Slim opladen van meerdere elektrische voertuigen

Software

O.F.C. de Groot en P.J. Marcelis

Technische Universiteit Delft

TU Delft Delft University of Technology

Challenge the future

Slim opladen van meerdere elektrische voertuigen

Software

door

O.F.C. de Groot en P.J. Marcelis

in overeenstemming met de vereisten voor het verkrijgen van de graad van

Bachelor of Science
in Electrical Engineering

aan de Technische Universiteit Delft,

Studentnummer: resp. 4114280 en 4024087
Begeleiders: Prof. dr. ir. P. Bauer
R. Hulleman

Abstract

De huidige beschikbare laadpalen zijn eenvoudige voedingsbronnen die geen rekening houden met omstandigheden zoals het energieverbruik van de rest van een gebouw. Hierdoor ontstaan pieken in het energiegebruik en dit brengt hoge kosten met zich mee. Software die nodig zal zijn voor het opladen van elektrische voertuigen met kostenbesparende oplaadpalen zal in deze thesis ontworpen en gerealiseerd worden.

De software is opgedeeld in vier onderdelen: het algoritme, de vermogensberekening, de gebruikersinterface en de aansturing. De vormgeving en de mogelijkheden van ieder onderdeel worden apart onderzocht. Voor het algoritme zal onderzocht worden hoe een laadschema gegenereerd kan worden dat rekening houdt met prioriteit van verschillende gebruikers. Vervolgens wordt voor de vermogensberekening bepaald wat het beschikbaar vermogen is voor de laadpalen en wordt er een schatting gemaakt van het toekomstig beschikbaar vermogen. De gebruikersinterface vormt de brug tussen de gebruiker en het algoritme, waarmee benodigde informatie opgevraagd wordt om het algoritme optimaal te laten functioneren. Als laatste wordt onderzocht hoe de hardware aangestuurd kan worden door de hoeveelheid stroom die iedere auto krijgt door te geven. De onderdelen worden los van elkaar gesimuleerd en in het hoofdstuk integratie wordt beschreven hoe het tot een werkend geheel wordt gemaakt. De thesis wordt afgesloten met een reflectie op het geleverde werk en er worden aanbevelingen gedaan voor eventueel vervolgonderzoek.

Voorwoord

Bachelor afstudeerproject

Wij zijn studenten Bachelor Electrical Engineering aan de Technische Universiteit Delft. De Bachelor wordt afgesloten met een Bachelor afstudeerproject. Voor het afstudeerproject delen de kandidaten zich op in groepen van vier à zes personen, die gedurende het vierde kwartaal (ongeveer elf weken) onderzoek verrichten aan en een prototype ontwikkelen voor een bepaald probleem. Een dergelijk probleem wordt vanuit de universiteit aangeboden.

Deze groepjes van vier à zes personen zullen vervolgens weer opsplitsen in subgroepen van twee personen, waarbij elke subgroep een onderdeel van het probleem zal aanpakken. Elke subgroep moet een thesis inleveren en verdedigen, en de gehele groep moet uiteindelijk een prototype of proof-of-concept kunnen presenteren.

Atstudeerproject bij een bedrijf: Priva

Zoals eerder genoemd levert de universiteit normaliter het probleem aan. Wij (de gehele groep) waren echter al ruim voor de start van het afstudeerproject overeengekomen dat een zelf gevonden project meer uitdaging en leerzaamheid zou bieden. Vanaf een half jaar voor de start van het afstudeerproject hebben wij dan ook contact gezocht met bedrijven, om te vragen of ze eventueel een geschikt project voor ons hadden.

Een van de bedrijven die reageerden was Priva. Zij hadden wellicht een geschikt project voor een viertal studenten elektrotechniek. Bij het hoofdkantoor van Priva waren namelijk zes oplaadpalen, maar onderhand al tien elektrische voertuigen. Om meer traditionele oplaadpalen bij te plaatsen, moest er geïnvesteerd worden in het verdeelstation van het elektriciteitsnet, omdat de piekstroom anders te hoog zou kunnen worden.

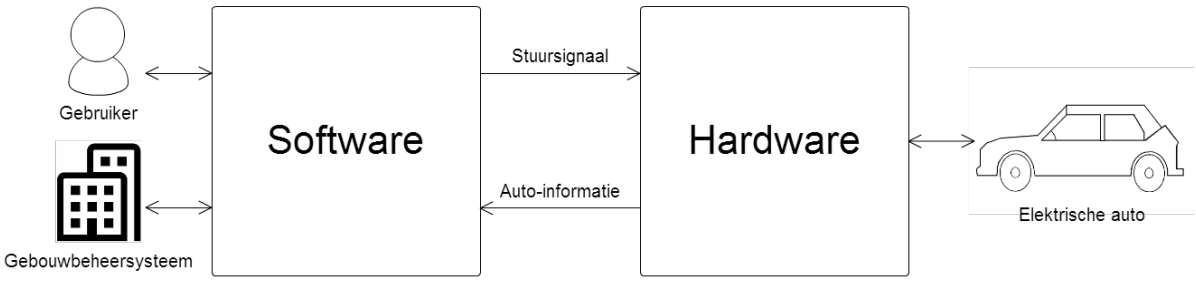
Priva was dus op zoek naar een manier om auto's slimmer op te kunnen laden, en hierin zagen ze een Bachelor afstudeerproject. Na overleg met Priva en de organisatie van het afstudeerproject, hebben wij bij de universiteit een enthousiaste begeleider gevonden in de persoon van professor P. Bauer. Zodoende kon ons bachelor afstudeerproject plaatsvinden.

Totaalsysteem

De groep heeft zich bezig gehouden met een nieuw systeem om meerdere elektrische auto's slim op te kunnen laden. In deze thesis wordt het software-deel geanalyseerd, worden ontwerpkeuzes onderbouwd en uiteindelijk getest om oplaadinstructies door te sturen naar de hardware. De andere subgroep heeft aan het hardware-deel van het systeem gewerkt [1]. Samen vormen de twee onderdelen een nieuwe manier van auto's opladen. Het totaalsysteem en de relatie tussen de hardware en software is weergegeven in figuur 1.

Wij tweeën hebben samen een subgroep gevormd vanuit een groep van vier personen. De groep heeft zich bezig gehouden met een nieuw systeem om meerdere elektrische auto's slim op te kunnen laden. Dit thesis gaat over het software-deel van het systeem. De andere subgroep heeft aan het hardware-deel van het systeem gewerkt [1].

O.F.C. de Groot en P.J. Marcelis
Delft, Juni 2014



Figuur 1: Schema van totaalsysteem met software en hardware. De software genereert een laadschema aan de hand van informatie die verkregen is van de gebruikers en het gebouwbeheersysteem. De stuursignalen voor de laadpunten worden doorgestuurd naar de hardware die deze verwerkt en de auto's daadwerkelijk gaat opladen. Er wordt door de hardware feedback gestuurd naar de software over de auto's.

Inhoudsopgave

Lijst van figuren

Lijst van tabellen

1

Introductie

1.1. Probleemidentificatie

De wereld bevindt zich in een sterke transitie richting elektrisch personenvervoer. In traditionele voertuigen met een verbrandingsmotor wordt de benodigde energie voor voortbewegen gehaald uit de decentrale energieopwekking in verbrandingsmotoren. Met de overgang naar elektrische voertuigen wordt de verbrandingsmotor (gedeeltelijk) vervangen door de elektrische motor en de benzinetank door een batterij waar elektrische energie opgeslagen wordt, die centraal in het net wordt opgewekt. Centrale energieopwekking is eenvoudiger te verduurzamen, iets waar de hedendaagse maatschappij druk mee bezig is.

Door bovenstaande transitie wordt de wereld minder afhankelijk van olie-exporterende landen, kunnen afspraken ten opzichte van de reductie van CO_2 worden gehaald en wordt opwekking uit duurzame energiebronnen gestimuleerd. Heden ten dage rijden er slechts 31 duizend elektrische auto's in Nederland [5], goed voor slechts 0,44% van alle voertuigen in Nederland. De grootste kritiek op elektrisch rijden is tweeledig: elektrische voertuigen zijn relatief duur in de aanschaf en de oplaadinfrastructuur is onvoldoende. Het infrastructuurprobleem werpt een muur op voor de doorontwikkeling van elektrisch vervoer.

De wereldwijde trend van duurzaamheid en milieubewustzijn is niet alleen in de voertuigsector te vinden. In de gebouwbeheersector wordt ook gezocht naar CO_2-reductie en efficiëntere gebouwen. De uitrol van integrale en intelligente gebouwbeheersystemen speelt hierin de belangrijkste rol [6]. Deze systemen maken bijvoorbeeld duurzame opwekking met zonnepanelen en opslag van energie in watertanks mogelijk. Zulke systemen vragen om intelligent beheer van energie. Wanneer werknemers collectief overstappen op elektrisch rijden, zullen oplossingen voor slim opladen nodig zijn [7]. Als veel auto's tegelijk opladen, ontstaan er namelijk grote verschillen in netbelasting door de dag heen. Deze verschillen kosten bedrijven geld, dus moeten ze voorkomen worden.

Deze thesis combineert de twee bovenstaande uitdagingen in één ontwerp. Het ontwerp draagt bij aan toegankelijke lokale infrastructuur voor het opladen van elektrische voertuigen met kostenbesparende oplaadpalen. Deze thesis zal zich richten op de benodigde software en logica om dit ontwerp te verwezenlijken.

1.1.1. De uitdaging

Hedendaagse elektrische voertuigen worden uitgerust met de modernste snufjes. Zo kunnen ze zich bijvoorbeeld bijna autonoom door het verkeer bewegen met een zo laag mogelijk verbruik. Bovendien zijn er steeds intelligentere gebouwen, waarin veel informatie verzameld en gekoppeld wordt om een zo zuinig en gebalanceerd mogelijke huishouding te voeren.

Deze twee intelligente systemen, de auto en het gebouw, raken elkaar in de oplaadpaal. Van zo'n oplaadpaal zou je dan ook verwachten dat die met de twee intelligenties om kan gaan. Hierdoor zou het opladen slim en efficiënt kunnen gebeuren.

Oplaadpalen van tegenwoordig blijken echter over geen intelligente stroomsturingen te beschikken. Als een gebouw bijvoorbeeld beschikt over duurzame energiewinning, zou het wenselijk zijn dat auto's sneller mogen opladen op het moment dat er meer duurzame energie opgewekt wordt. Met de producten die tegenwoordig op de markt zijn kan hier geen rekening mee gehouden worden. Ook lijken er geen slimme configuraties te zijn voor het opladen van meerdere elektrische voertuigen, waarbij voertuigen mogelijk langzamer zouden kunnen opladen, omdat ze langere tijd aangesloten staan. Oplaadpalen beschikken vaak wel over de hardware om zulke systemen te implementeren, maar de software is dan alleen voor betalingsverkeer ontwikkeld. De software zou een veel grotere rol kunnen spelen in een intelligentere vormgeving van de oplaadpalen.

1.2. Doelstelling van het onderzoek

Aan de hand van het probleem dat aangepakt moet worden is de volgende doelstelling op te stellen.

Hoofddoelstelling

> Een softwaresysteem ontwikkelen dat meerdere oplaadpalen voor elektrische auto's aanstuurt, zodat de auto's op tijd opgeladen zijn, waarbij een variabel maximaal beschikbaar vermogen niet overschreden wordt.

Subdoelstellingen

De hoofddoelstelling is tevens op te breken in subdoelstellingen. Deze worden apart van elkaar behandeld en leiden samen tot de hoofddoelstelling. In dit verslag zijn vier subdoelstellingen gespecificeerd:

1. Een algoritme ontwikkelen dat een slim oplaadschema voor de aangesloten elektrische voertuigen opstelt.
2. Kennis van de huidige en toekomstige waarde van maximaal beschikbaar vermogen verwerven.
3. Een interface voor gebruikers ontwerpen voor het invoeren en uitlezen van benodigde gegevens.
4. Communicatie met de hardware opzetten, voor het aansturen van en ontvangen van informatie over het opladen van aangesloten elektrische voertuigen.

De rest van het verslag is opgedeeld aan de hand van deze subdoelstellingen. Voor elke subdoelstelling worden onderzoeksvragen opgesteld, mogelijkheden besproken en implementaties ontworpen.

1.3. State-of-the-art

Om de reeds beschikbare mogelijkheden goed te kunnen benutten is een vooronderzoek uitgevoerd.

1.3.1. Peak shaving

Peak shaving is, zoals al door de benaming geimpliceerd wordt, een manier om pieken uit het energieverbruik te halen. Door een deel van deze pieken af te snijden en te verschuiven naar een moment van weinig energieverbruik wordt een uniformere verdeling bereikt.

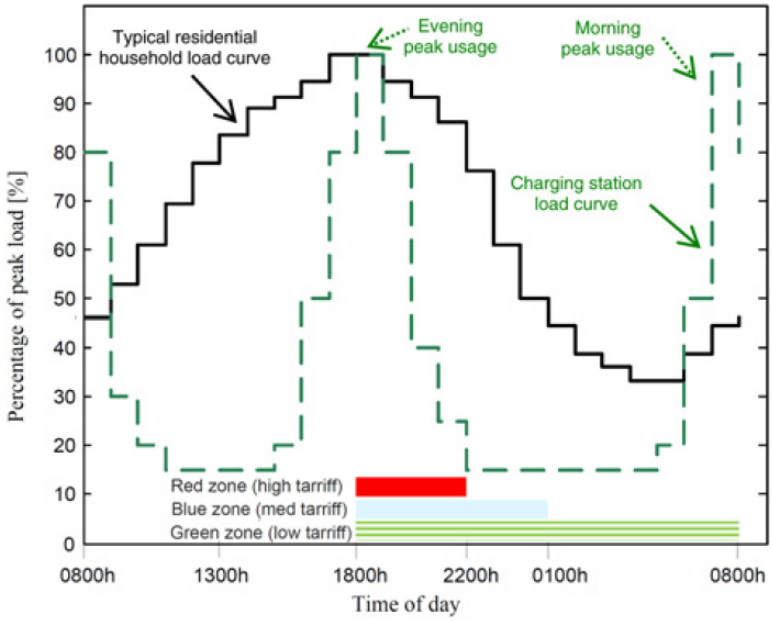
Waarom is peak shaving belangrijk?

Hoge pieken in het verbruik van het vermogen zal hoge stromen als gevolg hebben. Deze hoge stromen hebben als gevolg dat de transformatoren overbelast kunnen worden, waardoor de veroudering hiervan versnelt. Het verouderen van een transformator is een exponentiële functie van zijn temperatuur, welke op zijn beurt weer afhangt van de stroom die er doorheen loopt [8]. Daarom heeft een toename van piekverbruik door ongecoördineerd laadgedrag van elektrische voertuigen een grote invloed op het verouderen van de transformatoren die op het moment al erg belast worden. Naast de

transformatoren zullen ook kabels sneller verouderen. Door peak shaving kan het belastingsprofiel van deze componenten worden afgevlakt, met als gevolg dat de mechanische druk op deze componenten lager is en de kans kleiner wordt dat een van deze componenten zal falen.

Kostenbesparing

Wat voor veel gebruikers een nog belangrijkere beweegreden zal zijn om aan peak shaving te doen, is dat het geld bespaart [9]. Energie kost meer als de vraag ernaar stijgt, tijdens de piekuren dus. Dit is in huishoudelijke kringen duidelijk zichtbaar aan de hand van figuur 1.1. De verbruikte hoeveelheid energie is het grootst wanneer mensen terugkomen van hun werk. Als het oplaadstation op dat moment meteen zou beginnen met opladen, zal de al aanwezige piekbelasting nog veel hoger worden tijdens het moment dat het tarief voor het verbruik van energie het hoogst is. Hierdoor zijn de kosten een stuk hoger en deze moeten juist gedrukt worden.



Figuur 1.1: Grafiek van de dagelijkse belasting van huishoudens en oplaadstations [2]

Het model dat te zien is in figuur 1.1 kan geschaald worden naar het model van verbruik van energie bij bedrijven. De vorm zal namelijk gelijkaardig zijn, alleen wordt er natuurlijk veel meer energie verbruikt. De energie die naar het opladen van auto's gaat zal op andere tijdstippen plaatsvinden. Auto's komen namelijk bij bedrijven meestal ergens in de ochtend aan en zullen dan rond etenstijd vertrekken en moet er dus overdag opgeladen worden. Bij huishoudens is het probleem niet erg complex. Hier heeft men vaak te maken met maar één elektrisch voertuig waarbij de auto pas de volgende ochtend weer nodig zal zijn. Bij een bedrijf waar meerdere elektrische voertuigen tegelijkertijd moeten worden opgeladen, waarbij iedere auto een andere voorkeur heeft van vertrek met een andere state of charge, zal een prioriteitensysteem van belang zijn.

