. alla de fordon som är

uppkopplade och skickar data var femte sekund. En distribuerad lösning skulle inte kräva lika mycket resurser. Detta kan också lösas genom att låta spårkretsen skicka koordinater mer sällan om den befinner sig långt ifrån närmaste zongräns. Att implementera detta är dock inget vi prioriterade högt, och därav uteblev denna funktion.

GSM-modulen måste skicka data till servern för att den ska kunna avgöra fordonets position. Detta är kostsamt pga. all datatrafik. För att reducera denna kostnad skickar spårkretsen enbart data då fordonet rör på sig. Vi kan då även göra så att GSM-modulen sparar ström, men vi uteslöt det eftersom det innebar för stora förändringar på huvudkortet. Vi beräknade månadskostnaden för en person som använder fordonet en timme alla dagar i veckan till 12 kr, såvida fordonet inte skulle bli stulet.

Vissa av dagens bevakningssystem av fordon är mer avancerade än OmniTrace, några av dem styr tom. fordonets funktioner. OmniTrace är ett enklare spårsystem. Det är isolerat från resten av fordonets tekniska utrustning och kan inte påverka fordonet på något sätt om det skulle bli stulen. Spårsystem finns idag oftast bara i nya bilar och kan t.ex. ändra den maximala hastigheten till 20 km/h eller aktivera summers osv. OmniTrace kan byggas ut även om det inte är integrerat i bilens tekniska utrustning. Vi kan t.ex. bygga på en extern bränsleventil som kan begränsa bensinen till motorn om en bil skulle bli stulen. Fördelen med OmniTrace är att vi lätt kan larma på och av spårkretsen på många olika sätt; IR-fjärrkontrollen, ett SMS en ringning, användargränssnittet eller administratörsmjukvaran.

Spårsystemets isolering från bilens övriga system ger fler fördelar, det blir inte beroende av fordon, det vill säga att det kan installeras på alla möjliga fordon, gamla som nya.

6.2 Säkerhetsanalys

Säkerheten är en del i detta projekt som vi har hanterat mycket begränsat. De säkerhetsåtgärder vi har tagit har varit dels att spärra IP-adresser mellan server och spårkrets men också mellan server och de andra gränssnitten. Detta gör att obehöriga inte kan ansluta till serven och skicka felaktig eller modifierad information. Eftersom administratörsmjukvaran är bunden till en webbserver som har ett fast IP kan man begränsa de IP-adresser som får ansluta till serverns databas. Administratörsmjukvaran har inte kryptering implementerat så där finns en stor säkerhetslucka.
En annan stor medveten säkerhetslucka är att skicka kommandon via SMS, detta SMS kan lätt avkodas och det lösenord SMS:et innehåller är då känt.

Ingen kryptering alls förekommer i detta system. Det är en stor brist i säkerheten som måste implementeras om systemet skulle säljas som en produkt.

6.3 Utbyggnadsmöjligheter

Spårkretsen

OmniTrace kan byggas ut till ett väldigt avancerat larmsystem. Systemet utvecklat i projektkursen är bara i barnstadiet och har många uppgraderingsmöjligheter.

- En intressant funktion skulle kunna vara att installera en webbkamera som fotograferar förbrytaren i förarsätet på en stulen bil, det blir en extern modul som inte kommer att ingå i bilens övriga system.
- En annan funktion skulle kunna vara att stryp bränsletillförseln och därmed stanna bilen då den rapporteras som stulen.
- Ett annat exempel är att bygga en enhet till gasreglaget i bilen för att kunna styra hastighet.

Att integrera nya enheter i spårkretsen kräver ingen större ändring. Det som begränsar antalet enheter som kan slutas till spårkretsen är mikrokontrollern som har ett fixt antal anslutningsmöjligheter. Om man ska bygga ut med många externa enheter kan en uppgradering av mikrokontrollern behövas. En annan sak som vi har tagit upp innan är kryptering. Detta skulle bli första prioritet om systemet skulle byggas ut.

Spårkretsen klarar att kopplas till ett bilbatteri så länge spänningen håller sig mellan 9 till 37 volt.

Servern

Bygger man ut servern med för mycket funktioner kan det innebära att den belastas mer eftersom den för att göra mer uträkningar. En uppgradering av server skulle kunna vara att flytta över en del funktioner till spårkretsen, såsom geografiska zoner och schemaläggning. Man skulle också kunna effektivisera beräkningsalgoritmerna för att göra optimera serverns prestanda.

Den största och mest betydande uppgraderingen är krypterad trafik från och till servern. Vi kan även uppgradera servern genom att definiera fler kommandon att skicka till spårkretsen.. Uppgraderingen behövs troligtvis inte förrän man skulle ansluta en extern enhet till spårkretsen.

Administratörsmjukvaran

Eftersom administreringen följer direkt av systemets funktioner så byggs denna mjukvara ut då systemet uppgraderas. Även här är kryptering en självklar uppgradering så att inte obehöriga får access. Källkoden är mycket modulär och som visas i kapitlet Kodbeskrivning det är väldigt enkelt att lägga till nya sidor till gränssnittet. Det är inte heller några större svårigheter att gå in och modifiera koden.

Olika teman skulle kunna läggas till där utseendet kan skräddarsys utan att behöva ändra i källkoden, och man ska kunna ha flera olika utseenden beroende på definierbara kriterier. Dessa teman skulle även gå att modifiera direkt från gränssnittet, där man enkelt kan ändra parametrar, ladda upp nya bilder, osv. Funktionaliteten för att visa olika teman är redan inbyggt i mjukvaran, men det finns ingen funktionalitet för att byta eller redigera teman. Detta kan vara intressant för ett företag som vill ha en egen prägel på utseendet, t.ex. inkludera företagets logga.

Alarmtyper är ett bra exempel på utbyggnad. En alarmtyp skulle kunna vara att spela upp ett ljud i bilen då den blivit stulen och därmed skrämma iväg brottslingen.

Användargränssnittet

Funktionerna till ett eventuellt användargränssnitt bestäms av vad XML-protokollet begränsar det till att göra. Gränssnitten kan se ut hur som helst då det är meningen att beställaren ska skräddarsy ett eget för sitt specifika användningsområde av OmniTrace. Eftersom XML-dokumenten valideras i servern innebär detta att användargränssnittens möjligheter expanderas då servern gör det.

Databasen

Databasen är uppbyggd så att man kan lägga till flera olika åtgärder vid alarm. Detta kan göras enkelt genom att lagra nya åtgärder i tabellen för åtgärder. Detta gör databasen väldigt flexibel och anpassbar om systemet uppgraderas. Om spårkretsen är kapabel att strypa bränsletillförseln kan man definiera ett kommando för det. Sen när administratören lägger till det värdet i åtgärdstabellen kommer spårkretsen hämta kommandot och stanna fordonet.

Referenser

1.
Information om GSM-modulen
http://www.telit.co.it/faq.asp?productid=25

2.
PIC18F252 Mikrokontroller
http://www.microchip.com/stellent/idcplg?IdcService=SS_GET_PAGE&nodeId=1335&dDocName=en010276

3.
Information om GPS-modulen
http://www.roundsolutions.com/pdf/Trimble/Lassen%20iQ_Reference%20Manual.pdf

4.
DTD-guide
http://www.w3schools.com/dtd/default.asp

5.
AT-kommandon
http://www.roundsolutions.com/techdocs/docs/80234st10017a_AT_Descr_GM862-GSM_GPRS_r0.pdf

6.
Kursmål
http://www.student.chalmers.se/sp/course?course_id=4810

Bibliografi

Max232
http://www.elfa.se/pdf/73/730/07302326.pdf

78S10CV
http://www.elfa.se/pdf/73/730/07309560.pdf

IR323
http://www.elfa.se/pdf/75/07520372.pdf

ELIRM8608S
http://www.elfa.se/pdf/75/07520281.pdf

MPLAB® C18 C Compiler - User G (2006-02-08)
http://ww1.microchip.com/downloads/en/DeviceDoc/MPLAB_C18_Libraries_51297f.pdf
http://ww1.microchip.com/downloads/en/DeviceDoc/MPLAB_C18_Users_Guide_51288j.pdf

PIC18F252 datablad.
http://ww1.microchip.com/downloads/en/DeviceDoc/39564b.pdf

Trimble Lassen iQ/SQ Data Sheets Data Sheet Lassen IQ
http://www.roundsolutions.com/pdf/Trimble/Lassen%2BiQ%2Bv9.pdf

Trimble Lassen iQ/SQ Data Sheets Lassen iQ Serial Ports
http://www.roundsolutions.com/pdf/Trimble/Bulletin_Lassen%20iQ_%20Pin%20Connections_23Nov2004.pdf

GM862-GPRS Product Description Guide
http://www.roundsolutions.com/techdocs/docs/80234ST10012a_GM862-Family_Product_Description_issue20.pdf

GM862-GPRS HW User Guide
http://www.roundsolutions.com/techdocs/docs/1vv0300613_GM862-GPRS_Hardware_User_Guide_issue4.pdf