Prioriteitensysteem

Peak shaving is al eerder onderzocht [8]. Hierbij werd gebruik gemaakt van een algoritme dat alle betrokken auto's evalueert aan de hand van een prioriteitenformule:

$$P_r(E_t, t) = \frac{t - t_s}{t_f(E_t, t_d, P(E_t)) - t_s} \tag{1.1}$$

Hier is t_s de aankomsttijd en $t_f(E_t, t_d, P(E_t))$ het laatste moment waarop het opladen moet starten zodat het nog mogelijk is om een volledig opgeladen accu te hebben voor vertrek. De vertrektijd zal voorspeld moeten worden, want deze is niet bekend. Dan wordt er een bod gemaakt met deze

prioriteit in het programma Intelligator [8]. In dat programma wordt elke deelnemer in een virtuele vermogensmarkt gerepresenteerd door een 'agent' die een bod gaat maken in deze markt gebaseerd op zijn prioriteit. Dit is een interessante manier van kijken naar het probleem. Deze manier van analyseren en kwantificeren van de prioriteit van een bepaalde auto zal belangrijk zijn voor het algoritme dat opgesteld wordt in deze thesis.

Behalve een prioriteit aan de hand van de tijd, is het ook gewenst om naar het vermogen te kijken. Met name het beschikbare vermogen dat het gebouw te bieden heeft en het vermogen dat het voertuig nog nodig heeft om op te laden. Een prioriteitensysteem dat ook hiervan afhangt zal verder onderzocht moeten worden.

State of charge (SOC)

Het is niet mogelijk om bij alle elektrische voertuigen de state of charge uit te lezen [1]. Deze parameter zal echter wel een belangrijke rol gaan spelen in het oplaadalgoritme. Deze dient ook na elke tijdslot weer vernieuwd te worden [3]. Een methode om de state of charge te vernieuwen, als deze zich bevind tussen 10% tot 80%, wordt beschreven met de volgende formule (1.2).

$$soc_{new} = soc_{old} + [\frac{p^n_{actual}}{p_{lc} * 60} * 100] \tag{1.2}$$

In formule 1.2 wordt de nieuwe state of charge (soc_{new}) berekend door bij de oude state of charge het geleverde vermogen in de afgelopen minuut (in dit geval) op te tellen. Hierbij zal p^n_{actual} het actueel benodigde vermogen zijn en is p_{lc} het vermogen dat nodig zal zijn om de accu aan een snelheid van 1C (100% soc in een uur) op te laden [3]. Er wordt in de paper duidelijk gemaakt dat het opladen van de accu heel anders zal gaan lopen van 80% tot 100%, namelijk trager. Formule 1.2 zal op een gelijkaardige manier geïmplementeerd moeten worden in het algoritme aangezien het niet mogelijk zal zijn om de state of charge af te lezen uit alle elektrische voertuigen. Dit brengt moeilijkheden met zich mee aangezien de state of charge van een accu een niet-lineaire functie is, die dus slecht te benaderen zal zijn door een lineaire functie met een groot tijdsinterval. Aangezien de hardware groep de geleverde stroom aan de voertuigen gaat meten, is er met feedback van dit systeem wel een goede schatting mogelijk voor de gewijzigde state of charge. Hoe dit precies te implementeren zal nog onderzocht moeten worden.

1.3.2. Vermogensvoorspelling

Het voorspellen van het vermogen dat op een bepaald moment beschikbaar zal zijn of verbruikt zal worden, kan op verschillende manieren. Uit het vooronderzoek zijn twee methodes naar voren gekomen die hieronder kort besproken worden.

Voorspelling met huidig vermogen

In het onderzoek van [3] wordt een methode beschreven om de oplaadprofielen van de verschillende elektrische voertuigen te voorspellen. Deze voorspellingen worden gedaan aan de hand van huidige oplaadprofielen. Aangezien het ontbreekt aan historische data van laadgedrag van de verschillende elektrische voertuigen, worden deze gegenereerd aan de hand van formule 1.3.

$$p^n_{actual} = (\frac{p_{cc}}{\mu}) - (0.1 \cdot \frac{p_{cc}}{\mu}) + (W \cdot 0.1) \tag{1.3}$$

Hierbij zal p^n_{actual} het actueel benodigde vermogen zijn, p_{cc} is de vraag naar vermogen tijdens een gebied met constante stroom, W is een pseudo-willekeurige integer variabele die uniform verdeeld is over het interval [0,10], μ is de efficiëntie van de vermogensschakelaars en n is een moment in het oplaadinterval. Er wordt uitgegaan van een efficiëntie van de vermogensomzetters van ongeveer 85%. Hiermee zijn de oplaadprofielen gegenereerd en vervolgens wordt er gezocht naar een methode om de oplaadprofielen in de toekomst te gaan voorspellen. Een methode waarmee dit mogelijk is, wordt in het volgende onderdeel behandeld.

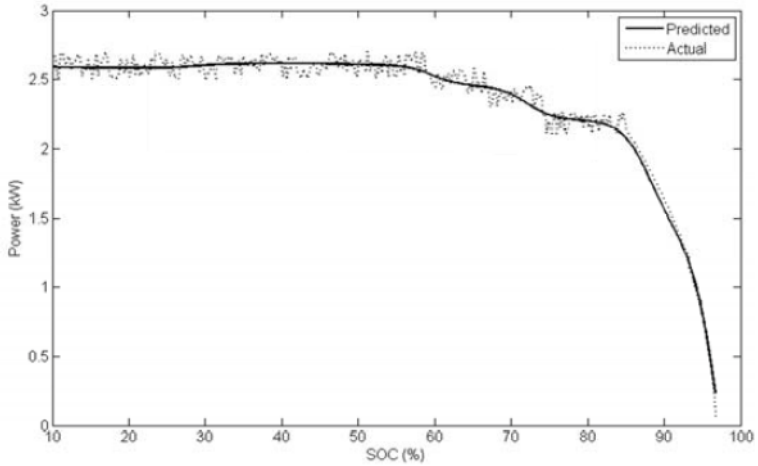
Artificial Neural Network

De state of charge van een accu is een niet-lineaire functie en verandert regelmatig door zijn chemische samenstelling. Het Artificial Neural Network (ANN) is een veelgebruikt algoritme in de computerwetenschappen en andere gerelateerde studies. Het is geïnspireerd door het centrale zenuwsysteem in de

hersenen van dieren en kan zowel leren uit voorgaande situaties als patronen herkennen. Daarom is dit model dus erg handig om de oplaadprofielen van elektrische voertuigen te voorspellen [3] [9]. Het voorspelde vermogen P_F werd tijdens de simulatie vergeleken met het gemeten actuele vermogen P_A aan de hand van de Mean Absolute Percentage Error (MAPE). Deze is steeds binnen het accepteerbare resultaat met een RMS-afwijking van ongeveer 3%. De MAPE wordt volgens formule 1.4 berekend.

$$MAPE = \frac{1}{n}\sum_{i}^{n}\frac{abs[P_F(i) - P_A(i)]}{P_A(i)} \cdot 100\% \tag{1.4}$$

In figuur 1.2 is de simulatie te zien van het voorspelde vermogen en het gemeten actuele vermogen.



Figuur 1.2: Resultaten van de voorspelling van het vermogen van de ANN module [3]

1.3.3. Energieuitwisseling gebouwbeheersysteem en elektrische voertuigen

Het optimalisatieprobleem is het minimaliseren van de kosten. Dit probleem is natuurlijk afhankelijk van het gedrag van het verbruik van de energie. Pieken vermijden is het belangrijkste om aan kostenbesparing te doen, een vlak energieverbruik is dus wenselijk. Met Vehicle-to-Grid (V2G) systemen kan valley-filling (en peak shaving) gedaan worden om een wenselijke energieverbruik te verkrijgen [10] [11] . Dit roept echter wat beperkingen op, deze zullen in het onderdeel Constraints behandeld worden. Het is duidelijk dat er aan energieuitwisseling tussen elektrische voertuigen en slimme netwerken te doen is. Het is nu nog de vraag of de voordelen zullen opwegen tegen de beperkingen van het systeem.

Constraints

Het V2G-systeem is zeer afhankelijk van de mogelijkheden van de accu's van de elektrische voertuigen. De levensduur van een accu wordt beïnvloed door de diepte van ontlading, het aantal laadcycli, op- en ontlaadsnelheden, etcetera [10]. Ook hebben consumenten vaak de behoefte om voertuigen zo snel mogelijk op te laden, zodoende de autonomiteit van het voertuig te vergroten. Het is duidelijk dat de grootste beperkingen voor de V2G-systemen zowel de levensduur van een accu als de oplaadtijd zijn. Naast deze beperkingen, zullen er ook beperkingen zijn op het vermogen van de elektrische voertuigen in het V2G-systeem. Er zit een limiet per EV op het op- en ontlaadbare vermogen. Ook kan de EV niet meer vermogen aftappen dan beschikbaar is in het netwerk.

Afweging

De grootste beperking zal de accu van het elektrische voertuig zijn. De meeste elektrische voertuigen hebben tegenwoordig een Lithium-ion accu. Bahalve dat deze een niet erg lange levensduur heeft, tussen de 400 en 1200 op- en ontlaadcycli, is deze ook nogal fragiel [12]. Daarom heeft deze accu een beschermcircuit zodat deze veilig gebruikt kan worden. Het is voorlopig een te grote uitdaging om deze type accu's actief te ontladen. Om deze redenen wegen de voordelen van het valley-filling niet op tegen

de restricties die de huidige technologie teweegbrengt. Het actief ontladen blijft voorlopig theoretisch een plausibele optie. Dit kan meegenomen worden naar een volgend onderzoek, dit zal steeds een interessantere manier van aanpak worden naar gelang de levensduur van de accu's verbetert [13]. In dit onderzoek zal gericht worden op het gecoördineerd opladen van elektrische voertuigen en niet het ontladen.

1.3.4. Interface voor het opladen

Een elektrische auto is een product dat direct door de eindgebruiker gebruikt wordt. Vaak is de auto persoonlijk eigendom of is de eindgebruiker verantwoordelijk voor de auto. De gebruiker is er dus zeer bij gebaat als hij/zij constant op de hoogte is van de activiteiten van de auto. Bij traditionele voertuigen kan je bijvoorbeeld alleen "opladen" als je bij een tankstation de slang aansluit. Bij een elektrische auto is fysieke controle over het opladen al minder vanzelfsprekend, omdat het opladen bij gangbare oplaadpalen uren duurt. Een ander controlekanaal is dus wenselijk.

Gebruikersinterface met de auto

De smartphone is een uitstekend medium om het nieuwe controlekanaal vorm te geven. Dit zien we ook terug in het grote aanbod van mobiele applicaties (apps) die geleverd worden bij elektrische voertuigen [14][15][16][17][18][19]. Alle apps bieden minstens inzicht in de basisstatussen van de auto, zoals oplaadniveau en (gemiddeld) verbruik. Sommige apps, zoals die van Volvo [19] en Nissan [16] maken van de app een volledige afstandbediening waarmee bijvoorbeeld het klimaat te beheersen is.

Het is belangrijk om deze informatie mee te nemen in het onderzoek. Gebruikers zitten bijvoorbeeld niet te wachten op meerdere interfaces voor dezelfde activiteit. Bovendien zouden besturingsmogelijkheden van de app de prestaties van een slim oplaadsysteem kunnen benadelen. In de app van Mitsubishi [15] is het bijvoorbeeld mogelijk om het opladen van de auto in te roosteren.

Gebruikersinterface met oplaadsystemen

Niet alleen autofabrikanten, maar ook andere stakeholders in de elektrische auto-industrie proberen de gebruikers te bedienen met applicaties. Vooral de exploitanten van oplaadpalen bieden met apps de mogelijkheid om oplaadplekken te vinden [20][21]. De site oplaadpalen.nl lijkt daarbij de koploper te zijn. Deze apps tonen alleen inzicht in beschikbaarheid, en niet in oplaadtijd of laadtoestand.

OBD-II, communicatieprotocol met de auto

Wat een wenselijke functie van het beoogde product zou zijn, is autonome communicatie met het elektrische voertuig. Op die manier zou vitale informatie over het voertuig direct uit te lezen zijn. In het onderzoek beschreven in [22] spreken ze over het uitlezen van het voertuig via een CAN-bus. In het onderzoek gaan ze ervan uit dat onder andere de state of charge en verstreken oplaadtijd uit te lezen zijn. De technische informatie hierover ontbreekt echter. Uit verder onderzoek komt het OBD-II protocol naar voren, een standaard voor het uitlezen van voertuiginformatie [23]. Echter blijkt er nog geen standaard ondersteuning te zijn voor het uitlezen van informatie specifiek voor elektrische voertuigen. Elk elektrisch voertuig heeft zijn eigen of geen ondersteuning voor informatieuitwisseling over OBD-II.

1.3.5. Communicatie oplaadsysteem met smart grids

Er een slimme koppeling nodig tussen twee al slimme partijen: de elektrische auto en het systeem waar het op aangesloten is [24]. Dit systeem kan van verschillende vormen zijn, en wordt, als het slim energiebeheer bevat, vaak een smart grid genoemd. Hoewel die slimme koppeling nog niet commercieel toegepast lijkt te worden, is er waarschijnlijk wel al onderzoek naar gedaan.

Internet of Energy, informatievoorziening van het elektriciteitsnet

Met de opkomst van decentrale energieopwekking is ook de vraag gegroeid naar decentrale informatievoorziening [25] [26]. Het traditionele elektriciteitsnet gaat in de tegenwoordige tijd niet meer op. Er zijn veel, meestal kleine duurzame energiebronnen. Bovendien is hun opgewekte vermogen onderhevig aan onvoorspelbare factoren zoals het weer. Systemen zullen onderling moeten gaan communiceren om in balans te blijven. Door de brug te slaan tussen het opladen van auto's en het beheren

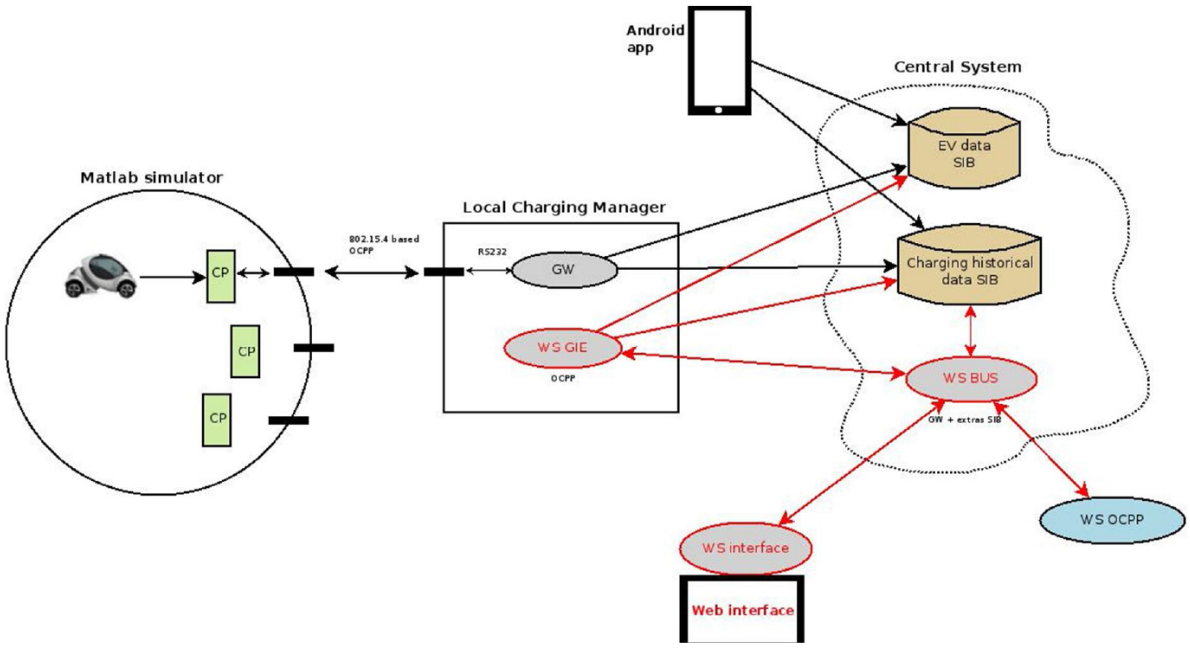
van gebouwen wordt er weer een schakel toegevoegd aan het Internet of Energy.

OCPP, communicatie tussen oplaadpaal en beheersysteem

Sinds kort is SAE J1772 aangenomen als standaard voor communicatie tussen het elektrische voertuig en de oplaadpaal. Er is echter nog geen standaard voor communicatie tussen de oplaadpaal en het smart grid. Hier zijn al enkele oplossingen voor [4], zoals

- ChargePoint Network of EV charging stations, van Coulomb Technologies
- GridPoint, een systeem om de stroom tussen het net en de auto te beheren
- OCPP, Open Charge Point Protocol, een communicatie-methode voor oplaadpalen met een centraal systeem.

In het onderzoek in bron [4] worden de IEEE 802.15.4 standaard en OCPP via SOAP/TCP gebruikt. Het onderzoek richt zich vooral op het vormgeven van de infrastructuur en niet op de informatieverwerving. Zij geven de infrastructuur vorm zoals in figuur 1.3 weergegeven is.



Figuur 1.3: Schematische weergave van het systeem uit bron [4]

In het diagram van figuur 1.3 zijn de volgende onderdelen te onderscheiden:

- Communicatie met OCPP en XML
- Het datamodel en communicatie-infrastructuur met Smart_M3 opgezet
- Een app voor gebruikers om aan te melden aan het oplaadsysteem en hun oplaadstatus uit te lezen
- Een database om oplaadgeschiedenis in bij te houden
- Een web-interface om statistieken uit te lezen

Uiteindelijk hebben ze hun systeem alleen gesimuleerd en daarmee aangetoond dat open source software geschikt is voor het ontwikkelen van een slim oplaadsysteem. Ze hebben echter niet gekeken naar bijvoorbeeld het werkelijk uitlezen van de state of charge van de voertuigen. De werkelijke implementatie van het voorgesteld systeem wordt niet behandeld.

Woonwijken

De groei van het aantal elektrische auto's heeft de grootste invloed op het woon-werkverkeer. Dit is verreweg waar de auto de grootste rol speelt. Het ongecontrolleerd opladen van elektrische voertuigen zal dus 's ochtends een piek geven bij het arriveren van de auto's op de werklocatie, en 's avonds een piek bij het retourneren van de voertuigen. Het onderzoek van [22] concentreert zich op het tweede probleem.

Omdat oplaadpalen in woonwijken decentraal zijn opgesteld, wordt er in het onderzoek een grote nadruk gelegd op draadloze communicatie. Verder wordt een structuur voorgesteld om het gehele informatiesysteem vorm te geven. Zij identificeren de volgende onderdelen

- Een Power Dispatching Center, dat aangeeft hoeveel vermogen er beschikbaar is. Dit is via het internet verbonden met het EVCMS.
- Een Electric Vehicles Management System (EVCMS) dat het oplaadschema bepaalt.
- Electric Vehicle Charging Controllers (EVCC's) die informatie van de auto uitleest en doorgeeft aan de EVCMS, en opdrachten van de ECVMS uitvoert. Deze zijn via GPRS verbonden met het EVCMS.

Om de belasting aan het net door het opladen te verspreiden, stellen ze voor om de voertuigen op volgende tijdsperiodes te geven om op de laden.

Om het oplaadgedrag van de auto's te voorspellen, communiceren ze met het batterij management-systeem van de auto via CAM-net over het nieuwe SAE J1939-protocol [22]. Vervolgens kunnen ze aan de hand van de nominale spanning, nominale capaciteit, initiële state of charge en interne weerstand het oplaadgedrag meenemen in het oplaadschema.

1.4. Onderzoeksvragen

Nadat het probleem is geïdentificeerd en de huidige stand van zaken binnen de oplossingsruimte van het probleem is onderzocht, hebben we duidelijke onderzoeksvragen kunnen opstellen. Elke set onderzoeksvragen correspondeert met een subdoelstelling, zoals opgesteld in sectie 1.2, met uitzondering van de laatste onderzoekvraag. Dit is een vraag betreffende de integratie van alle subonderdelen.

1. Het algoritme
   (a) Welke informatie is nodig om een goed algoritme te kunnen maken?
   (b) Hoe kan het algoritme het opladen van de auto's zo slim mogelijk indelen?
2. Vermogensberekening
   (a) Hoe kan het oplaadsysteem kennis krijgen van momentele energiebeschikbaarheid?
   (b) Hoe kan een nauwkeurige schatting gemaakt worden van het toekomstig beschikbaar vermogen?
3. Gebruikersinterface
   (a) Welke informatie moet de eindgebruiker ingeven?
   (b) Hoe kan de interface het beste vormgegeven worden?
   (c) Wat is de beste manier om informatie terug te koppelen naar de eindgebruiker?
4. Aansturing
   (a) Hoe kan de aanstuurinformatie en de feedback daarvan het beste gecommuniceerd worden?
   (b) Op welke manier kan er toch aansturing plaatsvinden als communicatie wegvalt?
5. Hoe kunnen alle onderdelen het beste geïntegreerd worden tot één geheel systeem?

1.5. Thesis-indeling

In de rest van de thesis zullen we per onderdeel van het systeem, zoals gedefinieerd aan de hand van de doelstellingen, het ontwerp behandelen.

Alvorens de onderdelen te behandelen, wordt in hoofdstuk 2 een plan van aanpak opgezet, wat bestaat uit een programma van eisen en enkele randvoorwaarden.

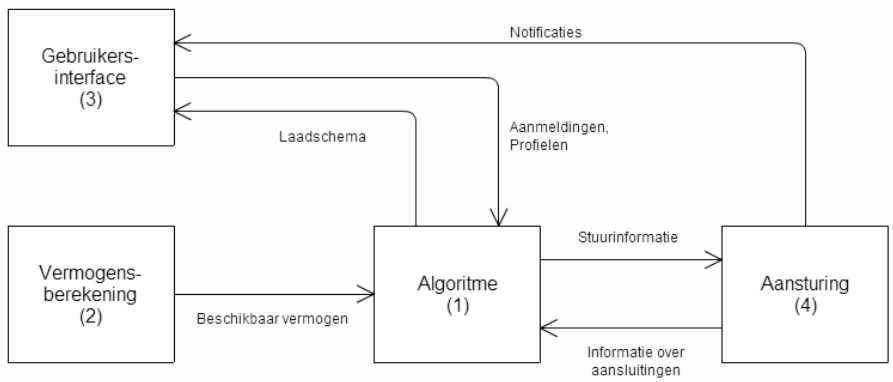
Vervolgens wordt in hoofdstuk 3 het eerste onderdeel, het algoritme, behandeld. In dit hoofdstuk wordt ook behandeld welke informatie er nodig zal zijn van de andere onderdelen. In hoofdstuk 4 zal dan de vermogensberekening onderzocht worden. Vervolgens wordt de interface van het systeem in hoofdstuk 5 vormgegeven.De aansturing zal behandeld worden in hoofdstuk 6. In hoofdstuk 7 zal de laatste onderzoeksvraag beantwoord worden, waarbij alle subonderdelen samengevoegd worden tot het totaalsysteem. In hoofdstuk 8 zullen de conclusie en aanbevelingen geformuleerd worden.

2

Plan van aanpak

2.1. Algemeen overzicht van het product

De introductie behandelde kort wat precies de grote blokken zullen zijn van het systeem. Nu de eisen aan het systeem gesteld zijn, is het mogelijk om een schematisch overzicht van het product op te zetten. Dit overzicht is te zien in figuur 2.1.



Figuur 2.1: Uitgebreid overzicht van het product, elk van deze pijlen zijn informatieflows

2.2. Programma van eisen

In hoofdlijnen is het ontwerp nu duidelijk. Bij het opstellen van de doelstellingen is kort samengevat wat het product zal moeten kunnen. Nu moet bepaald worden aan welke eisen het systeem precies moet voldoen.

2.2.1. Het algoritme

1. Het algoritme moet de gebruikersinformatie die nodig is voor het berekenen van het laadschema vanuit de interface (3) kunnen uitlezen.
   (a) Er moet bekend zijn aan welke paal een auto is aangesloten.
   (b) Per aansluiting moet bekend zijn hoe lang de auto kan opladen, en hoeveel deze opgeladen moet worden.
   (c) Voor planning moet bekend zijn wanneer er nog auto's aangesloten zullen worden.
2. Vanuit de vermogensberekening (2) moet informatie beschikbaar zijn over het te verdelen vermogen, zowel voor het moment van de berekening als de rest van de dag.

3. De berekening van het oplaadschema moet geïmplementeerd worden met de volgende eisen:
   (a) Auto's die aangesloten zijn, maar ook auto's die nog aan zullen komen, moeten meegenomen worden in het oplaadschema.
   (b) Het beschikbare vermogen mag niet worden overschreden (peak shaving).
   (c) Het algoritme moet robuust zijn en specifiek rekening houden met de volgende scenario's:
      i. Er is meer vermogen beschikbaar dan er nodig is. In dat geval moet de piekstroom zo laag mogelijk zijn, rekening houdend met het risico dat auto's niet volledig opgeladen raken (load balancing).
      ii. Er is minder vermogen dan de auto's nodig hebben om volledig opgeladen te zijn. Dan moet het vermogen zo eerlijk mogelijk verdeeld worden.
      iii. Er zijn meer auto's dan oplaadplekken. In dat geval moet er zodanig ingepland worden dat op een gegeven moment voertuigen moeten wisselen (wachtrij).
4. Bij een statusverandering vanuit de aansturing (4) moet het algoritme herberekend worden.
5. Na het verstrijken van een te bepalen tijdsinterval moet het oplaadschema herberekend worden.
6. Als het oplaadschema herberekend is, moet dit doorgegeven worden aan de aansturing (4).

2.2.2. Vermogensberekeningen

1. De berekeningen moeten het huidige beschikbare vermogen bepalen.
   (a) Als het mogelijk is wordt dit uitgelezen uit een gebouwbeheersysteem.
   (b) Anders moet er een betrouwbare schatting in het algoritme zitten.
2. De berekeningen moeten het toekomstig beschikbare vermogen kunnen schatten op de termijn van minstens één dag.
   (a) Dit wordt bereikt door een profiel te maken van een gemiddelde dag.
   (b) Er moet onderzocht worden hoe een afwijking van het huidig beschikbaar vermogen ten opzichte van het profiel doorgerekend moet worden in de schatting.

2.2.3. Gebruikersinterface

1. Alle gebruikerinformatie die nodig is voor het algoritme (1) moet via de interface verzameld kunnen worden.
2. Er moet een procedure komen waarmee gebruikers zich aan kunnen melden.
   (a) Er zal een interface nodig zijn om informatie handmatig in te voeren.
   (b) Een aanmelding moet doorgegeven worden aan het algoritme (1).
3. Bepaalde gebruikersinformatie zal niet of nauwelijks aangepast moeten worden, zodra deze ingevoerd is. Hier moet ook een interface voor komen.
   (a) Er moeten gebruikersprofielen aangemaakt kunnen worden.
   (b) Per geregistreerde gebruiker moet globaal of specifiek aangegeven worden wat zijn aanwezigheid per dag zal zijn.
4. De gebruikersinterface moet proactief gebruikers op de hoogte stellen van statusveranderingen van hun eigen voertuig.
5. Voor inzichtelijkheid en overzicht moet de interface ook algemene informatie verschaffen.
   (a) Het oplaadschema van die dag moet inzichtelijk zijn.
   (b) Er moet duidelijk zijn welke voertuigen aangesloten zijn en wat de status van die voertuigen is.
   (c) Ook moet te zien zijn wie er nog verwacht wordt.
   (d) Een overzicht van de totale oplaadtijden moet in te zien zijn.

2.2.4. Aansturing

1. De aansturing staat direct in verbinding met de hardware van het complementaire onderzoek [1]. Voor elk aansluitpunt zal informatieverkeer zijn.
   (a) Informatie uit het algoritme (1) moet doorgegeven worden aan de hardware.
   (b) Informatie updates vanuit de hardware moeten doorgegeven worden aan het algoritme (1).
2. Vanuit Priva is de wens om de aansturing met hun eigen systemen te realiseren. Priva ontwikkelt namelijk algemene software en hardware voor communicatie binnen gebouwen.
3. Statusveranderingen van auto's moeten herkend kunnen worden, waarbij een gepaste handeling volgt.
4. Er moet een optie voor handmatige besturing zijn, waarbij de aansturing vanuit het algoritme (1) overschreven wordt.

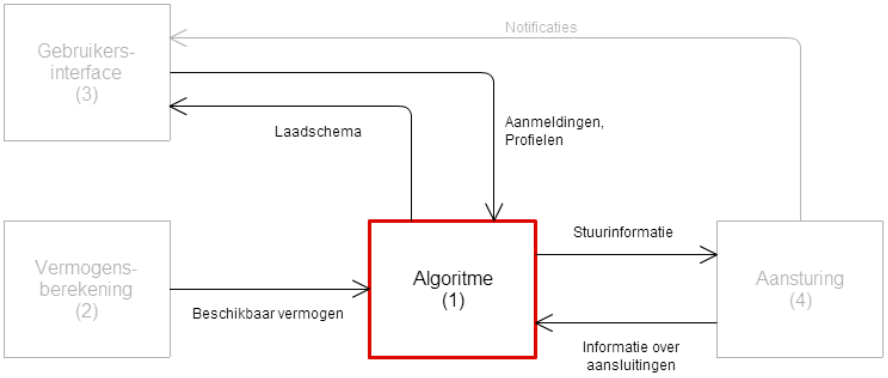
2.3. Randvoorwaarden

Omdat het project wordt uitgevoerd bij een externe opdrachtgever, zijn er ook enkele randvoorwaarden die opgelegd zijn door Priva. Bovendien bevindt het project zich in een kader van in totaal twee onderzoeken. Er moet dus ook rekening gehouden worden met de verbindingen met het andere onderzoek. Samengevat:

- Gebruik maken van Priva regelcomputer als communicatiesysteem met de hardware
- Rekening houden met limitaties van de stroom per auto, per oplaadpaal en voor het gehele systeem. De limitaties worden bepaald door het andere onderzoek bij dit product [1].

3

Het algoritme



Figuur 3.1: De positie van het algoritme in het gehele systeem

3.1. Inleiding

Zoals in figuur 3.1 te zien is, is het algoritme het belangrijkste onderdeel van de software van het systeem. Alle andere elementen zijn erop aangesloten en vaak ook afhankelijk. Het is dus van belang dat het algoritme stabiel en robuust vormgegeven wordt.

Vanuit het programma van eisen zijn er enkele vereisten opgelegd gekregen. In hoofdlijnen moet een maximaal beschikbaar vermogen eerlijk verdeeld worden onder aangesloten auto's, waarbij rekening gehouden moet worden met stroomlimieten per aansluiting. Deze verdeling zou tevens zo eerlijk mogelijk over de dag verdeeld moeten worden.

In dit hoofdstuk wordt allereerst bepaald welke informatie er nodig zal zijn voor een goed algoritme. Vervolgens wordt het algoritme stapsgewijs vormgegeven en getest.

3.2. Mogelijkheden voor het algoritme

"Hoe kan het algoritme het opladen van de auto's zo slim mogelijk indelen?" is de onderzoeksvraag. Hierin wijst "zo slim mogelijk" op een optimalisatie. Gezien er veel factoren meespelen bij het opladen, betreft het een gecompliceerde optimalisatie. Belangrijke factoren zijn onder andere het aantal auto's, de benodigde energie, de snelheid van opladen, het beschikbare vermogen en de beschikbare tijd. Bovendien is het een tijdsafhankelijk proces: tijdens het opladen verandert de state of charge van de auto's, waardoor de situatie ook verandert.

Wiskundige optimalisatie

Om een probleem wiskundig op te kunnen lossen moet deze in formules beschreven kunnen worden. Voor het voorgeschreven probleem zou voor alle n-aantal auto's een formule van het opgeslagen vermogen tegen te tijd opgesteld kunnen worden, waarbij het uiteindelijk opgeslagen vermogen gemaximaliseerd moet worden. Bovendien kan van elk van de m-aantal oplaadtijdsintervallen een formule opgesteld worden van de som van alle oplaadstromen in een interval, dat geminimaliseerd moet worden. Dit maximaliseren en minimaliseren spreekt elkaar tegen, dus wiskundige optimalisatie kan niet het volledige probleem oplossen.

Knapzakprobleem

Een andere manier om tegen het op te lossen probleem aan te kijken is op de manier van het knapzakprobleem. Het knapzakprobleem definieert het volgende: welke verzameling van objecten, elk met een gewicht en waarde, kan het beste meegenomen worden, zodanig dat het totale gewicht onder een bepaald limiet blijft en de meegenomen waarde zo groot mogelijk is.

In zekere zin komt dit overeen met het op te lossen probleem. Auto's (de objecten) hebben een gewicht: hoeveel vermogen gebruiken, en een waarde: hoezeer ze opgeladen moeten worden. Het totale vermogen heeft bovendien een limiet. Echter, het knapzakprobleem gaat ervan uit dat het gewicht van de objecten niet kan veranderen, terwijl bij dit probleem de auto's in principe ook langzamer kunnen opladen. Bovendien wordt er in het knapzakprobleem gesproken over het wel of niet meenemen van objecten. Omdat de oplaadsnelheid van de auto's in te stellen is, is er eigenlijk sprake van het gedeeltelijk meenemen van objecten.