Appendix A

DTD-dokument som validerar inkommande XML till server

```
1.  <!ELEMENT message (login?,get*,update*)>
2.  <!ELEMENT login (user,password)>
3.  <!ELEMENT user (#PCDATA)>
4.  <!ELEMENT password (#PCDATA)>
5.  <!ELEMENT get (latest|(from,to)|from)?>
6.  <!ELEMENT latest (#PCDATA)>
7.  <!ELEMENT from (#PCDATA)>
8.  <!ELEMENT to (#PCDATA)>
9.  <!ELEMENT update (passw|smspassw|nr+|restriction+|alarm)>
10  <!ELEMENT smspassw (#PCDATA)>
.
11  <!ELEMENT passw (oldpassw,newpassw)>
.
12  <!ELEMENT oldpassw (#PCDATA)>
.
13  <!ELEMENT newpassw (#PCDATA)>
.
14  <!ELEMENT nr (#PCDATA)>
.
15  <!ELEMENT restriction (timezone|geozone|(timezone,geozone))>
.
16  <!ELEMENT timezone (#PCDATA)>
.
17  <!ELEMENT geozone (coord+,coord+,coord+)>
.
18  <!ELEMENT coord (latitude,longitude)>
.
19  <!ELEMENT latitude (#PCDATA)>
.
20  <!ELEMENT longitude (#PCDATA)>
.
21  <!ELEMENT alarm (#PCDATA)>
.
    <!ATTLIST get
22  type (coordpackets|notificationnumbers|activationnumbers|restrictions|alarm)
.   #REQUIRED
    coordpackets-type (latest|interval|from) #IMPLIED>
    <!ATTLIST update
23  type (passw|smspassw|notificationnumbers|activationnumbers|restrictions|alarm)
.   #REQUIRED>
24  <!ATTLIST restriction
.   type (allowed|forbidden) "allowed"
    usertext CDATA #REQUIRED
    id CDATA #REQUIRED
```

```
    alarm CDATA #REQUIRED>
    <!ATTLIST timezone
25  type (yearly|monthly|weekly|daily|once) "once"
 .  start CDATA #REQUIRED
    end CDATA #REQUIRED>
26  <!ATTLIST alarm status (on|off) "off">
 .
```

Rad	Beskrivning till DTD (Inkommande meddelande till server)
1	Root-tagen *message* kan innehålla antingen en *login*, noll eller flera *get* eller *update*. Med andra ord så kan man skicka flera olika kommandon i ett och samma dokument.
2	*login*-tagen används för att logga in i systemet. Inne håller elementen *user* och *password.*
3, 4	*user* och *password*-taggarna innehåller vanlig data.
5	*get*-taggen används för att hämta information. Kan innehålla en *latest*-tag, en *from* och *to*-tag eller en enkel *from*-tag. Dessa används när koordinater ska hämtas från databasen.
6, 7, 8	Inom dessa tre taggar följer antingen en siffra (*latest*), två stycken timestamps (*from,to*) eller bara en timestamp (*from*).
9	En *update*-tag kan antingen innehålla en *passw-tag, smspassw-tag,* en eller flera *nr-taggar*, en eller flera *restriction*-taggar eller en *alarm*-tag. *update* används för att göra uppdateringar i systemet.
10	*smspassw*-taggen används för att uppdatera smslösenordet till spårdosan.
11	*passw*-taggen används när lösenordet för inloggningssystemet ska ändras. *oldpassw* och *newpassw* innehåller gammalt och nytt lösenord.
14	I *nr*-taggen skickas nya nummer, antingen aktiveringsnummer eller nummer som ska ha sms när spårdosan är ”stulen”, till systemet och där görs en kolla så att numret är av rätt format.
15	En *restriction* har antingen en tidszon (*timezone*), både tidszon och geografisk zon (*geozone*) eller bara en geografisk zon definierad.
17	En geografisk zon (*geozone*) måste minst bestå av 3 koordinater.
18	*coord*-taggen innehåller ett longitudvärde och ett latitude-värde
21	*alarm* används för att ändra status på alarmet i spårdosan, detta görs genom att servern skickar ett SMS till spårdosan.
22	*get*-taggen har ett obligatoriskt attribut, *type* som visar vad som efterfrågas. En *get* kan antingen vara att man vill ha position och annan GPS-data (*coordpackets*), telefonnummer som man kan bli notifierad på om ens fordon blivit stulen (*notificationnumbers*), telefonummer som man aktiverar alarmet med (*activationnumbers*), restriktioner (*restrictions*) eller status på alarmet i spårdosan (*alarm*). Om *type* är *coordpackets* ska även attributet *coordpackets-type* vara med i meddelandet.
23	*update*-taggen har likt *get*-taggen ett obligatoriskt attribut *type* som specificerar vad uppdateringen handlar om. En *update* kan antingen vara att man vill uppdatera lösenord till systemet (*passw*), lösenordet som man skickar i ett SMS när man vill aktivera alarmet på spårdosan (*smspassw*), telefonnummer som man ska kunna bli notifierad på om fordonet blivit stulen (*notificationnumbers*), telefonnummer som man ska kunna aktivera alarmet i spårdosan med (*activationnumbers*), restriktioner (*restrictions*) eller alarmstatus (*alarm*).
24	*restriction*-taggen har fyra attribut. *type* som antingen är *allowed* eller *forbidden* beroende på om restriktionen ska vara ett förbud att beträda zonen eller att det ska vara förbjudet att vara utanför zonen. *usertext*-värdet är information om restriktionen. *id* ska vara *-1* om restriktionen är ny och samma id som innan om en förändring gjorts på en restriktion, t.ex. om en zon tagits bort. *alarm*-värdet är det alarm som ska gälla för restriktionen.
25	*timezone*-taggen har tre attribut. *type* som anger om restriktionen ska vara återkommande. *start* är en tidsstämpel då restriktionen och *end* när den slutar.
26	*alarm*-taggen har ett *status*-attribut som antingen är *on* eller *off*.

DTD-dokument som validerar utgående XML från server

1.	*<!ELEMENT message (confirm-error?|confirm-login?|confirm-get*|confirm-update*)>*
2.	*<!ELEMENT confirm-error (#PCDATA)>*
3.	*<!ELEMENT confirm-login (#PCDATA)>*
4.	*<!ELEMENT confirm-get (errormessage|coordpacket+|nr+|restriction+)>*
5.	*<!ELEMENT errormessage (#PCDATA)>*
6.	*<!ELEMENT coordpacket (timestamp,northspeed,eastspeed,altitude,latitude,longitude)>*
7.	*<!ELEMENT timestamp (#PCDATA)>*
8.	*<!ELEMENT northspeed (#PCDATA)>*
9.	*<!ELEMENT eastspeed (#PCDATA)>*
10.	*<!ELEMENT altitude (#PCDATA)>*
11.	*<!ELEMENT latitude (#PCDATA)>*
12.	*<!ELEMENT longitude (#PCDATA)>*
13.	*<!ELEMENT nr (#PCDATA)>*
14.	*<!ELEMENT restriction (timezone|geozone|(timezone,geozone))>*
15.	*<!ELEMENT timezone (#PCDATA)>*
16.	*<!ELEMENT geozone (coord+,coord+,coord+)>*
17.	*<!ELEMENT coord (latitude,longitude)>*
18.	*<!ELEMENT coord (latitude,longitude)>*
19.	*<!ELEMENT latitude (#PCDATA)>*
20.	*<!ELEMENT longitude (#PCDATA)>*
21.	*<!ELEMENT confirm-update (#PCDATA)>*
22.	*<!ATTLIST confirm-login result (ok|error) #REQUIRED>*
23.	*<!ATTLIST confirm-get* *result(ok|error) #REQUIRED* *type (coordpackets|notificationnumbers|activationnumbers|restrictions|alarm) #REQUIRED>*
24.	*<!ATTLIST confirm-update* *result (ok|error) #REQUIRED* *type (passw|smspassw|notificationnumbers|activationnumbers|restrictions|alarm) #REQUIRED>*

25.	*<!ATTLIST restriction* *type (allowed | forbidden) "allowed"* *usertext CDATA #REQUIRED* *id CDATA #REQUIRED* *alarm CDATA #REQUIRED>*
26.	*<!ATTLIST timezone* *type (yearly|monthly|weekly|daily|once) "once"* *start CDATA #REQUIRED* *end CDATA #REQUIRED>*

Rad	Beskrivning till DTD (Utgående meddelande från server)
1, 2	I *message*-taggen kommer svar på de förfrågningar och uppdateringar som gjorts. Om ett *confirm-error* förekommer är det fel på det XML-meddelande som skickats till servern. I den taggen finns ett felmeddelande där orsaken är förklarad.
3, 22	*confirm-login* är svar på ett försök till inloggning. Ett attribut *result* som har värdet *ok* om inloggning lyckades och *error* om den misslyckades. Felmeddelandet kommer inom taggen och talar om det var fel användarnamn/lösenord eller dålig syntax.
4, 23	*confirm-get* har också ett attribut *result* som visar om förfrågningen lyckades eller ej. Attributet *type* talar om vilken *get* som svaret gäller. Om *get* lyckades så skickas det efterfrågade inom *confirm-get*-taggen. Om inte skickas en *errormessage*-tag med istället innehållande felmeddelandet.
6	*coord-packet* innehåller all data som kommer ifrån spårdosan, och den datan kommer i separata taggar inom *coord-packet*-taggen.
14, 25	*restriction*-taggen innehåller antingen en *timezone*-tag, en *geozone*-tag eller båda två. Attributen är strukturerade på exakt samma sätt som i XML-dokumenten som skickas till servern.
15, 26	*timezone*-taggen är precis likadan som i XML-dokumenten som skickas till servern.
21, 24	*confirm-update* har två attribut, det ena *result* talar om huruvida uppdateringar var lyckad eller inte. Om inte lyckad så finns felmeddelande inom *confirm-update*-taggen. Det andra attributet *type* gör så vi kan särskilja på de olika uppdateringarna vi gjort och därav se vilken som var lyckad och inte.

Nedan visas hur ett XML-dokument som skickas till servern kan se ut .

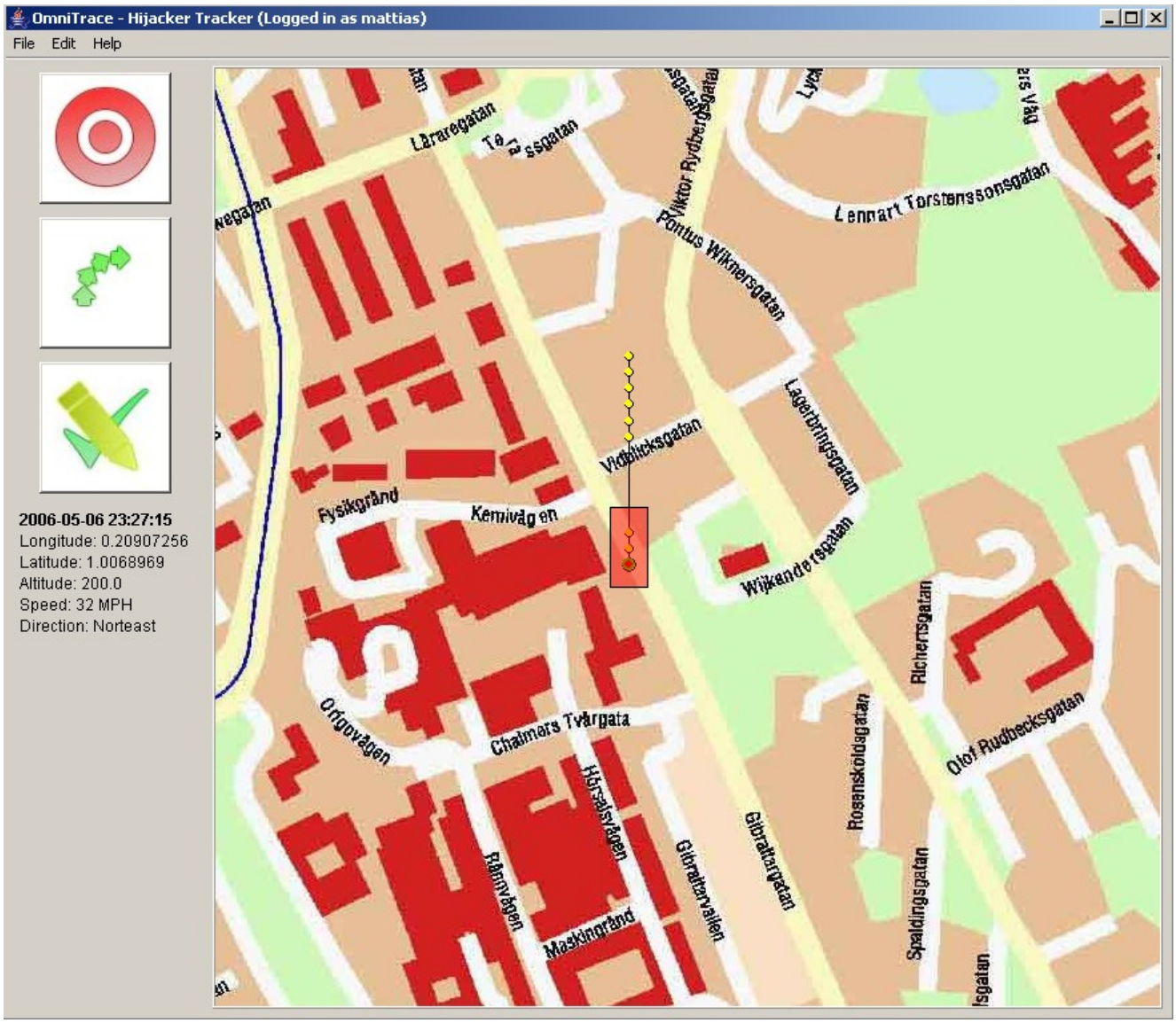
1.	*<?xml version="1.0"?>*
2.	*<message>*
3.	*<login>*
4.	*<user>bill</user>*
5.	*<password>sunisshining</password>*
6.	*</login>*
7.	*<get type="coordpackets" coordpackets-type="latest">*
8.	*<latest>10</latest>*
9.	*</get>*
10.	*<update type="restrictions">*
11.	*<restriction type="forbidden" usertext="Trip to Stockholm, kids using car" id="-1" alarm="1">*
12.	*<timezone type="once" start="#timestamp1" end="#timestamp2"></timezone>*
13.	*<geozone>*
14.	*<coord>*
15.	*<latitude>latitude coordinate1</latitude>*
16.	*<longitude>longitude coordinate1</latitude>*
17.	*</coord>*
18.	*<coord>*
19.	*<latitude>latitude coordinate2</latitude>*
20.	*<longitude>longitude coordinate2</latitude>*
21.	*</coord>*
22.	*<coord>*
23.	*<latitude>latitude coordinate2</latitude>*
24.	*<longitude>longitude coordinate2</longitiude>*
25.	*</coord>*
26.	*</geozone>*
27.	*</restriction>*
28.	*</update>*

29 .	*</message>*

Den första raden är en XML-deklaration som talar om hur dokumentet ska tolkas (rad 1). Root-tag är *message* (rad 2) som kan innehålla en *login*, *update* eller *get*-taggar. Om användaren vill logga in skickas naturligtvis *login*-tagen (rad 3) innehållande *user* och *password*-taggar. På rad 7 vill användaren se de 10 senaste loggarna på sitt fordon. Beroende på vad användaren vill åt ändras attributet *type* till ett utav de värdena som DTD-dokumentet accepterar. Detsamma gäller *update*-tagen (rad 10) där *type* ändras till det som ska uppdateras. Som namnet *restrictions* antyder, kan flera *restriction*-taggar skickas samtidigt. I exemplet vill användaren lägga till en *restriction* som dels innehåller en *timezone*-tag men också en tillhörande *geozone*-tag. Nämnvärt i *restriction*-taggen är attributet *id*, som är ett nödvändigt ont för att kunna identifiera en uppdatering på en restriktion.

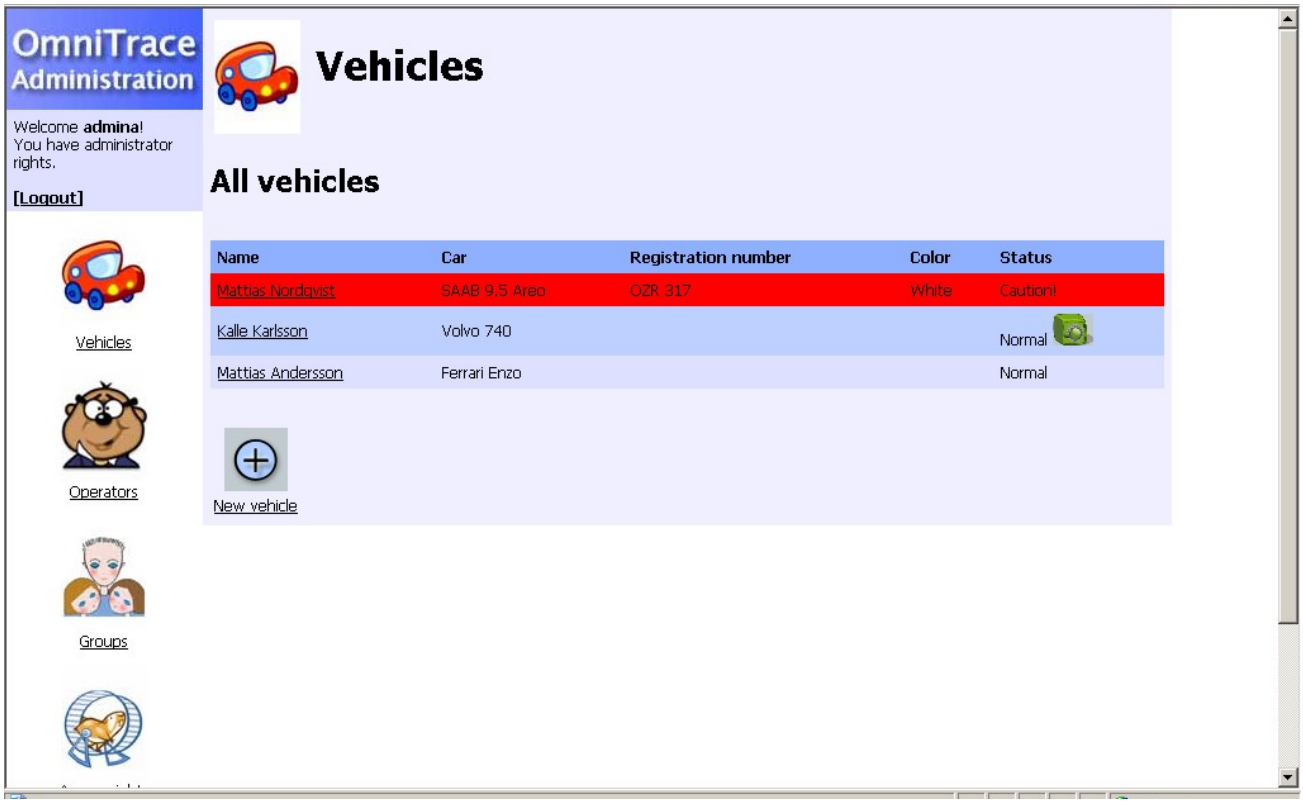
Appendix B

Detta är en bild på det användargränssnitt som vi implementerat. De gula punkter på bilden är de positioner fordonet befunnit sig på. Den röda rektangeln är en geografisk zon. Fordonets nuvarande position representeras av den gröna pricken.