Afweging

Bestaande technieken bieden slechts een gedeeltelijke oplossing voor het voorgelegde probleem. Er moet dus op zoek gegaan worden naar een algoritme dat beide bovenstaande theorieën samenvoegt tot een algoritme om het probleem zo slim mogelijk aan te pakken.

3.3. Specificaties

Als alle eisen aan het algoritme op een rijtje worden gezet, ontstaat er een functionele beschrijving van het algoritme zoals in tabel 3.1 weergegeven. Elke keer als het algoritme doorlopen wordt, moeten alle stappen die in de tabel staan doorlopen worden.

Tabel 3.1: Functionele beschrijving van het algoritme. (a) is het inlezen van benodigde informatie, In (b) wordt het oplaadschema voor de dag bepaald. In (c) worden resultaten opgeslagen en verstuurd.

Het algoritme		
a	Informatie aangesloten auto's ophalen	
	Beschikbaar vermogen ophalen	
b	Bepalen welke auto's (mogen) opladen	
	Bepalen met hoeveel stroom de auto's opladen	
	Status auto aan eind van tijdslot bepalen	
	Voor elk tijdslot nu en de rest van de dag	
c	Resultaten wegschrijven	
	Nieuwe instructies versturen	

Het algoritme wordt uitgevoerd als er nieuwe aansturingparameters nodig zijn. Bij het aanroepen van het algoritme moet informatie opgehaald worden van auto's die betrokken zijn bij de planning. Dit onderdeel wordt in sectie 3.4 behandeld. Vervolgens is het hart van het algoritme op te delen in twee delen: peak shaving, als er minder vermogen op een arbitrair moment is dan dat er nodig is, en load balancing, als er over de gehele dag meer vermogen beschikbaar is dan dat er nodig is. Deze twee onderdelen worden respectivelijk in sectie 3.8 en 3.9 beschreven. Tot slot, als bekend is welke waardes doorgegeven moeten worden aan de aansturing, wordt in sectie 6 behandeld hoe de laadinstructies gestuurd worden.

3.4. Benodigde informatie voor het algoritme

Voordat het algoritme opgezet kan worden, is een datamodel nodig. Het is dus van belang om duidelijk te hebben welke informatie nodig is om een goed oplaadschema te maken. Tevens moet er rekening gehouden worden dat iedere keer als het algoritme aangeroepen wordt, deze een planning moet kunnen maken voor de rest van de dag. Dit is van belang, omdat het verwachte resultaat van het opladen van die dag bepaald moet worden.

Vermogen

De grootheid waarin het algoritme werkt is energie. Er moet energie in de accu's van de auto's komen, en dit komt vanuit het lichtnet. De overbrenging van deze energie kost tijd, waarbij de overbrenging uitgedrukt kan worden in vermogen. Volgens het programma van eisen is er een maximaal beschikbaar vermogen, dus het algoritme moet op de hoogte zijn van het maximale beschikbare vermogen op het moment van uitvoeren en voor de rest van de dag. Het onderdeel *Vermogensberekening*, beschreven in hoofdstuk 4, moet voor deze informatie zorgen. Hoe de informatie tot stand komt zal dan ook in het betreffende hoofdstuk beschreven worden.

Door ontwerpkeuzes in de hardware van het systeem zijn er ook limitaties aan de snelheid van het opladen van individuele voertuigen [1]. In het huidig ontwerp is er een stroomlimiet van 32 ampère per fase-aansluiting. Twee auto's delen een 3-fasekabel, dus als de twee auto's beide 1-fasig opladen en op dezelfde fase zitten, hebben ze met z'n tweeën 32 ampère. Als ze allebei op een andere fase zitten, is het 32 ampère voor iedere auto. Het is dus voor het algoritme van belang om te weten of een auto zijn fasedraad moet delen. Deze informatie moet van onderdeel *Aansturing* (hoofdstuk 6) komen. Om de complexiteit van het probleem te drukken, kan een versimpeling gemaakt worden van de situatie: als twee auto's een fase delen, krijgen ze allebei maximaal 16 ampère, anders ieder 32 ampère.

Bovendien is gebleken dat de auto met niet minder dan 3 ampère aangestuurd kan worden, en dat de aansturing op twee decimalen nauwkeurig functioneert [1]. Deze gegevens kunnen samengevat worden in tabel 3.2.

Tabel 3.2: Eigenschappen van het stuursignaal

Eigenschap	**Symbool**	**Waarde**
Maximale stuurstroom	I_{stuur}^{max}	$16A$ bij gedeelde fase anders $32A$
Minimale stuurstroom	I_{stuur}^{min}	$3A$
Resolutie	ΔI_{stuur}	$0,01A$

Energie

De snelheid waarmee auto's opgeladen worden is hoofdzaak van het algoritme, maar om een goede planning te maken is het tevens van belang om op de hoogte te zijn van de totale hoeveelheid energie die de auto nodig zal hebben om weer vol te raken. De bedoeling is namelijk dat alle auto's vol komen te zitten tegen de tijd dat ze weer weg moeten.

Uit het state-of-the-art onderzoek bleek dat elektrische auto's de state of charge (de relatieve mate waarin een accu opgeladen is) gebruiken als indicatie voor oplaadstatus. Gezien dit relatief is, is het van belang om te weten wat de volledige accucapaciteit van een auto is. Als dan de state of charge op een bepaald moment bekend is, kan de nog benodigde energie berekend worden.

Uit het onderzoek van de hardware [1] is gebleken dat het niet haalbaar is om de state of charge direct van de auto uit te lezen. Er moet dus een andere manier gebruikt worden om de state of charge van een auto te bepalen. Vanuit de *Aansturing* is het wel mogelijk om via de hardware van het systeem op te vragen hoeveel energie een aangesloten auto ontvangen heeft sinds deze aangesloten is. Als dan via de *Interface* doorgegeven is wat de initiële state of charge van de auto is, kan met de volgende formule de benodigde hoeveelheid energie bepaald worden:

$$E_{nodig}[t] = E_{totaal} \cdot (1 - SOC_{initieel}/100\%) - E_{geleverd}[t] \tag{3.1}$$

Waarbij E energie is, uitgedrukt in $Joule$ $[J]$. $E_{nodig}[t]$ is de hoeveelheid energie die een auto nog nodig zal hebben op tijdstip t. Dit is afhankelijk van enkele variabelen. E_{totaal} is de totale opslagcapaciteit van de auto. $SOC_{initieel}$ is de state of charge van een voertuig zodra deze aangesloten wordt. $E_{geleverd}[t]$ is de hoeveelheid energie dat geleverd is aan het voertuig sinds deze aangesloten is. Deze waarde moet opgehaald worden uit de *Aansturing*. E_{totaal} en $SOC_{initieel}$ komen vanuit de *Interface*.

Tijd

Om alle berekeningen rond te kunnen krijgen, moet naast vermogen en energie ook tijd bekend zijn. In dit geval de tijd die auto's maximaal hebben om op te laden. Deze tijd komt overeen met de tijd dat een auto aangesloten is. Per voertuig moet dus bekend zijn wanneer deze aan zal komen en wanneer deze zal vertrekken. Dit zou op verschillende manieren bepaald kunnen worden.

Kunstmatige intelligentie Informatie van aankomst- en vertrektijden zou bijvoorbeeld met kunstmatige intelligentie verzameld kunnen worden. Aankomst- en vertrektijden vanuit het verleden worden dan gebruikt om toekomstige tijden te kunnen voorspelen. Het voordeel hiervan is dat de eindgebruiker geen informatie handmatig hoeft op te geven over zijn/haar aanwezigheid. De kunstmatige intelligentie zal echter alleen patronen kunnen herkenen, waardoor bijvoorbeeld geen rekening gehouden wordt met incidentele vrije dagen.

Handmatig ingevoerde waardes Het is ook mogeljjk om van gebruikers van het systeem te vragen om hun aanwezigheid van tevoren door te geven. Dit zou dan via de *Interface* moeten. Eindgebruikers zijn dan iets meer tijd kwijt aan het gebruik van het systeem, maar er zou dan wel betrouwbare informatie beschikbaar zijn over het verwacht gebruik van het systeem.

Zeker gezien de complexiteit van het programmeren van kunstmatige intelligentie voor zo'n klein onderdeel van het systeem, gaat de voorkeur uit naar het handmatig invoeren van waardes. De *Interface* van het systeem moet het algoritme dus kunnen voorzien in aankomst- en vertrektijden van betrokken voertuigen.

Conclusie

Dus, concluderend, heeft het algoritme iedere keer als het draait de volgende informatie nodig:

1. Maximaal vermogen op moment van berekenen en de rest van de dag.
2. Informatie van voertuigen die aangesloten staan, namelijk:
   (a) De maximale stroom waarmee opgeladen kan worden
   (b) Totale accu-capaciteit
   (c) Initiële state of charge
   (d) Hoezeer de auto al is opgeladen sinds deze aangesloten is
   (e) Waarschijnlijke vertrektijd

3. Informatie van voertuigen die aan zullen komen:

   (a) Totale accu-capaciteit

   (b) Verwachte initiële state of charge

   (c) Waarschijnlijke aankomsttijd

   (d) Waarschijnlijke vertrektijd

3.5. Uitvoer van het algoritme

Het uiteindelijke doel van het algoritme is om te bepalen hoe snel de aangesloten auto's mogen opladen. Het algoritme gebruikt hiervoor gegevens uit het verleden en heden en idealiter ook van de toekomst. Dit is in de vorige sectie beschreven. Het resultaat slaat logischerwijs alleen op het heden en de toekomst. Het resultaat voor het heden wordt gebruikt om het systeem mee aan te sturen. Het resultaat voor de toekomst moet inzicht bieden in het te verwachten resultaat.

Voordat het resultaat van het algoritme opgevolgd wordt, moet dat geïnterpreteerd worden door de hardware van het systeem. In overleg met het ontwerpteam van de hardware is dus ook gezocht naar welke informatie doorgegeven moet worden. Hoe deze informatie via de *Aansturing* doorgegeven wordt, wordt overigens beschreven in hoofdstuk 6.

Het stuursignaal dat de hardware door kan geven aan de auto, geeft aan met hoeveel ampère de auto mag opladen [1]. Het gemakkelijkste voor de verwerking van de stuursignalen vanaf het algoritme is dus als deze ook in ampère doorgegeven worden. Bovendien is de hardware niet op de hoogte van welke auto aan welke aansluiting staat. Het is dus aan het algoritme om de stroomwensen per aansluiting door te geven, niet per auto.

De tweede uitvoer van het algoritme moest inzicht geven in het gehele oplaadschema van de dag. Dit zou via de *Interface* teruggekoppeld worden naar gebruikers van het systeem. Alle ruwe data van de planning moet dus teruggestuurd worden naar de interface. Hoe dit gebeurt, wordt beschreven in hoofdstuk 5.

3.6. Implementatie

Het hoogste abstractieniveau van software is een black box. In zekere zin is er nu ook een black box-model opgezet voor het algoritme. Het is duidelijk welke informatie het algoritme in gaat en wat er uitkomt. Nu is het dus zaak om deze black box in te vullen, zodat de gewenste uitgang gevormd wordt. In de volgende secties zal het algoritme in stappen ontwikkeld worden. Het einddoel is om software te schrijven die zo goed mogelijk aan alle eisen voldoet.

Script-taal

Vanuit de studie wordt voor wiskundige problemen voornamelijk gewerkt met Matlab. Dit is een programma ontwikkeld om mathematische berekeningen in te programmeren en mee uit te voeren. Voor het implementeren van het algoritme zou Matlab een uistekende optie zijn. In Matlab kunnen eenvoudig matrixberekeningen uitgevoerd worden, en resultaten zijn zeer eenvoudig weer te geven in grafieken. Bovendien is het mogelijk om een Matlab-script te compileren tot een losstaand programma, zodat deze ook stand-alone aan te roepen en uit te voeren is.

Timing

In het algoritme worden oplaadblokken van 15 minuten gehandhaafd. Als een auto dus een bepaalde stuurstroom krijgt toebedeeld, zal deze na maximaal 15 minuten herzien worden. De stuurstroom zal eerder herzien worden in het geval dat het algoritme binnen die 15 minuten opnieuw aangeroepen wordt, bijvoorbeeld door een statuswijziging in de aansturing (zie sectie 6.3, "Communicatie bij statuswijziging auto").

Simulatie

Om een goed beeld te scheppen van de functionaliteit van het algoritme moeten bepaald worden hoe de resultaten weergegeven worden. Tevens is het belangrijk welke test-case gebruikt wordt. Het algoritme moet namelijk uiteindelijk veel verschillende situaties aankunnen. Het kan verschillen wanneer voertuigen aankomen, hoeveel voertuigen er zijn, hoeveel deze opgeladen moeten worden, hoelang de voertuigen hebben om opgeladen te worden, wat de verdeling is van de aankomsten en de vertrekken ten op zichte van elkaar. Er zijn dus veel variabelen, wat dus betekent dat een algoritme op veel situaties getest moet worden. Echter, zoals ook uit het oplaadgedrag bij Priva blijkt, is er een redelijk typische situatie: alle auto's komen rond half 9 aan en vertrekken weer rond 5 uur. Dit komt overeen met een gemiddelde werkdag, wat dus best logisch is. Soms zijn er werknemers die een halve dag werken, waarbij auto's dus een andere beschikbare oplaadtijd hebben. Deze situatie moet dus ook meegenomen worden in het testscenario.

Voor de simulaties wordt ook aangenomen dat alle auto's een maximale oplaadstroom van 16 ampère hebben. Dit is een soort worst-case scenario, gezien de andere opties (met 32 ampère of 3-fasig opladen) alleen maar sneller zijn.

Tabel 3.3: De auto's gebruikt voor simulatie (1: volledig elektrische auto, alleen met accu's. 2: state of charge)

Auto	FEV[1]	Aankomsttijd	Vertrektijd	Capaciteit $[kWh]$	SOC[2] $[\%]$	Benodigd $[kWh]$
1	nee	07:00	17:00	15	21	11,9
2	ja	08:00	17:00	24	40	19,2
3	nee	08:10	18:00	15	30	10,5
4	nee	08:50	17:10	18	20	14,4
5	nee	13:30	16:30	13	30	9,1
6	nee	08:40	17:20	16	22	12,5
7	ja	08:50	13:00	16	20	12,8
8	nee	08:50	16:00	15	20	12,0

Het totaal gevraagd vermogen is dus $94,33\ kWh$. Voor het beschikbare vermogen zullen wel twee scenario's onderscheiden worden, maar voor de vereenvoudiging met een constant maximaal vermogen gedurende de dag. In scenario 1 is er $35A$ beschikbaar voor het opladen, gelijk aan $8,05kW$, wat niet genoeg is om alle auto's op te laden. In het tweede scenario $90A$, gelijk aan $20,7kW$, wat wel genoeg om alle auto's op te laden. Deze twee situaties vragen twee verschillende oplossingen, namelijk respectievelijk peak shaving en load balancing. Het analyseren van die twee specifieke situaties is dus van belang voor een goede beoordeling van het algoritme.

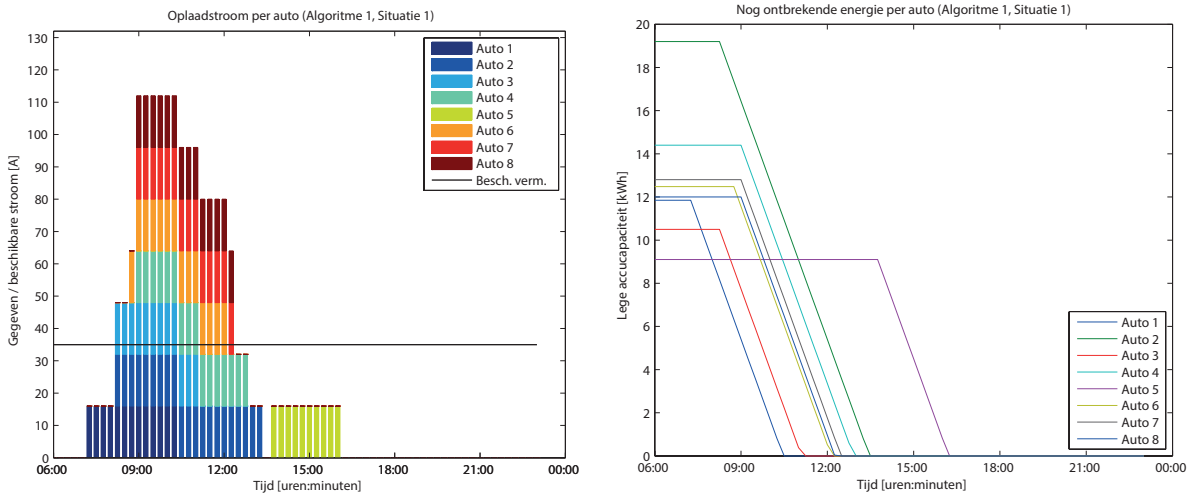
3.7. Algoritme 1: aansturen

De allereerste stap in het ontwikkelen van het algoritme is ook het meest eenvoudig. Slechts de fysieke beperkingen van het systeem worden meegenomen in dit algoritme, wat dus neerkomt op de 16 ampère per auto. De werking van het algoritme is dus als volgt samen te vatten

$$I_{stuur} = I_{stuur}^{max} \tag{3.2}$$

Hierin is I_{stuur} de stuurstroom voor een auto, de stroom waarmee de auto mag gaan opladen. I_{stuur}^{max} is de eerder gedefinieerde maximale stroom waarmee een auto mag opladen.