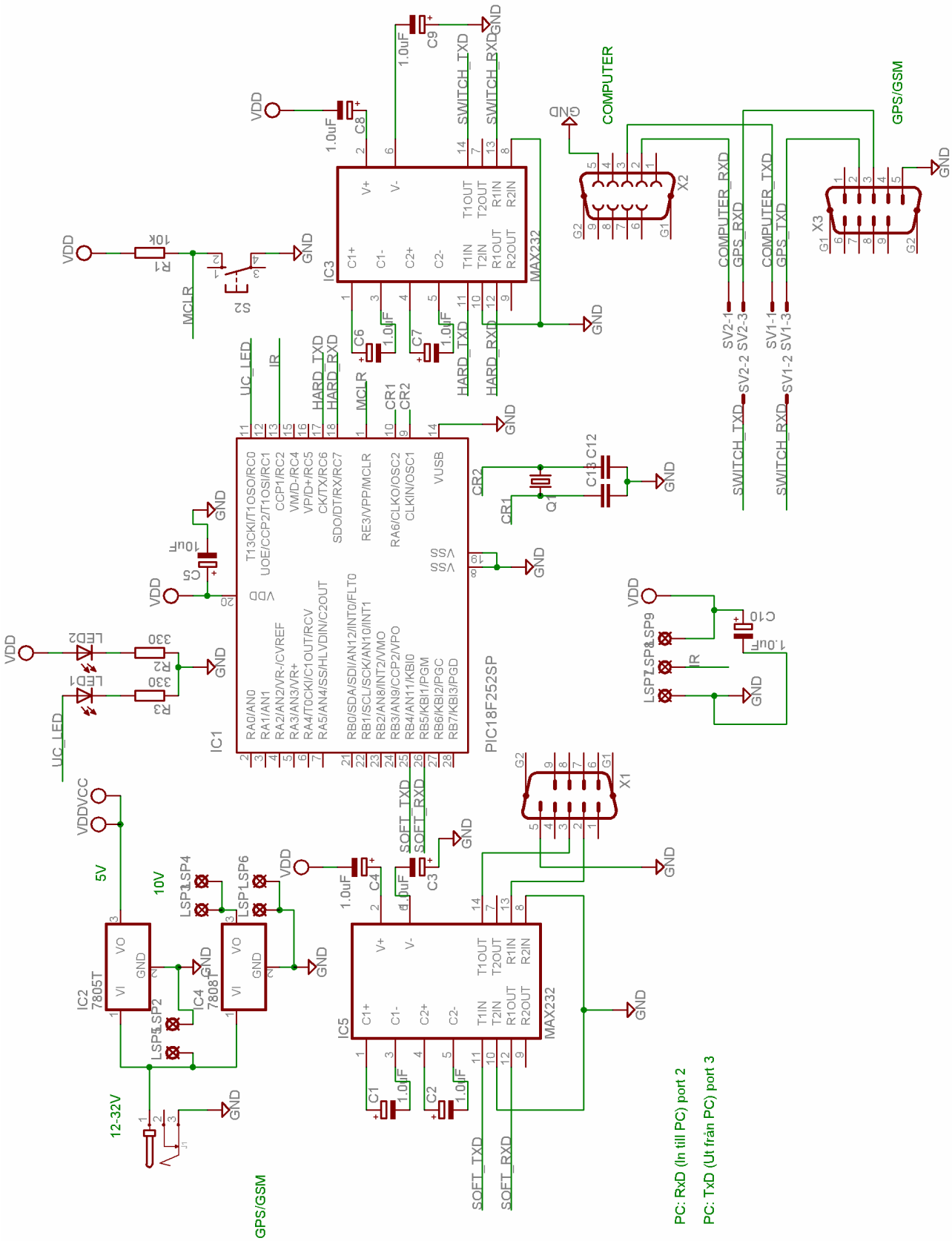
Appendix C

Administratörsgränssnittet. På bilden ser man de fordon som finns i systemet och deras status.



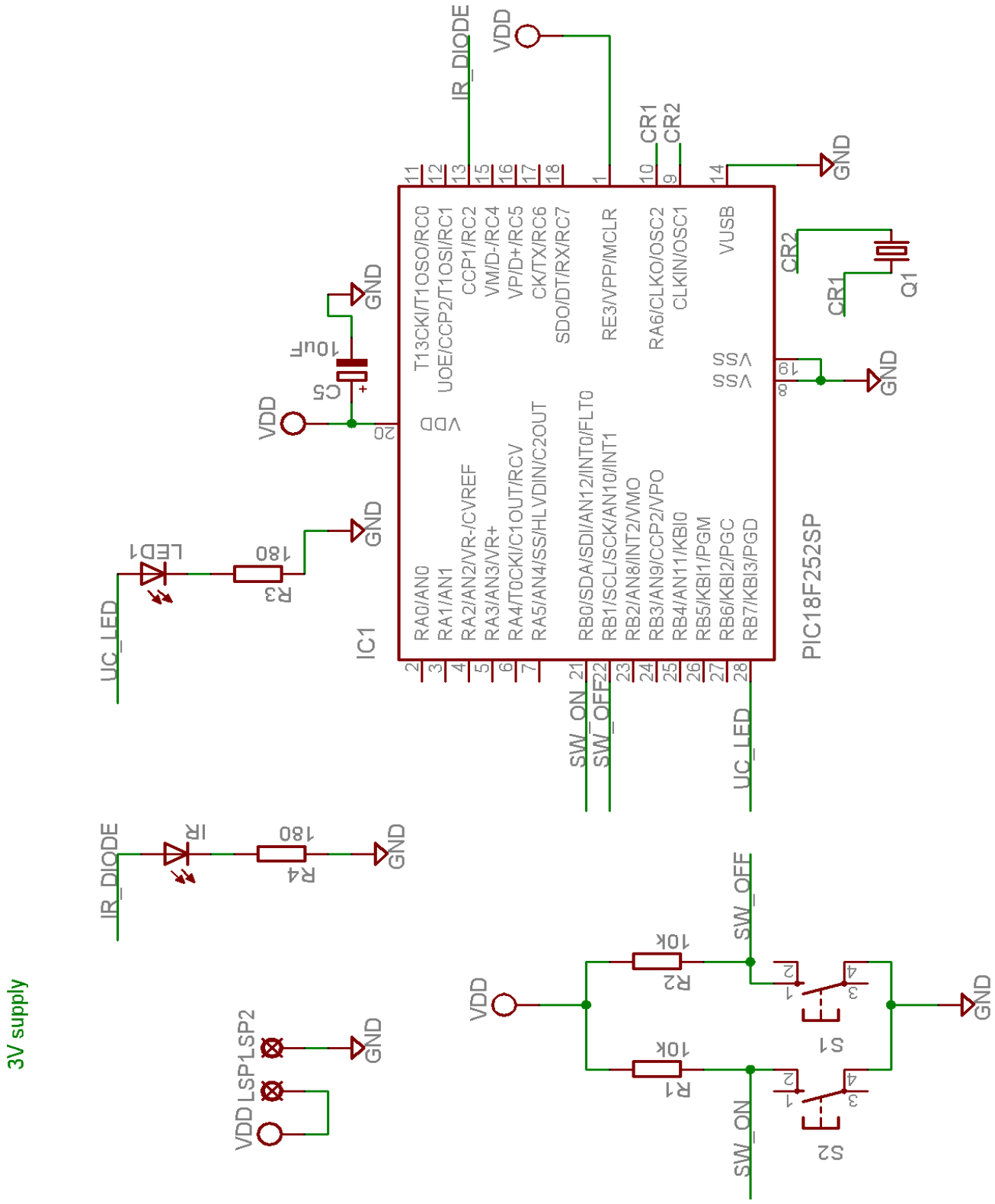
Appendix D

Detta är kopplingsschemat för huvudkortet.



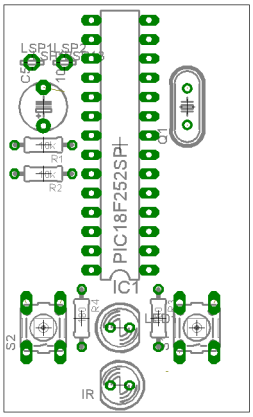
Appendix E

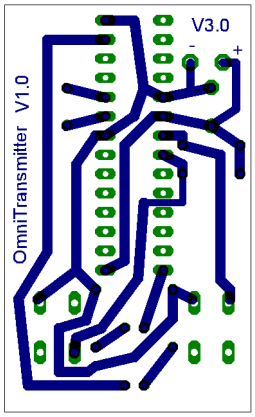
Kopplingsschema för IR-fjärrkontrollen.

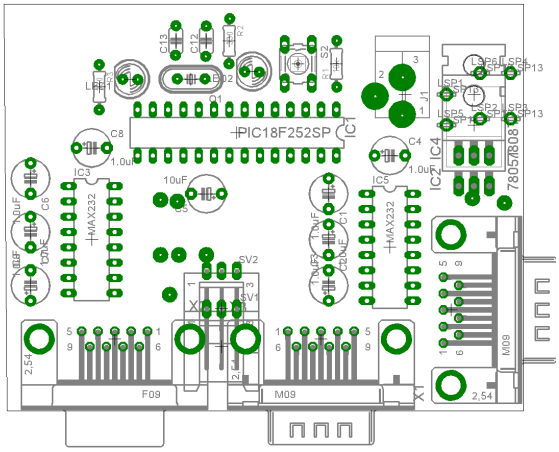


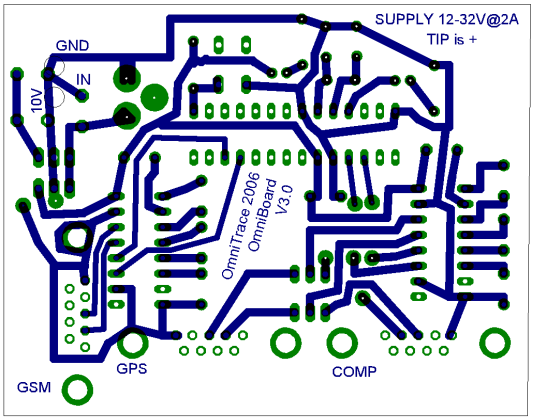
Appendix F

Komponent- och kretskortslayout för IR-fjärrkontrollen samt huvudkortet.