Testresultaten scenario 1



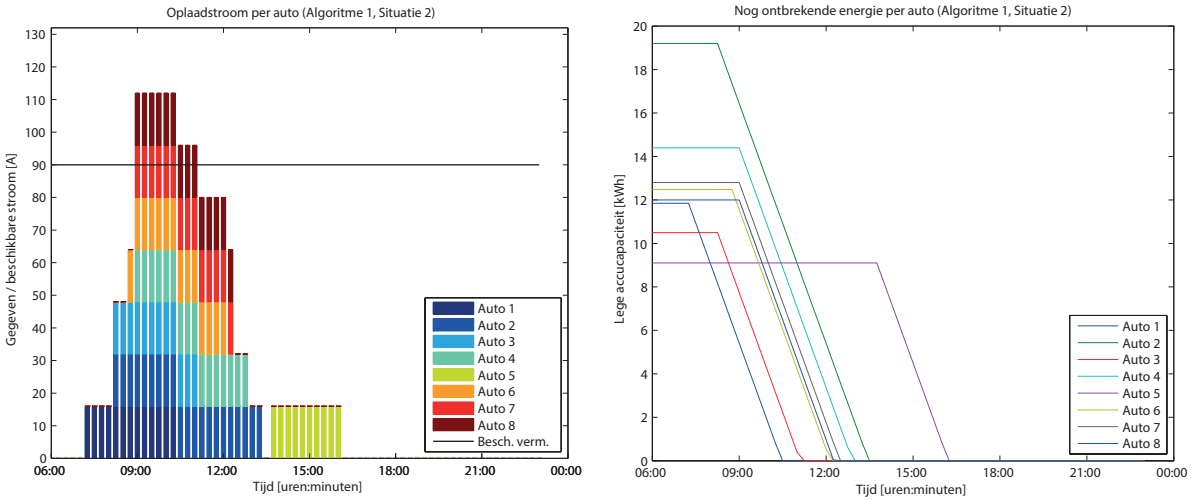
(a) Oplaadstroom per auto

(b) Diepte van de ontlading per auto

Figuur 3.2: Resultaten van algoritme 1, scenario 1

In figuur 3.2a is te zien dat het algoritme zich niets aantrekt van het maximale beschikbare vermogen. Alle auto's laden vanaf het moment dat ze aangesloten worden maximaal op, wat dus rond 09:00 uur een piek veroorzaakt van 112 ampère. Dit resultaat is gelijk aan hoe traditionele oplaadpalen werken. Hierdoor zijn voertuigen bovendien al halverwege de dag klaar met laden. Dit is, met oog op de eisen, onwenselijk gedrag. Het algoritme dient dus nog uitgebreid te worden.

Testresultaten scenario 2



(a) Oplaadstroom per auto

(b) Diepte van de ontlading per auto

Figuur 3.3: Resultaten van algoritme 1, scenario 2

Omdat dit algoritme geen rekening houdt met de maximaal beschikbare stroom, is er geen verschil tussen de resultaten van scenario 1 en scenario 2.

3.8. Algoritme 2: peak shaving

Peak shaving is één van de twee hoofdzaken van het algoritme. Peak shaving is van belang als er minder vermogen beschikbaar is dan waarmee de auto's maximaal kunnen opladen. In figuur 3.2a

en 3.3a is bijvoorbeeld te zien dat het in beide scenario's het geval is. In het eerste scenario is de stroomvraag zelfs 77 ampère te hoog. De totale stroom moet dus flink beperkt worden, dus alle auto's zullen minder hard mogen opladen.

Script

De essentie van het script is de volgende formule:

$$I_{stuur} = min(ratio_{auto} \cdot I^{max}_{totaal}, I^{max}_{stuur}) \tag{3.3}$$

Hierin is $ratio_{auto}$ een eenheidsloze waarde tussen 0 en 1, die aangeeft hoeveel aandeel van de totale beschikbare stroom I^{max}_{totaal} de auto zou krijgen.

Formule 3.8 is een uitbereiding op de eenvoudigere formule voor het vorige algoritme (formule 3.2. De stroom voor een auto wordt dus bepaald door het deel dat hij toebedeeld krijgt van het totale vermogen, maar dit wordt begrenst op de maximale stroom die de auto kan krijgen. Deze formule gaat op als geldt

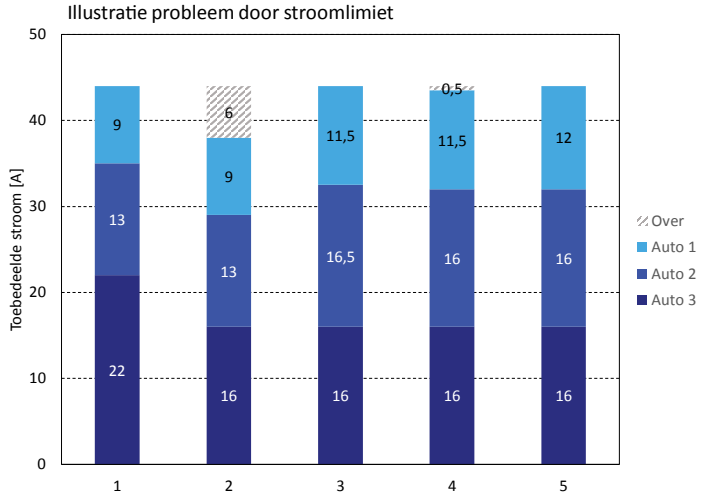
$$\sum_{i=1}^{\#\ auto's} ratio_i = 1 \tag{3.4}$$

zodat de totale maximale stroom niet overschreden wordt. Hoe de ratio, met andere woorden de prioriteitwaarde van een auto, verdeeld wordt, wordt verderop in dit hoofdstuk nader toegelicht onder het kopje "Prioriteren". Voor wordt aangehouden dat het beschikbare vermogen eerlijk verdeeld wordt, dus

$$ratio_{auto} = \frac{1}{\#\ auto's} \tag{3.5}$$

Maximale stroom

Deze formule brengt wel het volgende probleem met zich mee: als $ratio_{auto} \cdot I^{max}_{total} > I^{max}_{stuur}$, gaat er als het ware stroom verloren, omdat 3.4 geldt. Het probleem is geïllustreerd in figuur 3.4. Kolom 1 geeft weer hoe de beschikbare stroom in een willekeurige situatie op een willekeurig moment verdeeld zou kunnen worden. Auto 3 heeft echter meer stroom dan mogelijk is, dus die stroom moet beperkt worden (kolom 2). Zodoende is er 6 ampère "over".



Figuur 3.4: Illustratie van de uitdaging door het maximale stroom-limiet

Daarom moet het totale algoritme uitgerust worden met een subalgoritme dat die 6 ampère herkent en weer gaat verdelen. Het subalgoritme inventariseert dus eerst hoeveel stroom er nog verdeeld kan worden. Tevens wordt er gekeken welke auto's nog in aanmerking komen voor die extra stroom (auto's die dus nog niet aan hun maximale stroom zitten). Deze auto's krijgen naar verhouding van de eerder

bepaalde ratio extra stroom. In de situatie van figuur 3.4 krijgen auto 1 en 2 dus nog extra stroom aan de hand van de formule

$$I_{stuur}^{nieuw} = I_{stuur}^{oud} + \frac{ratio_{auto}}{\sum_{i=1}^{\#\ auto's\ die\ extra\ krijgen} ratio_i} \cdot I_{extra} \tag{3.6}$$

Waarbij I_{stuur}^{nieuw} de nieuwe stuurstroom is voor het voertuig, I_{stuur}^{oud} de stuurstroom die in eerste instantie berekend is, $ratio_{auto}$ de prioriteringswaarde van het voertuig en I_{extra} de totale stroom die nog verdeelt kan worden.

In kolom 3 van figuur 3.4 is dus te zien dat auto 1 en 2 naar ratio extra stroom krijgen. Auto 2 komt daarmee echter weer boven de maximaal toegestane stroom van 16 ampère. Het subalgoritme zal dus meerdere keren uitgevoerd moeten worden, totdat er geen stroom meer te verdelen is of alle auto's maximaal aan het opladen zijn.

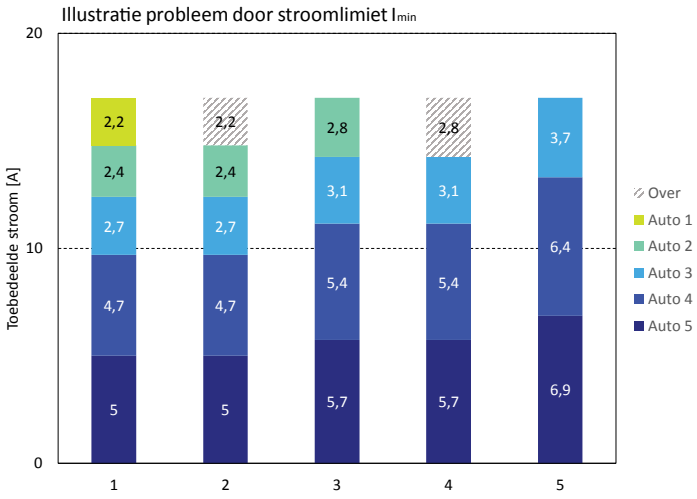
Minimale stroom

De hardware geeft naast een limitatie voor maximale stroom, ook een restrictie betreffende minimale stroom. Als een toebedeelde stroom onder $I_{stuur}^{min} = 3A$ zit, kan deze namelijk niet gecommuniceerd worden met de auto. Deze stroom moet dus verhoogd worden tot minimaal 3 ampère, of de desbetreffende auto kan op dat moment niet worden opgeladen.

Stroom verhogen Om de stroom te kunnen verhogen naar minimaal 3 ampère, moeten andere aangesloten auto's stroom inleveren. De reden dat deze auto's in eerste instantie al meer stroom kregen dan de auto onder 3 ampère, is omdat zij blijkbaar een hogere prioriteit hadden. Het zou dus oneerlijk zijn om een hogere prioriteit de dupe te laten zijn van tekortkomingen van het systeem, als het ook anders kan.

Auto niet laden Door een voertuig dat minder dan 3 ampère krijgt (hierna te noemen auto A) af te schakelen voor dat tijdslot, kunnen andere auto's, die inherent een hogere prioriteit hebben, weer sneller opladen. Aan het einde van het tijdslot, als de prioriteiten opnieuw berekend worden, zullen de andere auto's relatief meer opgeladen zijn dan auto A. Daardoor zal auto A weer meer stroom krijgen in het nieuwe tijdslot en waarschijnlijk niet meer afgeschakeld worden.

Dit is dus een zeer natuurlijke manier om het probleem op te lossen. Uiteindelijk is het probleem dan ook opgelost zoals in figuur 3.5 schematisch is weergeven.



Figuur 3.5: Illustratie van de uitdaging door het minimale stroom-limiet

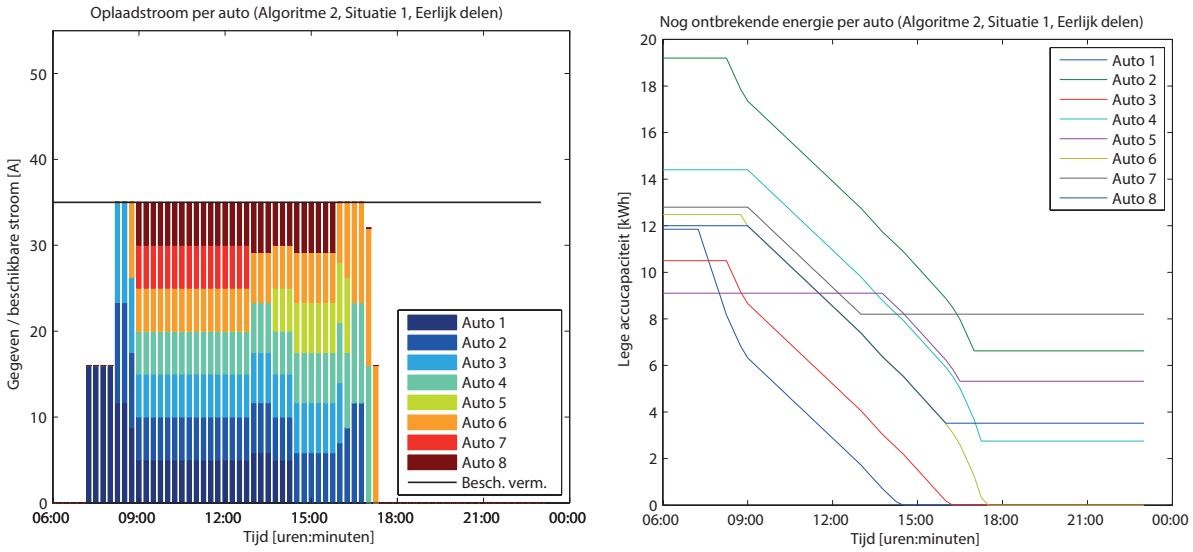
Kolom 1 van figuur 3.5 geeft weer de initiële verdeling van de beschikbare stroom weer, net zoals in figuur 3.4. Er zijn dus drie auto's die onder de minimale stroomgrens zitten. Door de auto met

de minste toebedeelde stroom geen stroom te geven (kolom 2), kunnen de andere auto's weer extra stroom krijgens volgens formule 3.6. In kolom 3 is te zien dat auto 3 door die extra stroom boven de 3 ampère is gekomen, maar auto 2 niet. Auto 2 zal dus ook niet opgeladen kunnen worden, waarna de extra vrijgekomen stroom (kolom 4) weer verdeeld kan worden over de andere auto's (kolom 5).

Wat hier te zien is, is dat de stroom beperkt wordt door tussen auto's te schakelen in de tijd. Als in een tijdslot uiteindelijk opgeladen wordt zoals in kolom 5 van figuur 3.5 weergegeven is, zal aan het eind van de tijdslot auto 3 meer opgeladen zijn dan auto's 1 en 2 (de afgeschakelde auto's). Bij het volgende tijdslot zal auto 3 dus relatief minder stroom krijgen, en waarschijnlijk ook minder dan 1 en 2. Auto 3 zal dan dus afgeschakeld worden, waardoor auto 1 of 2 opgeladen kan worden. Op die manier wisselt het opladen van de auto's met een lage prioriteit zich in de tijd af, en worden ze toch allemaal opgeladen. Het is dus wel van belang om de prioriteit te bepalen aan de hand van de resterende benodigde energie.

Simulatie

In figuur 3.7 is het resultaat te zien van het peak shaving algoritme in het geval van scenario 2. Als de grafiek in figuur 3.7a vergelijkt wordt met die in figuur 3.3a, is te zien dat de stroom in de periode tussen 09:00 uur en 12:00 uur begrenst is op het maximaal beschikbare vermogen. Hierdoor zijn de op dat moment aangesloten auto's ook later klaar met opladen. Desondanks zijn de auto's allemaal vol als ze weg moeten, zoals in figuur 3.7b te zien is.