Appendix G

Tabell: Accounts

Name	Type	NULL	Default	Extras
Id	bigint(20) unsigned	NULL	NULL	auto_increment
username	varchar(20)		''	
password	varchar(30)		''	
first_name	varchar(100)		''	
last_name	varchar(100)		''	
car_maker	varchar(100)		''	
car_model	varchar(100)		''	
car_registration_number	varchar(20)		''	
car_color	varchar(30)		''	

Tabell: Coords

Name	Type	NULL	Default	Extras
Id	bigint(20) unsigned	NULL	NULL	auto_increment
timestamp	bigint(40)		'0'	
northspeed	float		'0'	
eastspeed	float		'0'	
altitude	float		'0'	
latitude	float		'0'	
longitude	float		'0'	
user_id	bigint(20) unsigned		'0'	

Tabell: Operators

Name	Type	NULL	Default	Extras
id	int(11)	NULL	NULL	auto_increment
username	varchar(50)		''	
password	varchar(50)		''	
administrator	enum('Yes','No')		'No'	
first_name	varchar(50)		''	
last_name	varchar(50)		''	
email	varchar(50)		''	
phone_number	varchar(30)		''	

Tabell: Restriction Area

Name	Type	NULL	Default	Extras
Id	int(11)	NULL	NULL	auto_increment
restriction_id	int(11)	NULL	NULL	
next_poly_id	int(11)	NULL	NULL	
Longitude	float		'0'	
Latitude	float		'0'	

Tabell: Restrictions

Name	Type	NULL	Default	Extras
id	int(11)	NULL	NULL	auto_increment
owner_id	int(11)	NULL	NULL	
type	enum('Square','Circle','Locked')		'Locked'	
scheduled	enum('Yearly','Monthly','Weekly','Daily','Once')		'Once'	
start_time	Datetime	NULL	NULL	
end_time	Datetime	NULL	NULL	
alarmTyp	enum('Sms')		'Sms'	

e				
poly_start	int(11)	NULL	NULL	

Tabell: Vehicles

Name	Type	NULL	Default	Extras
Id	bigint(20) unsigned	NULL	NULL	auto_increment
Username	varchar(100)		''	
Password	varchar(50)		''	
sim_phone_number	varchar(15)		''	
sms_password	varchar(255)		''	
activation_phone_numbers	mediumtext		NULL	
sms_password_changed	enum('Y','N')		'N'	
activation_phone_numbers_changed	enum('Y','N')		'N'	
alert_phone_numbers	mediumtext		NULL	

Appendix H

9Uppgift3.pdf 2006-03-16

Spårningssystem för bil - Projektplan

Grupp 9
johboh+03@dtek.chalmers.se

Lars-Erik Lis
Mattias Andersson
Michael Bodin
Johan Böhlin
Marcus Hansson
Mattias Nordqvist
Patrik Opacic

Kravspecifikation

1. ALLMÄNT

Denna kravspecifikation avser ett spårningssystem för bil.

Produkten gör det möjligt att kunna spåra sin bil vid eventuell stöld via en dator eller mobiltelefon. Användaren kommer att ha flera alternativ för aktivering/avaktivering av spårningen. Positioneringen möjliggörs av en GPS-mottagare och spårningen följer av kommunikation mellan GSM-modul och server.

2. REFERENSER

Datablad för ingående hårdvara kommer att finnas.

3. DEFINITIONER

GPS Global Positioning System, positioneringssystem.
GSM Global System for Mobile communication, globalt system för mobilkommunikation.
GPRS General Packet Radio Service, paketbaserad datakommunikation i GSM-nät.
XML Extensible Markup Language, en standard för att organisera textinformation oberoende av datorplattform, operativsystem etc.
UART Universal Asynchronous Receiver Transmitter

4. TEKNISKA KRAV

4.1. Yttre gränsytor

Styrktets <-> GPS
configureGPS(GPSMode); // Konfigurerar GPS:ens energiläge och vilket sub-protokoll av NMEA protokollet vi ska använda.
GPSData = getGPSData(); // Tar emot strukturerad data från GPS:en.

Styrktets <->GSM
connectGPRS(server,username,password); //Ansluter till mobiloperatörens GPRS tjänst.
configureGPRS(GPRSMode); //Konfigurerar GPRS:ens energiläge.
GPRSData = getGPRSData(); //Tar emot strukturerad data från GPRS:en.
sendGPRSData(GPRSData); //Skickar strukturerad data till GPRS:en.
GSMData = getGSMData(); //Tar emot data från GSM:en. Detta kan vara tex. ett SMS.

Server <-> GSM
traceData = getTraceData(); //Tar emot strukturerad data, innehållande longitud, lattitud, hastighet, tid och riktning, från styrenheten i fordonet.
authenticate(unitID); //Autenticiera från GSM:en mot servern.
configureUnit(unitMode,deafultSecurity); //Konfigurerar styrenhetens geografiska zoner, tidzoner och standard beteende i fordonet.
activateUnit(unitID); //Aktiverar en styrenhet via SMS.

Notering: Aktivering via SMS krävs då en anslutning till GSM-modulen inte kan ske utifrån, utan GSM-modulen måste själv ta initiativ till anslutning.

4.2.1 Funktionskrav för färdigt system

- Aktivering och avaktivering på följande sätt
    - Via SMS
    - Schemaläggning
    - Geografiska zoner
- Vid aktivering väljer man säkerhetsnivå:
    1. Larma polis eller annan auktoritet och logga position.
    2. Larma ägaren och logga position.
    3. Logga position på server utan åtgärder
- Uppladdningsbar strömkälla för självförsörjning.
- Servern ska tillhandahålla autenticering, loggning av mätdata från fordon, gränssnitt för administratör och ett XML-gränssnitt för 3:e-parts dataåtkomst.
- Ett användargränssnitt för användaren.

4.2.2 Funktionskrav för olika prototyper av spårkrets

4.2.3.1 Prototyp 1.0

- Spårkresten skall kunna ta emot koordinater från GPS och skicka dem till servern. Detta skall göras direkt efter uppstart.

4.2.3.2 Prototyp 1.1

- Spårkretsen skall byggas ut med egen strömförsörjning.
- Den skall kunna aktiveras och börja sända med sms eller vid rörelse.

4.2.3.3 Prototyp 1.2

- En IR-fjärrkontroll ska byggas som kan aktivera respektive avaktivera spårkretsen. Enheten skall också kunna aktiveras med upprigning.

4.2.3.4 Prototyp 1.3

- Man skall kunna ställa in olika säkerhetsnivåer på spårkretsen och den skall agera efter dessa.
- Spårkretsen skallklara av att ta emot geografisk zon från servern och starta då den befinner sig utanför zonen.

4.2.4 Funktionskrav för olika prototyper av server

4.2.4.1 Prototyp 1.0

- Servern skall kunna logga koordinater från spårkretsen och skriva dessa till en fil.

4.2.4.2 Prototyp 1.1

- Servern skall kunna lagra koordinaterna i en databas.
- XML - protokoll skall implementeras för datautbyte med ett användargränsnitt.

4.2.4.3 Prototyp 1.2

- Skall kunna hantera schemaläggningstider en bil markeras som stulen om den rör sig på en tid då den egentligen ska stå stilla.
- Skall kunna hantera geografiska områden funktionen det vill säga, markera bilen som stulen när den åker utanför dessa.

4.2.4.4 Prototyp 1.3

- Eventuella övriga funktioner.

4.2.5 Funktionskrav för olika prototyper av användargränssnitt

4.2.5.1 Prototyp 1.0

- En karta med position på vart spårkretsen befinner sig.

4.2.5.2 Prototyp 1.1

- Administrationsverktyg för att få in fler bilar i systemet.

4.3. Samband mellan funktionerna

Strömförsörjningen bör fungera för att samtliga funktioner skall kunna fungera tillfredsställande. Vid aktivering ska senast använda nivå tillämpas om inte användaren ändrat nivå. Loggning startar alltid vid aktivering och slutar vid avaktivering. Användargränssnittet jobbar mot servern och gör serverns loggning av positioner begripligt för användaren.

4.4. Driftkrav

Produkten ska klara av:

- Temperaturer ner till -20°C och upp till 70°C
- Fukt
- Vibrationer
- Störningar från bil

Den ska också kunna göra en automatisk omstart då detta behövs. Den skall också vara utformad på ett sådant sätt att den går att installera i en bil.

4.5. Kapacitetskrav

- Man ska kunna ange hur ofta man vill läsa av positioner från GPS-modulen, genom att ange ett loggningsintervall.
- Den självförsörjande strömkällan skall, med utgångspunkt att den är fulladdad, tillfredsställa systemets strömbehov i 48 timmar.

4.6. Säkerhetskrav

Det är, ur integritets- och säkerhetssynpunkt, väldigt viktigt att endast behöriga personer får tillgång till spårningen.

4.7. Konstruktionskrav

- Systemet ska vara modulbaserat och utformat så att gränssnitten är tydliga och klara.
- Servern ska vara plattformoberoende

4.8. Användarens kapacitet och kompetens

- Användaren bör ha förmågan at kunna använda en telefon med hjälp av en användarmanual.
- Tjänsteadministratör bör ha datorvana.

Utvecklingsplan

1. UPPDRAG

Projektet består i att utveckla ett spårningssystem för bilar som kan förebygga nuvarande problem med bilstölder. I bilen kommer det att sitta en spårdosa som aktiveras vid stöld. En spårcentral, som består av en server, tar emot koordinater som loggas och kan presenteras för bilägaren via en dator eller mobiltelefon. Även polisen skulle kunna ta del av informationen via direktlarm.

Scenario:
Bilägaren parkerar och larmar sin bil.
Bilen blir stulen.
Spårdosan skickar information om bilens position till servern.
Bilägaren får information på sin mobil om att bilen är stulen.
Polisen och bilägaren kan spåra bilens position via en hemsida.

2. REFERENSER

Referenser saknas.

3. DEFINITIONER

Definitioner saknas.

4. UTVECKLINGSMODELL

Visionen står klar och samtliga inom gruppen är överens om att utveckling enligt spiralmodell faller mest naturligt. De olika prototyperna ses i karvspecifikationen under punkt 4.2, och den utdelade tiden för varje prototyp står i tidsplanen.
Spårningssystemet som helhet är uppdelat i tre delar:

- Spårdosa
- Server
- Användargränssnitt

Delarna kommer att realiseras parallellt, i etapper, eller versioner, där både spårdosan och servern i dagsläget har tre versioner utstakade. Användargränssnittet är ännu inte definierat. Mer information om versionerna kommer göras tillgängligt så snart projektets hemsida är i drift.
Då de ingående delarna realiseras i versioner enligt modell kommer det falla naturligt att de mest grundläggande funktionerna implementeras, för att sedan byggas ut enligt vision. Det säkrar projektets utbyggbarhet under de kommande åren i drift.

5. PROJEKTSTYRNING

5.1. Projektets organisation och resurser

Projektledare: Michael Bodin
Projektbibliotekarie: Mattias Andersson
Utvecklare: Johan Böhlin
Testare: Patrik Opacic
Granskningsledare: Marcus Hansson

5.2. Tids- och kostnadsplan

	Sep	Okt	Nov	Dec	Jan	Feb	Mar	Apr	Maj	tim
Kravspec.	----------|									200
Konstr.dok.		----------								200
Testspec		----------|								250
EK1				----------|					300
ST1							---|			100
EK2							--------|			250
ST2								---|		150
EK3+EK4								---------|		250
ST3+ST4									---|	150
Slutlig dok.									--------	150
Leverans									-	10
										2010

EK = enhetskonstruktion
ST = systemtest
Numrering av processerna ovan representerar olika versioner av vår produkt.

Kostnader:
Hårdvarukomponenter 4000
Kommunikationsavgifter 1000
Transportavgifter 400
Teambildning 700
Konferenskostnader 300
Litteratur, kontorsmaterial 600
Totalt 7000:-

5.3. Aktivitetsbeskrivningar

Kravspecifikation Definition av alla delar av projektet
Milstolpe, | Granskning av arbete sen föregående milstolpe
Konstruktionsdokument En detaljerad beskrivning av samtliga gränssnitt mellan de olika delarna av systemet
Testspecifikation Definition av alla felkällor och dess åtgärder
EK, enhetskonstruktion Hopsättning av alla komponenter och kodning av programvara
ST, systemtest Testning av alla delar i systemet och dess egentliga funktion
Slutgiltig dokumentation Sammanställning av hela projektet
Leverans Överlämning av produkten till beställare

5.4. Progressrapportering

Sker genom tidsstämpling och mötesprotokoll via hemsida.

6. PROGRAMUTVECKLING

6.1. Standarder och rutiner

Vi kommer att använda oss av Javakonventioner och XML-protokollet. Gränssnittet mellan GSM och server definieras senare. För mikrokontrollern används Microcontroller-C.

6.2. Existerande programvara

Okänt i dagsläget

7. KONTROLL AV ARBETSRESULTAT

Alla deltar på något sätt i granskningen. Granskningsledare Marcus Hansson kommer att fördela arbetet.

8. DOKUMENTHANTERING

Vi kommer att följa en konstruerad dokumentmall där typsnitt, storlek, etc. är specificerat. Vid ändring i dokument måste hela dokumentet omgranskas för att bibehålla konsistens. För att undslippa problem när flera vill redigera ett dokument kommer en utcheckningsfunktion, av dokument från hemsidan, att implementeras.

9. LEVERANS

Leveransen innehåller krets och etui, serverprogramvara, användargränssnitt, dokumentation och manual.

Konstruktionsdokument

1. ALLMÄNT

Produkten gör det möjligt att kunna spåra sin bil vid eventuell stöld via en dator eller mobiltelefon. Användaren kommer att ha flera alternativ för aktivering/avaktivering av spårningen. Positioneringen möjliggörs av en GPS-mottagare och spårningen följer av kommunikation mellan GSM-modul och server.

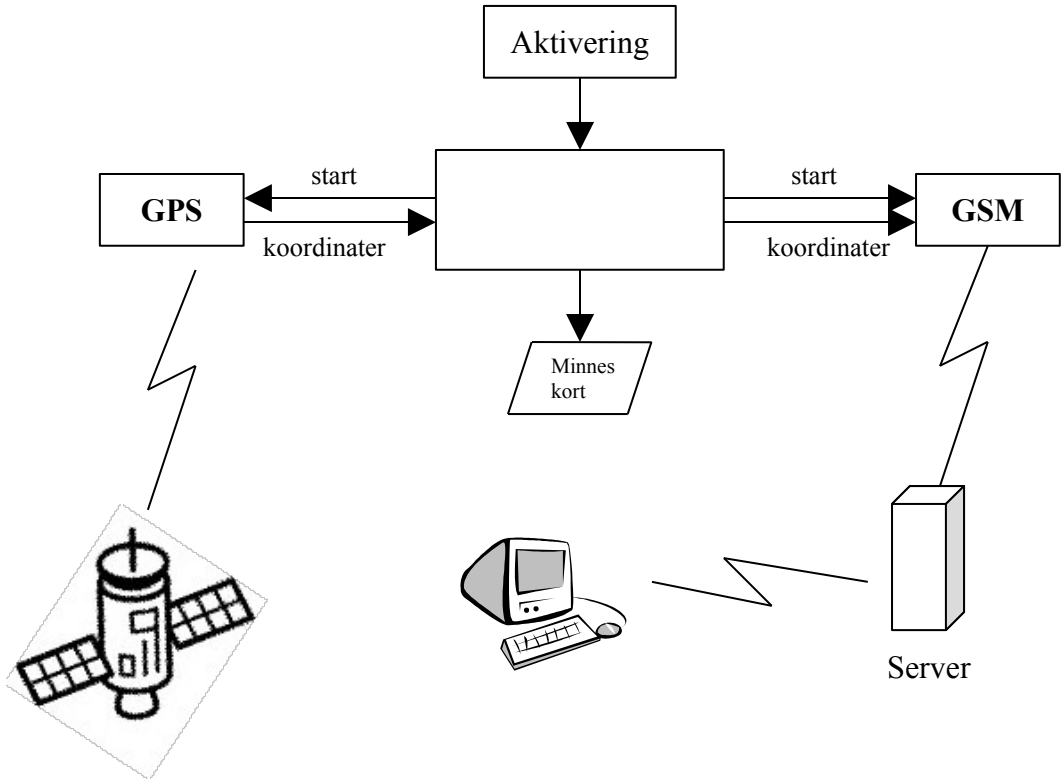
2. REFERENSER

Okänt för tillfället

3. DEFINITIONER

Inga för tillfället

4. ÖVERGRIPANDE KONSTRUKTION



Verifieringsplan

1. ALLMÄNT

Projektet består i att utveckla ett spårningssystem för bilar som kan förebygga nuvarande problem med bilstölder. I bilen kommer det att sitta en spårdosa som aktiveras vid stöld. En spårcentral, som består av en server, tar emot koordinater som loggas och kan presenteras för bilägaren via en dator eller mobiltelefon. Även polisen skulle kunna ta del av informationen via direktlarm.

2. KRAFSPECIFIKATION

Kravspec. Den bifogade.

3. DEFINITIONER

GPS Global Positioning System, positioneringssystem.
GSM Global System for Mobile communication, globalt system för mobilkommunikation.
GPRS General Packet Radio Service, paketbaserad datakommunikation i GSM-nät.
XML Extensible Markup Language, en standard för att organisera textinformation oberoende av datorplattform, operativsystem etc.
UART Universal Asynchronous Receiver Transmitter.

4. TESTOMGIVNING

4.1 Programvara

Serverprogramvaran som tar emot koordinater från spårsändaren, på valfritt system.

4.2 Hårdvara

Själva spårsändaren

5.0 FÖRBEREDELSER

Installation av server.

6.0 ALLMÄNNA KRAV

Hela systemet skall fungera enl. kravspec.

6.2.1 Testfall 1:

Mål: Testa grundläggande funktioner i systemet.

Tillvägagångssätt: Larma spårkretsen med IR-fjärrkontrollen. Flytta runt spårkretsen under 10 min.

Godkännandekriterier: När spårkretsen börjar röra på sig ska användaren få en sms-notifiering, servern ska börja logga koordinater och fortsätta göra så under hela tiden.

6.2.2 Testfall 2:

Mål: Kolla om spårkretsen aktiveras när den rör på sig och avaktiveras när den står stilla.

Tillvägagångssätt: Larma spårkrets med IR-fjärrkontrollen. Flytta runt spårkretsen under ca 2 min och stanna i ca 2 min – upprepa 3 ggr.

Godkännandekriterier: Vid rörelse ska servern få koordinater och vid stopp ska servern sluta få koordninater. Aktivering bör ske genast vid rörelse och avaktivering får ej ta längre tid än 2 minuter.

6.2.3 Testfall 3:

Mål: Kolla om SMS-, IR,- och telefon-larmning fungerar/avlarmning av spårkretsen fungerar korrekt.

Tillvägagångssätt: Avlarma spårkrets med IR-fjärrkontrollen. Flytta runt spårkretsen och gör följande: Larma med SMS/IR/telefon och låt den vara aktiv under ca 2 min. Avlarma och vänta ytterligare ca 2 minuter. Upprepa 3 ggr för varje larmningsmetod.

Godkännandekriterier: Vid larmning ska servern börja få koordinater och vid avlarmningska servern sluta få koordinater. Ingen större fördröjning får ske. SMS-notifiering ska ske vid varje larmning.

6.2.4 Testfall 4:

Mål: Kolla om precisionen är OK och om kartan är rätt kalibrerad.

Tillvägagångssätt: Larma spårkrets med IR-fjärrkontrollen. Flytta spårkretsen till tre olika välkända platser och notera positionsindikatorn på kartan vid varje plats.

Godkännandekriterier: Positionsindikatorn på kartan ska stämma med en felmarginal på högst 5 m exklusive GPS:ens felmarginal.

6.2.5 Testfall 5:

Mål: Kolla om geografiska zoner fungerar som de ska.

Tillvägagångssätt: Konfigurera användaren och lägg till en tillåten zon runt spårkretsen (med sms-notifiering). Flytta runt spårkretsen i geografiska zonen under ca 5 min. Flytta sedan spårkretsen in och ut ur den geografiska zonen 3 ggr.