(a) Oplaadstroom per auto

(b) Diepte van de ontlading per auto

Figuur 3.6: Resultaten van algoritme 2, scenario 1 bij gelijke prioriteit

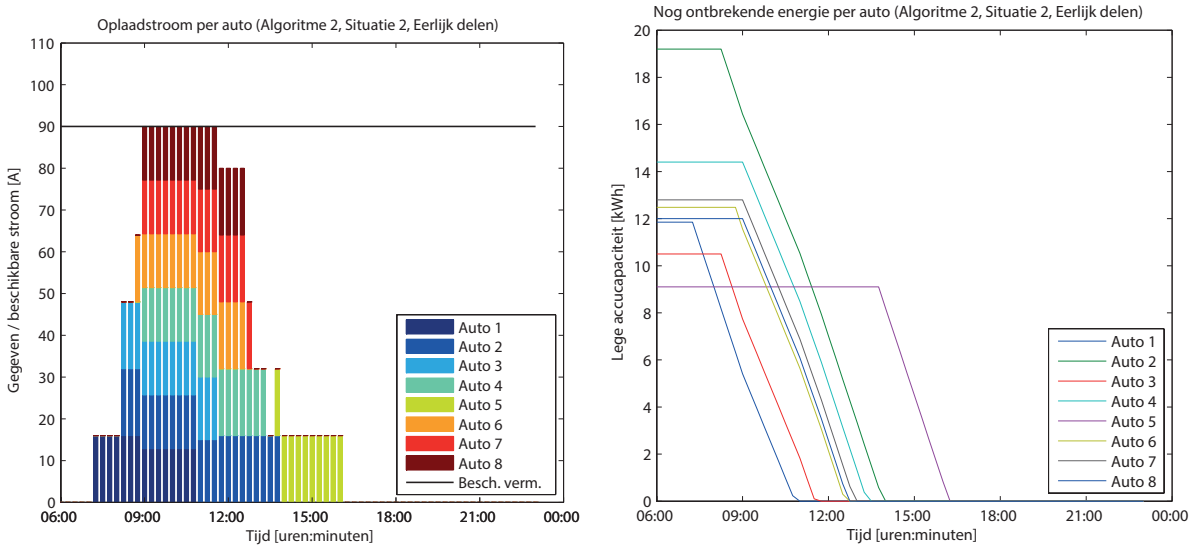
In tegenstelling tot scenario 2 is er in scenario 1 niet genoeg vermogen beschikbaar om alle voertuigen op te laden. in figuur 3.6 is dan ook te zien dat de stroom ook begrenst wordt op de maximale stroom, maar de voertuigen zijn aan het einde van de dag niet vol. Dit is ook af te lezen uit figuur 3.6. De exacte waardes van de overgebleven ontbrekende energie is tevens weergegeven in tabel 3.4 in de kolom 'gelijke prioriteit'.

Prioriteren

Het is dus van belang dat de auto's zo eerlijk mogelijk niet volledig opgeladen zijn. Factoren die gebruikt kunnen worden om die eerlijkheid te bewerkstelligen, zijn de volgende.

1. **Gelijke verdeling**, zoals in bovenstaande resultaten ook gebruikt is.

2. Als een voertuig een **puur elektrisch** voertuig is, zal het van groter belang zijn dat deze volledig, of zo vol mogelijk, opgeladen is.



(a) Oplaadstroom per auto

(b) Diepte van de ontlading per auto

Figuur 3.7: Resultaten van algoritme 2, scenario 2 bij gelijke prioriteit

3. De **resterende tijd** die een auto nog heeft om op te kunnen laden. Als een auto minder tijd heeft, zou deze sneller opgeladen moeten worden.

4. De resterende **benodigde energie** van een auto. Als een auto meer energie nodig heeft, zou deze sneller opgeladen moeten worden.

5. De **diepte van de ontlading** van een auto. Ongeveer hetzelfde als de resterende benodigde energie van een auto, maar dan relatief aan de totale capaciteit van de auto.

Door deze prioriteiten te gebruiken om het schema voor scenario 1 op te lossen, komen de volgende resultaten tot stand. De nummering in de tabel komt overeen met de nummering in bovenstaande opsomming.

Tabel 3.4: Overgebleven benodigde energie $[kWh]$ per prioriteringswaarde. Het nummer van de waarde komt overeen met het nummer van bovenstaande opsomming.

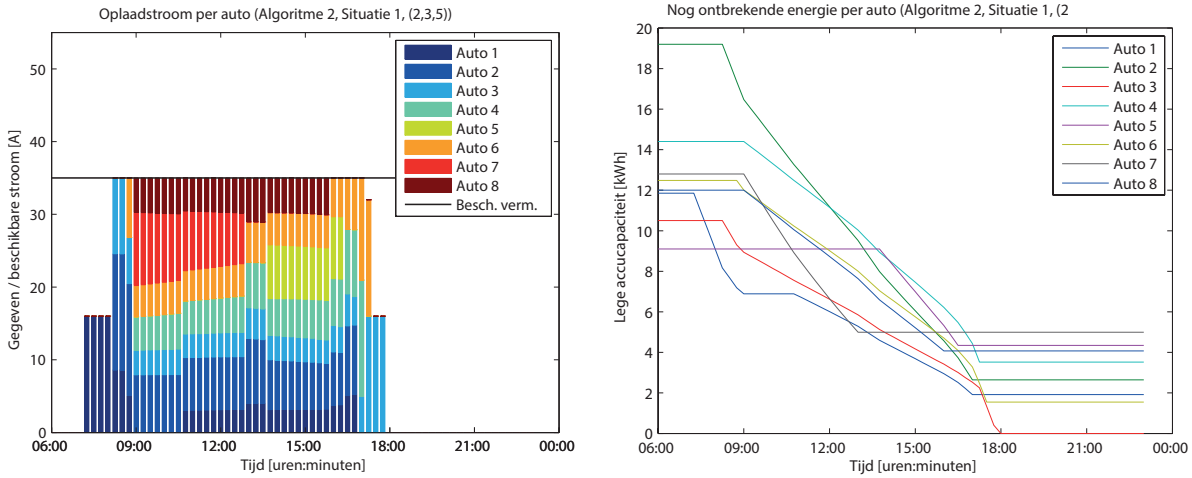
Auto	Benodigd	Oplaadtijd	Waarde 1	Waarde 2	Waarde 3	Waard 4	Waarde 5
1	11,9 kWh	10:00 u	-	-	-	1,59	1,17
2	19,2 kWh	09:00 u	6,63	0,19	6,69	3,31	5,57
3	10,5 kWh	09:50 u	-	-	-	-	-
4	14,4 kWh	08:20 u	2,75	5,55	3,12	2,05	2,32
5	9,1 kWh	03:00 u	5,32	6,17	5,15	4,60	3,81
6	12,5 kWh	08:40 u	-	2,31	0,05	0,78	0,58
7	12,8 kWh	04:10 u	8,20	5,64	7,70	7,85	7,57
8	12,0 kWh	07:10 u	3,52	5,30	3,11	3,83	3,31

De resultaten zijn zoals verwacht. Als een auto een hogere prioriteit krijgt, is hij minder leeg aan het einde van de dag. Auto 2 en 7 bijvoorbeeld, de volledig elektrische auto's, zijn bij gebruik van prioriteringswaarde 2 een stuk minder leeg dan in de andere. Het is ook opvallend dat het resterend benodigd vermogen bij waardes 4 en 5, en vooral bij 5, gelijker verdeeld is dan in de basisituatie (prioriteringswaarde 1).

Omdat in elke proef formule 3.4 gold, zal ook gelden

$$\sum_{j=1}^{\#\ ratio's} \sum_{i=1}^{\#\ auto's} ratio_{i,j} = 1; \quad j \subset gekozen\ prioriteitwaardes \tag{3.7}$$

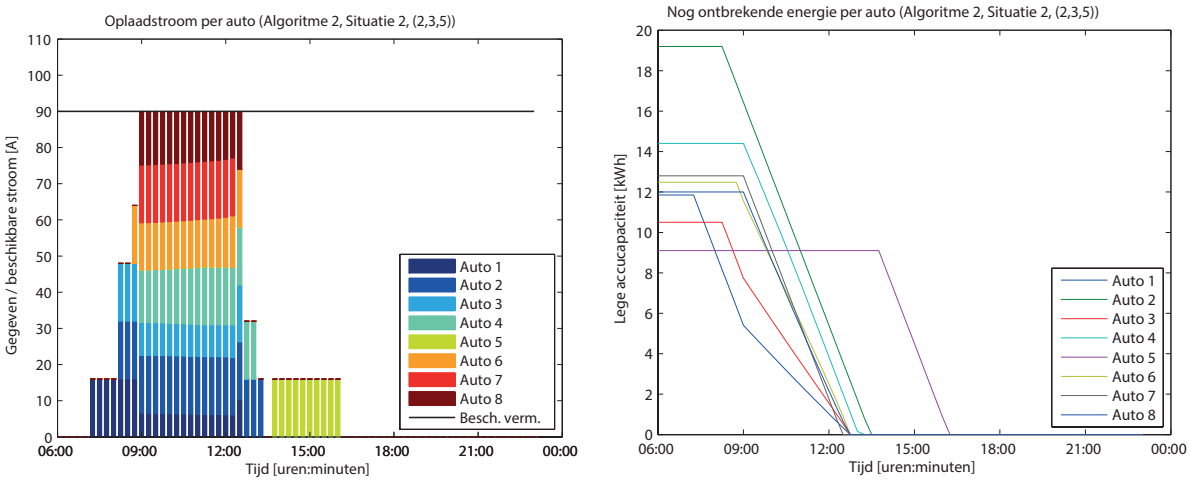
Prioriteringswaarde zijn dus te combineren, om zo tot de meest eerlijke prioriteitwaarde te komen. De meest volledige prioriteringswaarde is een combinatie van waardes 2, 3 en 5. Volledig elektrische voertuigen moeten meer prioriteit krijgen, en voertuigen zouden harder op mogen laden als ze minder tijd hebben. Bovendien zouden voertuigen een zo veel mogelijk gelijke state of charge moeten krijgen. Zoals eerder onder het kopje "Minimale stroom" ook is uitlegd, moet de resterende benodigde energie meegenomen worden in de prioriteitswaarde. Op die manier worden alle belangrijke wegingen meegenomen. Gecombineerde prioriteitwaardes zouden eventueel onderling nog gewogen kunnen worden. Dat wordt aanbevolen voor later onderzoek.



(a) Oplaadstroom per auto

(b) Diepte van de ontlading per auto

Figuur 3.8: Resultaten van algoritme 2, scenario 1 bij gekozen prioriteringswaarde



(a) Oplaadstroom per auto

(b) Diepte van de ontlading per auto

Figuur 3.9: Resultaten van algoritme 2, scenario 2 bij gekozen prioriteringswaarde

In figuur 3.8 zijn de resultaten van het simuleren van het tweede algoritme met de gekozen prioriteringswaarde voor scenario 1 weergegeven. Door dit te vergelijken met 3.6 is de invloed van de

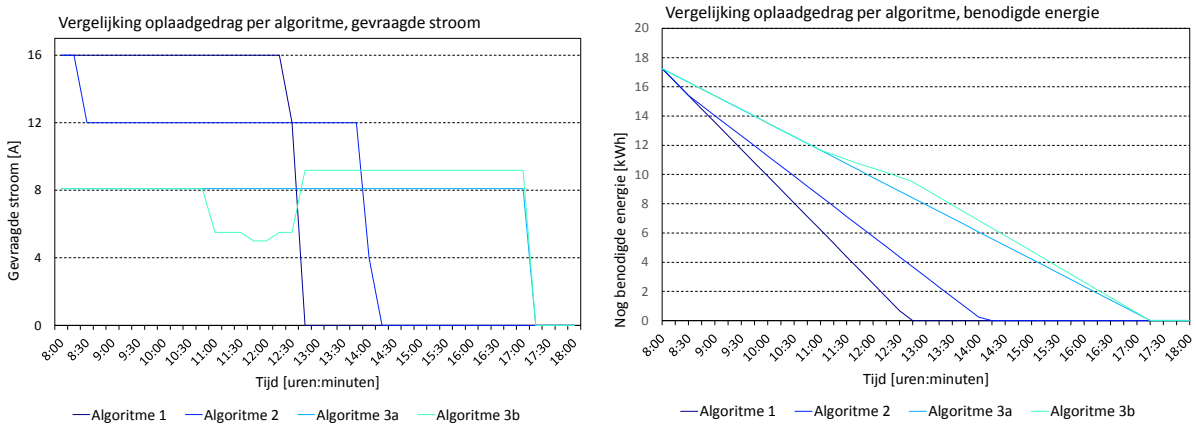
prioriteringswaarde te zien. De stromen zijn nu meer dynamisch verdeeld, en de uiteindelijke nog ontbrekende energie per auto is eerlijker verdeeld.

Tevens is in figuur 3.9 het resultaat van het uiteindelijke peak shaving-algoritme voor scenario 2 weergegeven. Als dit vergeleken wordt met de resultaten in figuur 3.7, is te zien dat de prioriteringswaarde niets uitmaakt voor het eindresultaat, als er genoeg vermogen is.

3.9. Algoritme 3: load balancing

De resultaten van het vorige algoritme bieden een uitstekende verbetering in het geval van scenario 1 (figuur 3.8). In scenario 2 (figuur 3.9) echter, is een zeer groot verschil tussen minimale en maximale stroomgebruik over de dag heen te zien. Qua load balancing is er dus nog te winnen. Een zo gelijk mogelijk verdeelde stroom betekent een zo gebalanceerd mogelijke belasting, en tevens een lagere energierekening voor de afnemer.

Om de situatie nog verder te verduidelijken is in figuur 3.10 een hypothetische situatie geschetst. Een auto heeft ongeveer $17\ kWh$ nodig om volledig opgeladen te zijn. Algoritme 1 geeft de auto vanaf het begin de maximale toegestane stroom, waardoor de auto rond 13.00 uur al klaar is met opladen. Algoritme 2 limiteert de stroom al enigszins door rekening te houden met een maximale totale stroom voor alle voertuigen, waardoor een auto al meer verspreid in de tijd opgeladen wordt. Echter gaat dit dus niet voor situaties op waar de maximale totale stroom geen limiet voor een individuele auto verzorgt.



(a) Oplaadstroom per algoritme

(b) Nog ontbrekende energie per algoritme

Figuur 3.10: Oplaadgedrag van de algoritmes

Daarom is er nog een uitbereiding nodig van het algoritme, zoals in figuur 3.10 geillustreerd als Algoritme 3a. Het principe is dat de stroom zo gekozen wordt, dat de auto precies klaar is met opladen zodra deze weg moet. In 3.10 is dan ook nog Algoritme 3b geïllustreerd, waarbij de auto tussen 11.00 uur en 12.30 uur minder hard mag opladen, bijvoorbeeld doordat de maximaal beschikbare stroom daar het limiet bepaalt. De bedoeling is dat de auto daarna zodanig veel stroom krijgt, dat deze weer precies op tijd opgeladen is.

Script

De werking van het beschreven algoritme kan weer samengevat worden in één formule. Deze formule is een uitbereiding op formule 3.3:

$$I_{stuur} = min(ratio_{auto} \cdot I_{totaal}^{max}, I_{stuur}^{max}, I_{stuur}^{dynamisch}) \tag{3.8}$$

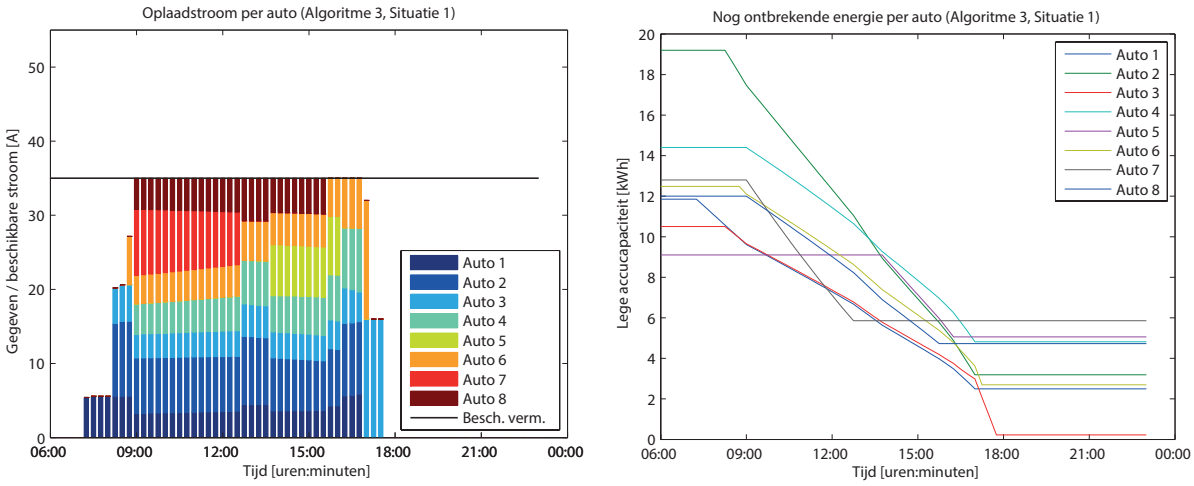
$I_{stuur}^{dynamisch}$ is dan de stroom die nodig is om de auto precies op tijd volledig opgeladen te krijgen. Deze waarde wordt iedere tijdstap in het algoritme opnieuw bepaald volgens de volgende formule

$$I_{stuur}^{dynamisch} = \frac{P_{auto}^{resterend}}{230\ V \cdot (t_{auto}^{vertrek} - t_{nu})} \tag{3.9}$$

waarin $P_{auto}^{resterend}$ het resterende benodigde vermogen van de auto is in J, wat gedeeld moet worden door de spanning waarmee opgeladen wordt $(230\ V)$ om de benodigde stroom te krijgen.

Simulatie

Na uitbereiding van het algoritme was het mogelijk om de eventuele verbeteringen van de aanpassing in beeld te brengen. De resultaten ervan zijn te zien in figuur 3.11 en 3.12.



(a) Oplaadstroom per auto

(b) Diepte van de ontlading per auto

Figuur 3.11: Resultaten van algoritme 3, scenario 1

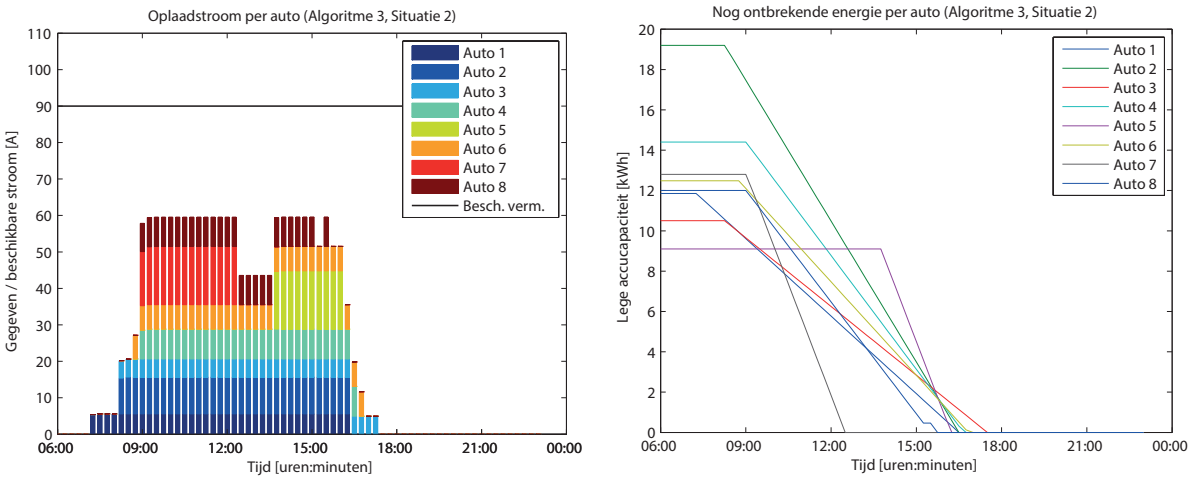
Om het verschil van algoritme 3 ten opzichte van algoritme 2 in de situatie van scenario 1 beter weer te geven, is in tabel 3.5 nog in cijfers neergezet hoeveel energie er nog in de auto's kan aan het einde van de dag. Alle auto's blijken namelijk minder opgeladen te zijn dan in het geval van algoritme 2. Dit komt omdat algoritme 3 in het begin van de dag de eerste auto's minder stroom geeft. Het opladen stelt zich in het begin namelijk in op een stroom waarmee de auto aan het eind van de dag volledig opgeladen zou zijn. Als deze stroom echter voor de rest van de dag beperkt wordt, zal de auto niet zijn volledige oplaadmogelijkheid benut hebben. Algoritme 2 is dus beter in het geval dat er niet genoeg vermogen is om alle auto's op te laden.

Tabel 3.5: Vergelijking eindresultaten algoritme 2 en 3 voor scenario 1

	Algoritme 2		**Algoritme 3**	
	kWh_{rest}	SOC_{rest}	kWh_{rest}	SOC_{rest}
Auto 1	1,92	13%	2,49	17%
Auto 2	2,64	11%	3,19	13%
Auto 3	0	0%	0,22	1%
Auto 4	3,52	20%	4,82	27%
Auto 5	4,34	33%	5,06	39%
Auto 6	1,55	10%	2,70	17%
Auto 7	4,99	31%	5,86	37%
Auto 8	4,07	27%	4,73	32%
Totaal	**23,03**		**29,07**	

In het geval van scenario 2, zoals weergegeven in figuur 3.12, heeft algoritme 3 wél toegevoegde waarde. Als de resultaten vergeleken worden met de resultaten van algoritme 2 in scenario 2 (figuur

3.9), is te zien dat de auto's nog steeds allemaal zijn opgeladen, maar dat de belasting een stuk beter verspreid is dan met algoritme 2.



(a) Oplaadstroom per auto

(b) Diepte van de ontlading per auto

Figuur 3.12: Resultaten van algoritme 3, scenario 2

3.10. Conclusie en aanbeveling

Aan de hand van de simulatieresultaten is te concluderen dat algoritme 2 het beste is als er niet genoeg vermogen is om alle auto's op te laden. Algoritme 3 komt het beste uit de bus als er meer dan genoeg vermogen is. Dit kan dus geïmplementeerd worden door altijd eerst algoritme 3 te draaien, en als blijkt dat niet alle auto's, of een deel van de auto's, niet helemaal zullen opladen, wordt algoritme 2 gedraaid.

In het beste geval wordt er nog een 4e algoritme ontwikkeld, dat algoritme 3 uitgebreid met een anticausaal gedeelte. Dat algoritme zou dan in een geval als in figuur 3.11 terugkijken naar eventuele plekken waar auto's nog plek hebben om extra opgeladen te worden.

Een resultaat zoals in scenario 2 (figuur 3.12) zou tevens nog verbeterd kunnen worden door in de periodes 9.00 uur - 12.30 uur en 13.30 uur - 15.00 uur minder stroom te geven aan auto's die dan ervoor, ertussen of erna meer zouden kunnen opladen. Dit is echter een te complex algoritme om in de span van dit onderzoek te kunnen ontwikkelen.

Verwachtingen met betrekking tot ingebruikname

Voor de simulatie zijn heel wat vereenvoudigingen gedaan. De twee scenario's die gekozen zijn voor de simulatie zullen dan ook niet het werkelijke gebruik van het algoritme 100% beschrijven. De meeste vereenvoudigingen zijn echter zo gekozen dat ze een werkelijke situatie zo veel mogelijk indekken.

Als een auto namelijk eerder weggaat, of binnen een berekeningskwartier al klaar is met opladen, zal het algoritme automatisch een nieuw schema berekenen. In dat schema zal er meer vermogen beschikbaar blijken dan eerst, waardoor andere auto's meer zouden kunnen opladen.

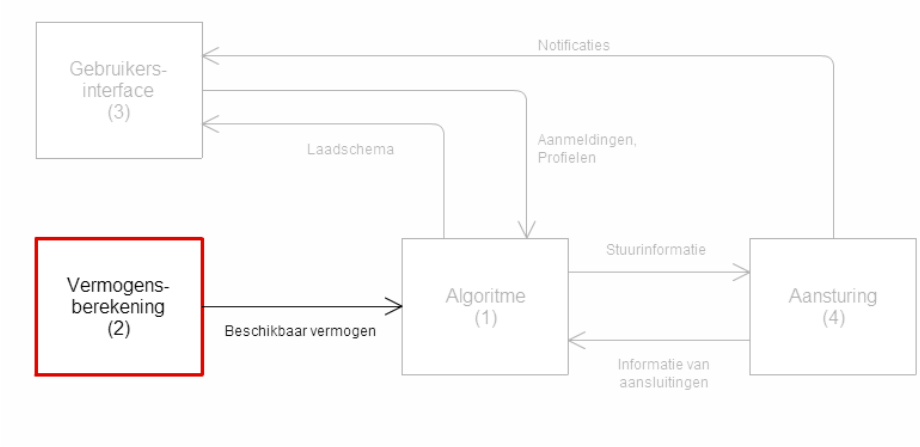
Oplaadgedrag van de auto's

Er is bij de simulaties overigens uitgegaan van dat de auto's daadwerkelijk met de toebedeelde hoeveelheid stroom zullen gaan opladen. Als bij de systeemintegratie blijkt dat de auto's minder opladen dan dat van ze gevraagd wordt, zal het algoritme uiteindelijk te weinig vermogen geven om volledig opgelaad te raken. De actuele state of charge van een auto wordt door de huidige implementatie wel bij iedere herberekening goed meegenomen, maar de uiteindelijke state of charge zal altijd lager uitvallen dan verwacht.

4

Vermogensberekening

Een belangrijk deel van dit project is het inspelen op het beschikbare vermogen. Een grote aanleiding naar dit project was hoge energiekosten van de elektrische laadpalen. Deze elektrische laadpalen functioneerden als 'domme' voedingsbronnen die een constante stroom de elektrische voertuigen instuurden. Dit systeem speelt op geen enkele manier in op het energieverbruik van andere energiemodules. Het probleem hierbij is dat dit kan zorgen voor ongewenste pieken van het energieverbruik. Het doel van dit hoofdstuk is het ontwikkelen van een module die een efficiënte en implementeerbare oplossing bedenkt om inzicht te krijgen in hoeveel vermogen beschikbaar is. In figuur 4.1 is aangegeven hoe deze module zich verhoudt ten opzichte van het gehele systeem.



Figuur 4.1: Uitgebreid overzicht van het product

4.1. Mogelijkheden

Voordat gewerkt wordt aan een oplossing zullen verschillende mogelijkheden onderzocht worden. Deze mogelijkheden zullen beoordeeld worden op enkele factoren. De belangrijkste zijn de complexiteit van de oplossing, de nauwkeurigheid van de voorspelling die deze mogelijkheid biedt en of het voldoet aan de eisen die gesteld zijn in het algoritme (zie sectie 3.4).

1) Constant niveau

Door deze methode wordt het energieverbruik van de laadpalen gelimiteerd door een maximaal niveau van beschikbaar vermogen vast te leggen voor de module. Dit is verreweg de makkelijkste mogelijkheid, maar houdt in geen enkel geval rekening met het energieverbruik van andere energiemodules.

2) Huidig beschikbaar vermogen
Een centraal systeem gaat na hoeveel energie al verbruikt wordt door iedere module in het gebouwbeheersysteem (GBS), en zal op basis hiervan beslissen hoeveel energie beschikbaar gesteld mag worden voor een module op basis van een prioriteringsysteem. Er wordt dan een bepaalde hoeveelheid vermogen beschikbaar gesteld en deze waarde wordt dan doorgegeven aan het algoritme die dit als tijdelijk limiet zal beschouwen voor de berekeningen.

3) Huidig en toekomstig beschikbaar vermogen
Deze mogelijkheid bouwt voort op mogelijkheid (2). Bij het bepalen van het toekomstige vermogen is een betrouwbare schatting nodig, anders is het niet van toegevoegde waarde. Het kan zelfs schadelijk zijn voor het laadschema als deze voorspelling er ver naast zit en hier niet goed mee omgegaan wordt. Een goede voorspelling is onderhevig aan verschillende parameters. De beschikbaarheid van historische data van het verbruik van vermogen is hier een kritische factor. Bij gebrek aan voldoende en betrouwbare data is de kans op een slechte schatting groot. Het is in ieder geval noodzakelijk voor het algoritme dat het toekomstige vermogen geschat wordt (zie hoofdstuk 3.4), anders kan er geen laadschema voor de rest van de dag opgesteld worden.

4) Artificial Neural Network
Zoals al eerder vermeld in het state-of-the-art onderzoek (1.3.2) is een Artificial Neural Network (ANN) een veelgebruikt algoritme in de computerwetenschappen en andere gerelateerde studies. Het is geïnspireerd door het centrale zenuwsysteem in de hersenen van dieren. Dit algoritme kan zowel leren uit voorgaande situaties als patronen herkennen.

Afweging
Mogelijkheid 1 houdt geen rekening met het energieverbruik van andere energiemodules en is daarom geen oplossing voor het probleem dat zich heeft gesteld. Mogelijkheid 2 speelt dynamisch in op het energieverbruik van andere energiemodules en is daarom een mogelijke optie. Mogelijkheid 3 is een goede oplossing die ook nog eens noodzakelijk blijkt te zijn voor het algoritme. De moeilijkheidsgraad ligt hier bij het betrouwbaar schatten van het toekomstige beschikbare vermogen. Uit eerder onderzoek is gebleken dat mogelijkheid 4 veelbelovende resultaten biedt [9] [3]. Echter is de complexiteit van deze mogelijkheid zeer groot. Dit is iets dat verwacht kan worden van een algoritme dat het gedrag van hersenen poogt na te bootsen.

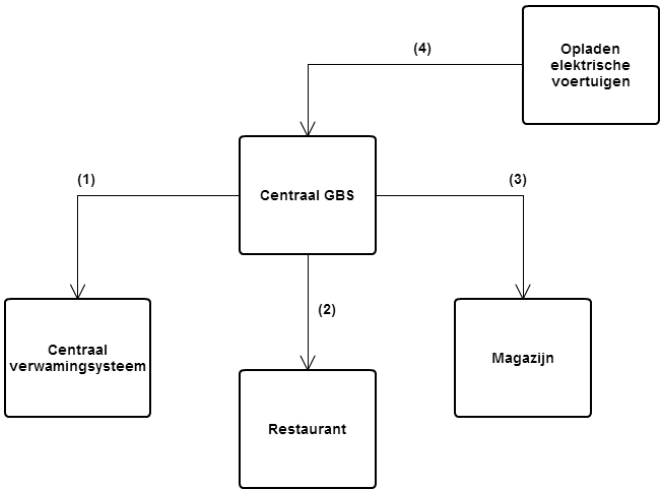
Uit deze analyse van de mogelijkheden is duidelijk dat mogelijkheid 1 niet plausibel is als oplossing voor het gestelde probleem. Mogelijkheid 4 daarentegen biedt een plausibele oplossing, echter is de complexiteit te groot om toe te passen binnen dit project. Het is wel aanbevolen voor een volgend onderzoek. Mogelijkheden 2 en 3 zijn allebei plausibele oplossingen voor de probleemstelling. Echter is het huidige beschikbare vermogen op zichzelf te weinig informatie over het vermogen om een laadschema te genereren. Mogelijkheid 2 voldoet dus niet aan de eisen van het algoritme. Er blijft daardoor slechts één mogelijkheid over die aan alle factoren van de beoordeling voldoet en dat is 3.

4.2. Implementatie
Er is gekozen voor een systeem dat het huidige beschikbare en het toekomstige beschikbare vermogen onderzoekt. deze twee onderdelen zullen afzonderlijk behandeld worden.

4.2.1. Huidige beschikbare vermogen
De vermogensberekening zal het huidige beschikbare vermogen moeten achterhalen van iedere energiemodule en op basis van een bepaalde prioritering een keuze maken hoeveel vermogen er aan iedere module gegeven wordt. Aangezien het project bij een externe werkgever wordt gedaan, is het interessant om na te gaan wat voor bestaande systemen gebruikt kunnen worden.

Gebouwbeheersysteem Priva
Het project wordt verricht bij Priva die een GBS heeft waar gebruik van gemaakt kan worden. Dit GBS heeft een centraal systeem dat alle energiemodules afgaat en vraagt hoeveel vermogen elke module nodig heeft volgens een systeem van polling zoals te zien is in figuur 4.2. Elke module heeft een

regelcomputer die geïntegreerd is in deze kring. Dat dit systeem beschikbaar is voor implementatie verkleint de complexiteit van deze oplossing drastisch. Aangezien de complexiteit van de oplossing een belangrijke beoordelingsfactor is, is dit systeem bij Priva een ideaal systeem voor de integratie van de vermogenberekeningen.
Een iteratieslag duurt ongeveer 10 seconden. Dit communicatieproces is continu aan de gang.



Figuur 4.2: Vereenvoudigd schema van het polling systeem dat beschikbaar is bij Priva. Het centrale systeem vraagt aan iedere energiemodule hoeveel vermogen deze nodig heeft. Daarna zal de module van de elektrische voertuigen vragen aan het centrale systeem hoeveel er beschikbaar is.

Communicatie met algoritme

Het algoritme bepaalt een laadschema die stand houdt voor een tijdsblok van 15 minuten, terwijl het pollingsysteem van het centrale GBS een continu proces is. Logischerwijs is de beschikbare hoeveelheid vermogen niet constant binnen een tijdsblok van 15 minuten. Er moet dus bedacht worden hoe er door de vermogensberekening omgegaan dient te worden met de polling die ongeveer iedere 10 seconden plaatsvindt. Twee mogelijkheden hiervoor worden behandeld.

Gemiddelde over tijd Er kan gekozen worden om de beschikbare hoeveelheid vermogen die na iedere polling van ongeveer 10 seconden ontvangen wordt, op te slaan in een lijst. En voordat het tijdsblok van 15 minuten afgelopen is, kan een geometrisch gemiddelde berekend worden die dan verstuurd wordt naar het algoritme. Dit gemiddelde zal berekend worden uit ongeveer 90 waardes $(15 \cdot \frac{60s}{10s})$ voor het beschikbare vermogen.

Meest recente waarde Dit vereist veel minder rekenkundig vermogen van de berekeningen en het is minder belastend voor het geheugen. Verder is het exacte beschikbare vermogen nodig op het moment dat het algoritme een laadschema gaat genereren en is de meest recente waarde het meest representatieve voor dat moment zelf.