Godkännandekriterier: Spårkretsen får ej larmas då den är inom den tillåtna geografiska zonen. Varje gång spårkretsen går utanför zonen ska den larmas och en sms-notifiering ska skickas. När spårkretsen går in i zonen ska den avlarmas. Servern ska få ut koordinater under hela förloppet.

6.2.6 Testfall 6:

Mål: Kolla om förbjudna zoner fungerar korrekt.

Tillvägagångssätt: Konfigurera användaren och lägg till en förbjuden geografisk zon runt spårkretsen. Lägg också till en förbjuden zon innanför den tillåtna. Flytta spårkretsen in och ut ur den förbjudna zonen 3 ggr.

Godkännandekriterier: Spårkretsen får ej larmas då den är inom den tillåtna zonen. Varje gång spårkretsen går in i förbjudna zonen ska den larmas och när den går ur ska den avlarmas. Servern ska få koordinater under hela förloppet.

6.2.7 Testfall 7:

Mål: Kolla om schemaläggaren fungerar.

Tillvägagångssätt: Konfigurera användaren och lägg till en tidsbegränsning som sträcker sig från nuvarande tid exklusive 7 min. Avlarma spårkretsen och flytta den under 10 min.

Godkännandekriterier: Spårkretsen får ej larmas under den tillåtna tiden och ska sedan genast larmas. Servern ska få koordinater under hela förloppet.

7.0 ÖVRIGT.

Inget speciellt än så länge.

Omnitrace Administration

Instruction manual

Last revision: 2006-05-18

Last author: Mattias Andersson,
mono@student.chalmers.se

Copyright 2006 by Mattias Andersson, Mattias Nordqvist, Patrik Opacic, Johan Böhlin, Lars-Erik Lis, Marcus Hansson and Michael Bodin.

1.1 Introduction

The Omnitrace Administration is used to administrate the fundamental settings of the Omnitrace OmniServer.

The interface is self-explainatory and this instruction manual serves more as a guide to the new administrator or operator.

This instruction manual does however not cover any other part of the Omnitrace product, such as the XML protocol, only the Administration software.

2.1 Login

The login procedure is simple one where you only enter your username, your password and press the *'Login'*-button. The administration software will verify the information and if correct, you will enter the Administration interface.

Ill 2.1 – Login screen

3.1 The interface

The interface consists of two main parts, the menu to the left, and the selected menu option to the right.

You can also see your access level in the top left corner, where you can also logout.

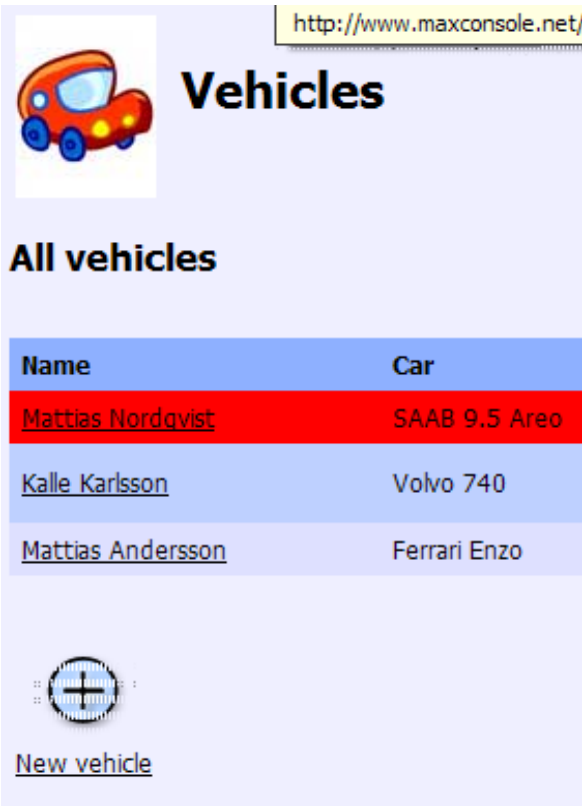
Ill 3.1 – The interface

4.1 Vehicle administration

The first thing you'll see is a list of your assigned vehicles, or all vehicles if you have administrator status. You can simply add more vehicles by pressing the *'New vehicle'*-button.

Vehicles marked with red background have moved beyond their assigned limits and should be monitored immediately.

Vehicles who are marked with a safe have an active alarm which means that the vehicles is not allowed to move at all. This is the most common form of restriction and is controlled with an IR-remote directly from the vehicle.



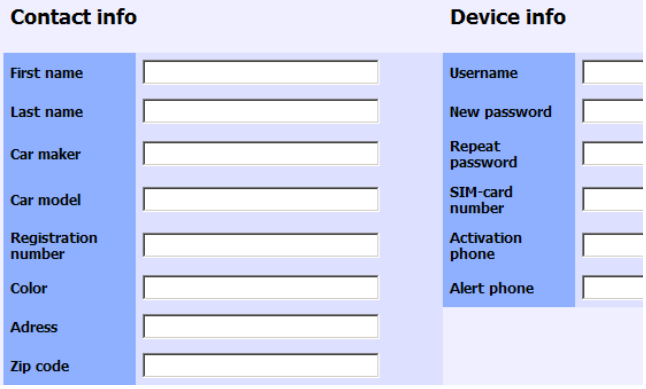
Ill 4.1 – Vehicles overview

4.2 Adding new vehicles

You will be presented with a form (see illustration 4.2) in which you enter the personal information about the owner of the vechicle, and the device information.

The device information is used by the transmitter unit in the vehicle while the contact information is used by the operator and the owner.

More specific details can be found in chapter 4.4.



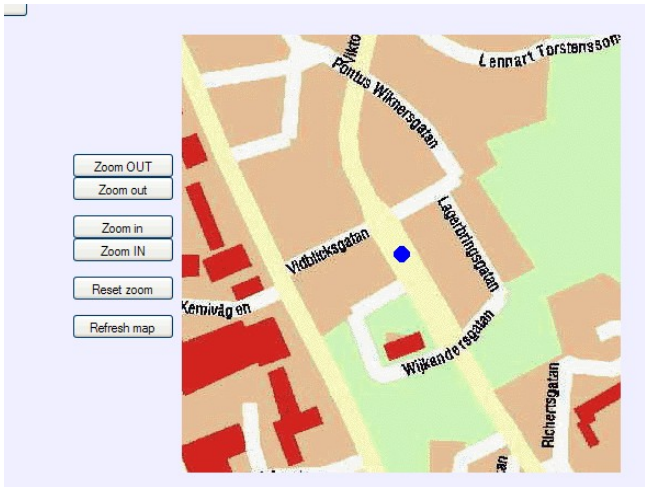
Ill 4.2 – Creating new vehicle account

Ill 4.3 – Viewing vehicle information

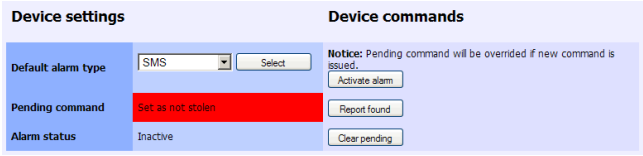
4.3 Monitoring vehicles

When clicking on a vehicle you will first be presented with the contact- and device information about the vehicle. You can press the 'Update'-button to update these.

Below the information you will be presented



Ill 4.4 – Tracking vehicle on the map



Ill 4.5 – Issuing vehicle commands

with a map marked with the current location of the vehicle. You can press the buttons to control the zoom settings.

The last part is used to change the device status and send commands. In the illustration x.x you can see that the default alarm type is set to *'SMS'* which means that if the vehicle moves while the alarm is active, this is the type of alarm that will be used.

Pending command is a command that is waiting to be sent to the vehicles. The nature of the system works so that the transmitter unit and the server synchronizes when the transmitter unit connects to the server. Any pending commands will then be issued.

In the illustration example the vehicle has been stolen and the next time the transmitter unit logs in the stolen status will be reset to normal.

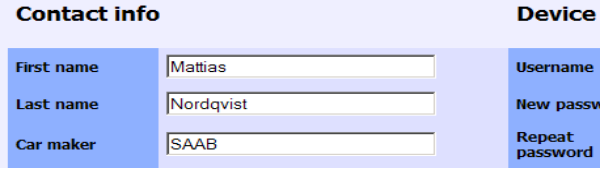
4.4 Updating vehicle account

In illustration 4.3 there is an *'Update'*-button which can be used to update all information regarding a vehicle.

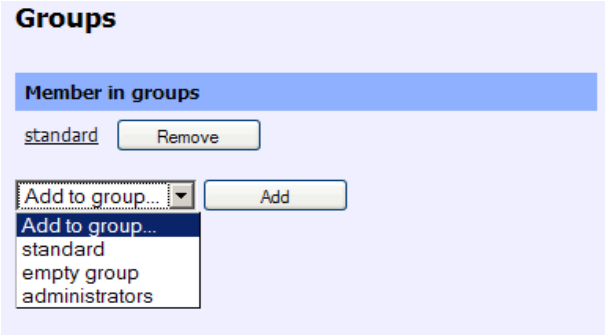
The first part is the contact- and device information which was presented in chapter 3.1. You can simply edit all data presented and then press the *'Save'*-button to save the new data.

Groups will be explained in detail in chapter 6.1 but as you can see from illustration 4.7 you can assign the vehicles to operator groups. Remove it from a group by clicking the *'Remove'*-button.

Access rights are a set of rules which define which IP:s are allowed to connect to the server. This should be the IP of the gateway which is used by the cell phone service provider. In case the IP is dynamic, or more



Ill 4.6 – Updating contact information

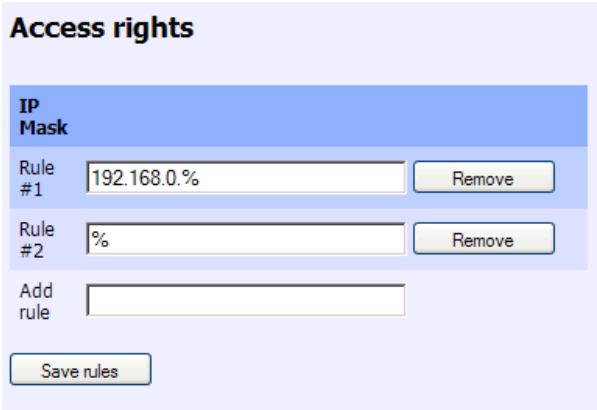


Ill 4.7 – Updating vehicle group assignments

than one IP is used you can simply add more or use masks.

Example: **192.168.0.%**

All IP:s beginning with *'192.168.0.'* are allowed to connect to the server.



Ill 4.8 – Updating vehicle access rights

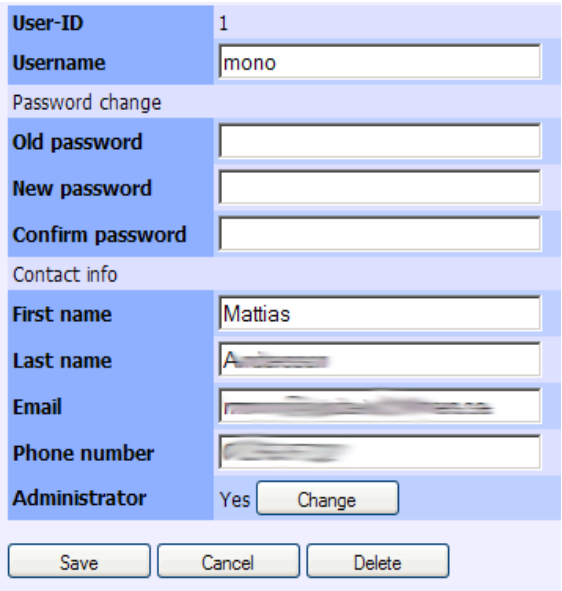
5.1 Operator administration

This section is only available to administrators.

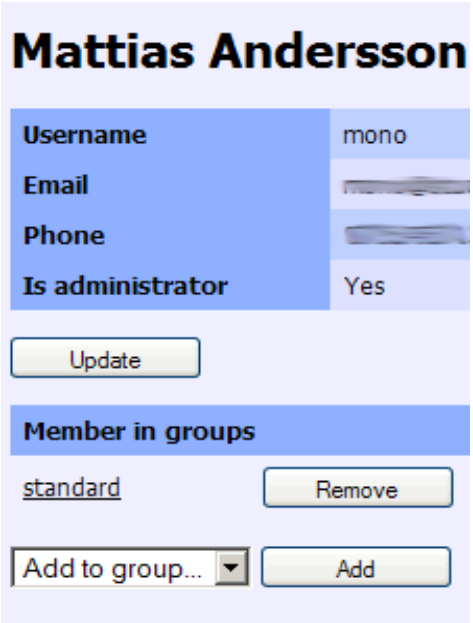
Pressing the *'Operators'*-icon in the menu takes you to the operator overview. This contains a list of all operators and the *'Create operator'*-button.

Clicking on the username of an operator brings up information about that operator. Here you can select which operator groups, which are explained in chapter 6.1, to which the operator belongs. You may also click the *'Update'*-button to update the operator information.

Ill 5.1 – Operator overview



Ill 5.2 – Update operator account



Ill 5.3 – Operator account

6.1 Groups

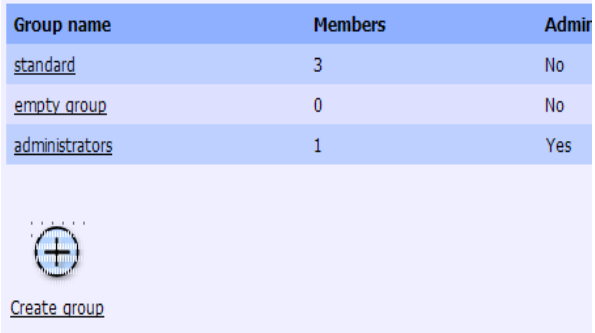
Administrators are shown all groups, while operators are only shown the settings and members for his own groups.

All operators and vehicles can belong to one or more groups. An operator gain access to all vehicles which belong to the groups of that operator.

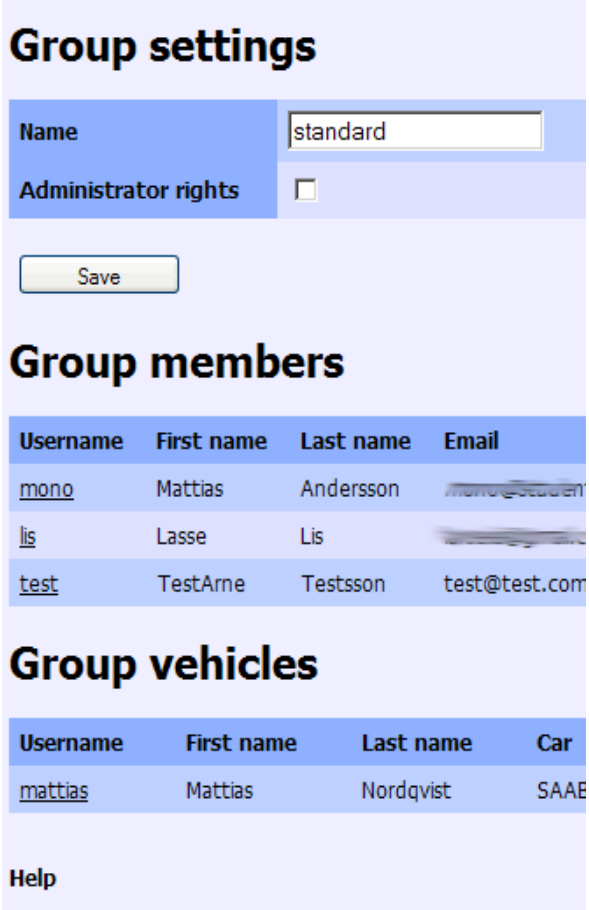
A group may have administration rights. This overrides the per operator setting and the operator need only to be in one group with administration rights to inherit them.

Clicking on a group name opens up the settings for that group. Here you can change the name and set administration rights.

You can also see all operators and vehicles assigned to this group, and remove them if needed. You add operators to the group from the operator settings section, and vehicles from the vehicle settings section.



Ill 6.1 – Group overview



Ill 6.2 – Group settings

7.1 Access rights

This section is only available to administrators.

In this section administrators can control which series of IP numbers which can connect and fetch information from the server using the XML protocol.

Example: **192.168.0.%**

In this example all clients with an IP that begins with *'192.168.0.'* are allowed to connect to the server.

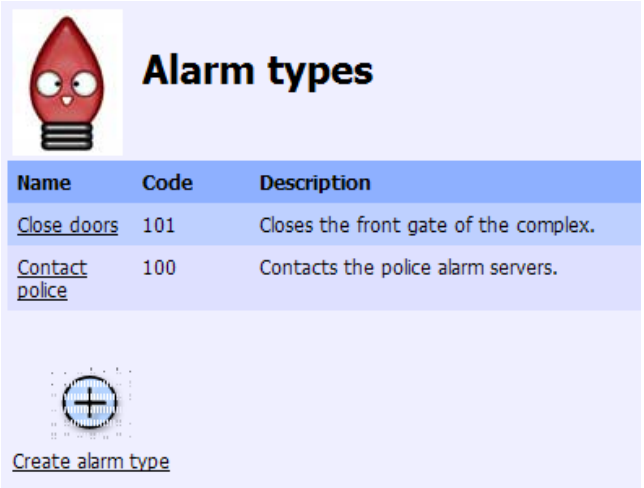
Ill 7.1 – Access rights settings

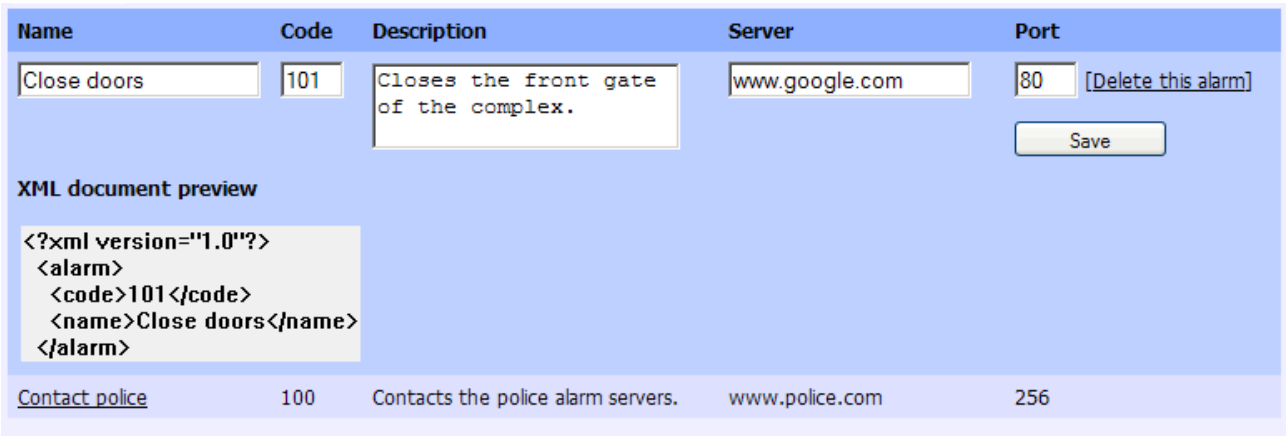
8.1 Alarm types

This section is only available to administrators.

In this section administrators can define customizable alarm types. When a vehicle moves outside of a defined geographical or time-based restriction an alarm of either pre-defined type or customizable type is sounded. The server then connects to the pre-defined server and port and posts an XML document to which the receiving system can then react appropriately. This of course requires that the system can receive TCP/IP connections.

The overview lists all defined alarm types and gives you the option to create new alarm types. Clicking on one of these will bring up the selected alarm type. Enter the information and press the *'Save'*-button to store the new data.





Ill 8.2 – Alarm types settings