Afweging Het verschil in bijdrage aan het zuiniger omgaan met het energieverbruik is beduidend minder dan het verschil in het belasten van het regelsysteem. Voor het opslaan van de meest recente waarde zal de prestatie van de berekening veel minder beïnvloed worden dan bij het berekenen van het gemiddelde over de tijd. Er zou gekozen kunnen worden om veel minder dan 90 waardes op te slaan en daarvan een gemiddelde te maken. Echter is het niet zo dat het gemiddelde van het beschikbare vermogen een betere representatie is op het moment van opvragen hiervan op het einde van een tijdsinterval. Er wordt daarom gekozen voor het gebruiken van de meest recente waarde van het beschikbare vermogen.

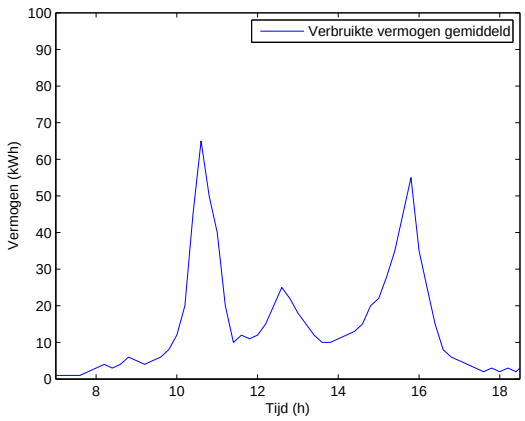
De meest recente waarde van het beschikbare vermogen wordt doorgestuurd via de regelcomputer naar het algoritme via XML. Dit wordt verder uitgelegd in sectie 6.2.

4.2.2. Toekomstige beschikbare vermogen

Op voorhand is er een plafond ingesteld waar het verbruik van het gehele gebouwsysteem niet overheen mag gaan, dit zal een belangrijke indicatie zijn voor hoeveel vermogen er beschikbaar is voor de module van de laadpalen. De berekeningen die nodig zullen zijn voor het schatten van het toekomstige beschikbare vermogen zullen gebruik maken van het huidig beschikbare vermogen, aangezien dit relateert met de werkelijke situatie. Het toekomstig beschikbare vermogen wordt allereerst geschat door het model van het verbruik op een gemiddelde dag af te trekken van het plafond (zie formule 4.1; voor het gemak wordt dit vanaf nu iedere keer afgekort naar schatting). Vervolgens wordt op het huidige beschikbare vermogen voortgebouwd. De wijze waarop is door de schatting te verschuiven langs de as van het vermogen tot deze snijdt met het huidige beschikbare vermogen. Vervolgens zal duidelijk zijn hoeveel vermogen er ongeveer in de toekomst beschikbaar zal zijn voor de module van de laadpalen. Dit wordt dan na ieder tijdblok opnieuw berekend met de nieuwe waarde voor het huidige beschikbare vermogen.

$$schatting = plafond - model\ gemiddelde\ dag \tag{4.1}$$

Het gebruikte model voor het verbruikte vermogen is zichtbaar in figuur 4.3, dit is geen realistisch model maar is slechts ter illustratie. De methode die wordt toegepast is schematisch weergegeven in figuur 4.4.
Er is niet nagegaan of na de verschuiving ook schaling van dit model moet plaatsvinden. Dit wordt aanbevolen voor verder onderzoek.

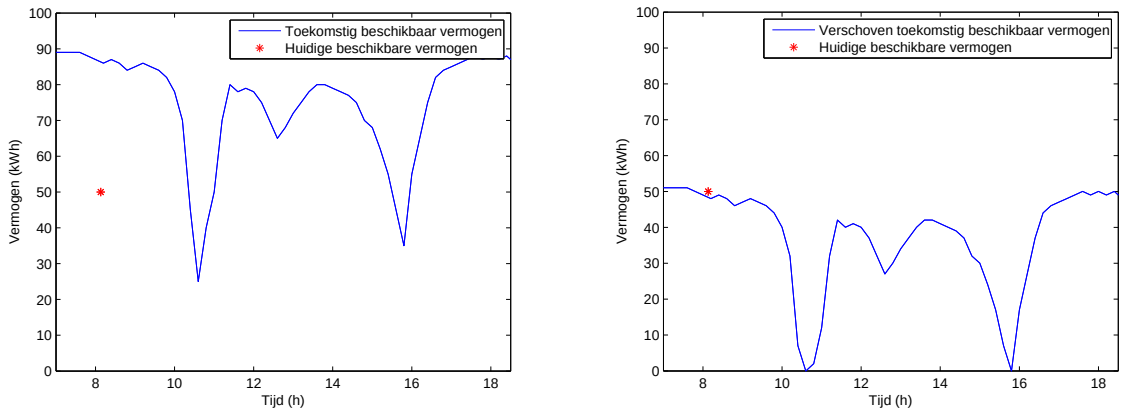


Figuur 4.3: Model van het verbruikte vermogen op een gemiddelde dag

Algoritme toekomstig beschikbaar vermogen

Het algoritme bevat de dataset van het model van het vermogen dat verbruikt wordt op een gemiddelde dag en het plafond wat op voorhand afgesproken was. De dataset wordt van het plafond afgetrokken en dit is de eerste schatting van het toekomstig beschikbare vermogen (afgekort naar schatting). Deze verkregen schatting heeft op zijn x-as de tijd. Wanneer het huidige beschikbare vermogen ontvangen wordt door dit algoritme met het bijbehorende tijdstip, zal de waarde uit de schatting gezocht worden die overeenkomt met dit tijdstip. Vervolgens wordt het verschil berekend tussen het huidige beschikbare vermogen en de gevonden waarde uit de schatting. Dit verschil wordt toegevoegd aan iedere waarde van de schatting. Zo wordt de schatting verschoven tot deze snijdt met het huidig beschikbare vermogen. Dit is geïllustreerd met figuur 4.4.
Waar dit algoritme vanuit gaat is dat het huidige tijdstip altijd gevonden wordt in de tijdreeks. Dit is niet mogelijk aangezien de dataset discrete waardes bevat met bijbehorende discrete tijdsintervallen. Hier moet een oplossing voor bedacht worden.

Eerst gaat in een if-statement gekeken worden of de huidige tijd overeenkomt met een discrete tijdwaarde, als dit niet overeenkomt, wordt in de elseif gecontroleerd of deze mogelijk tussen twee tijdsintervallen in ligt. Als dit het geval is, kan met formule 4.2 een schatting gemaakt worden van



(a) Het huidige beschikbare vermogen en een schatting van het toekomstig beschikbare vermogen

(b) De schatting van het toekomstig beschikbare vermogen is verschoven tot deze snijdt met het huidige beschikbare vermogen

Figuur 4.4: Illustratie van de methode om het toekomstig beschikbare vermogen te schatten

de overeenkomende waarde uit de dataset van de eerste schatting van het toekomstig beschikbare vermogen.

$$waarde = \frac{tijd\ nu - tijd(index)}{tijdsinterval} \cdot (schatting(index+1) - schatting(index)) + schatting(index) \tag{4.2}$$

Hierbij bevindt de huidige tijd zich dus tussen twee discrete tijdwaardes in. De index van de laagste discrete tijdwaarde wordt genomen. De huidige tijd is dus altijd groter dan deze discrete tijdwaarde. De schatting wordt dan gemaakt door te kijken hoe ver deze huidige waarde gevorderd is in dit tijdsinterval. Aan de hand van deze vordering wordt een schatting gemaakt van de overeenkomende waarde uit de dataset van het model van het verbruikte vermogen (formule 4.2).

4.3. Simulatie

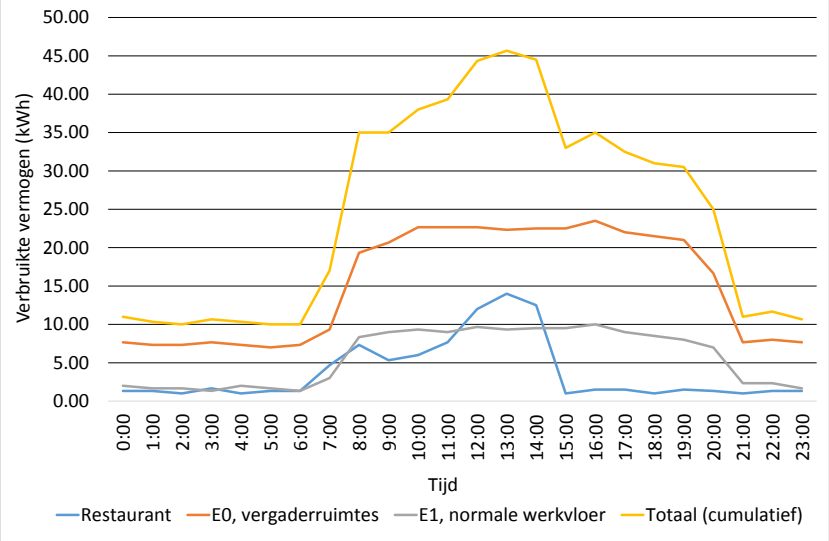
Het project wordt uitgevoerd bij een externe werkgever, Priva. Het is daarom een logische stap om het model van het verbruikte vermogen van een gemiddelde dag op te vragen bij deze werkgever. Op basis van deze dataset(s) kan de methode die beschreven is in 4.2.2 toegepast worden in Matlab en kan het gesimuleerd worden.

Profiel verbruikte vermogen

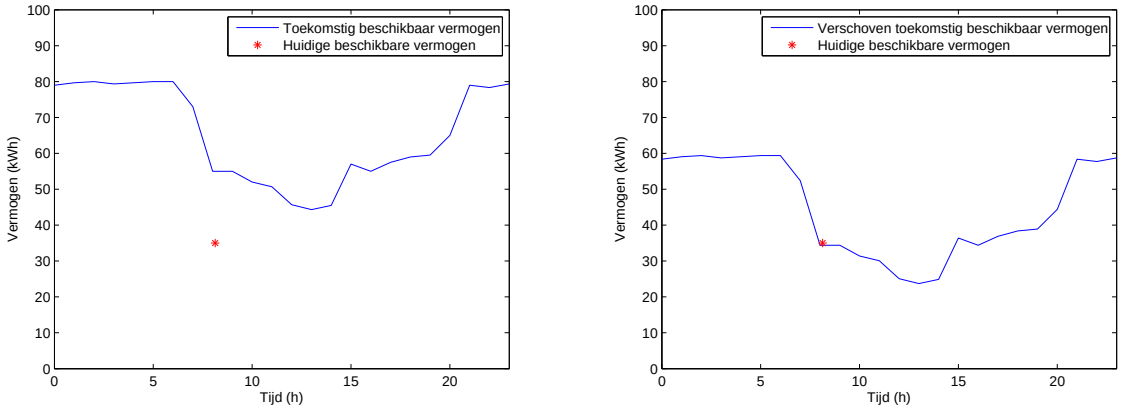
Elk afdeling bij Priva heeft zijn eigen data over het verbruik van vermogen per dag. Van verschillende afdelingen is per dag een gemiddelde gemaakt. Dit is in excel verwerkt en het resultaat is te zien in figuur 4.5.

Algoritme toekomstig beschikbaar vermogen

De totale waarde van enkele afdelingen is geïmplementeerd in het algoritme voor het simuleren van het toekomstig beschikbare vermogen. Het resultaat van deze simulatie is weergegeven in figuur 4.6. Uiteraard moet nog om een representatief beeld van de hele situatie te krijgen het verbruik van het vermogen van iedere afdeling meegenomen worden.



Figuur 4.5: Gemiddelde verbruikte vermogen van enkele afdelingen van Priva per dag



(a) Het huidige beschikbare vermogen en een schatting van het toekomstig beschikbare vermogen. Voor het model van een gemiddelde dag zijn gegevens van Priva gebruikt.

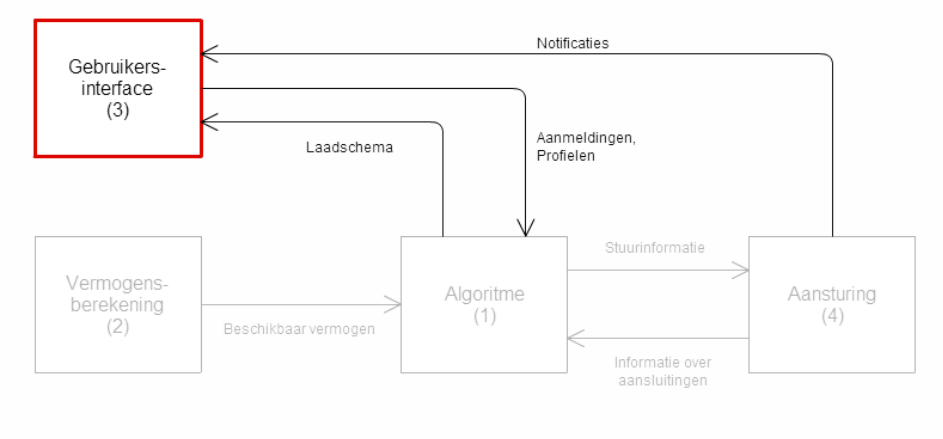
(b) De schatting van het toekomstig beschikbare vermogen is verschoven tot deze snijdt met het huidige beschikbare vermogen. Voor het model van een gemiddelde dag zijn gegevens van Priva gebruikt.

Figuur 4.6: Simulatie van de methode om het toekomstig beschikbaar vermogen te bepalen met gegevens van Priva

5

De gebruikersinterface

De gebruikersinterface vormt de brug tussen het algoritme en de gebruiker. Via deze zal informatie vergaard worden die nodig zal zijn bij het maken van een oplaadschema. Het is dus belangrijk dat deze interface robuust is en gemakkelijk te gebruiken. De gebruiker moet hier dagelijks mee willen omgaan. Deze interface hoort toe te voegen aan de ervaring van het elektrisch opladen, in plaats van een extra drempel te vormen.



Figuur 5.1: Plaats van de gebruikersinterface in het product

Om de keuze te maken hoe de gebruikersinterface vorm te geven, zijn er enkele zaken waarmee rekening gehouden moet worden. Deze staan opgesomd in de onderstaande lijst.

- Welke informatie is er nodig van de gebruikers?
- Hoe wordt informatie verkregen?
- Hoe kan het zo gebruiksvriendelijk mogelijk ingericht worden?
- Wat laten zien aan de gebruikers?
- Hoe en wanneer wordt een notificatie verstuurd naar de gebruiker?

De gebruiker is een heel belangrijke beoordelingsfactor voor de interface en om die reden zijn er drie elektrische rijders die werkzaam zijn bij Priva geïnterviewd. Deze interviews zullen in onderdeel 5.2 behandeld worden. Eerst zal nog behandeld worden welke informatie van de gebruikers opgehaald moet worden.

5.1. Informatie nodig van gebruikers

De informatie die nodig is van de gebruikers kan gecategoriseerd worden in twee onderdelen. De eerste categorie is de informatie die essentieel is voor het algoritme om een betrouwbaar laadschema te genereren. Dit is in het onderdeel 3.4 behandeld en zal daarom kort samengevat worden. De tweede categorie is de informatie die nodig is om zowel de gebruiker als de laadpaal waar de gebruiker op aansluit te herkennen. De herkenning is noodzakelijk voor feedback.

Tabel 5.1: Categoriën voor informatie die nodig is van de gebruiker

Essentieel voor algoritme	**Herkenning**
State of charge	Aansluiting voertuig
Aankomst- en vertrektijd	Identificatie gebruiker
Nog benodigde hoeveelheid energie	

State of charge

Uit onderzoek van de hardware is het gebleken dat het niet haalbaar is om de state of charge direct van de auto uit te lezen [1]. Het is daarom nodig om deze op te vragen van de gebruiker via een bepaalde interface. Hierbij is dus een interface nodig die gebruikt kan worden wanneer de gebruiker zijn auto aansluit op de laadpaal, want de state of charge kan natuurlijk per dag verschillen. Het is niet mogelijk om op voorhand dit realistisch te schatten.

Aankomst- en vertrektijd

Deze twee kunnen op verschillende manieren doorgegeven worden. Bij de dagelijkse aanmelding kan de vertrektijd ingevoerd worden via een bepaalde interface die gebruikt kan worden bij het aansluiten van de auto op de laadpaal en de aankomsttijd kan doorgegeven worden via de interne klok van deze interface. Het is ook mogelijk om de aankomst- en vertrektijd in te voeren voor de komende dagen in een bepaalde agenda. Deze agenda kan geïmplementeerd worden in een interface die vanaf de werkplek te bereiken is.

Nog benodigde hoeveelheid energie

Hoeveel energie de auto nog nodig heeft om op te laden is niet iets dat door de gebruiker berekend en ingegeven hoort te worden. Dit kan berekend worden aan de hand van de state of charge en de accucapaciteit van het voertuig. De state of charge is al behandeld. De capaciteit van de accu zou niet bij iedere aanmelding weer ingegeven hoeven te worden, dit zal namelijk niet wijzigen zolang de gebruiker niet van voertuig verandert. Deze moet eenmalig geregistreerd worden en dit kan via een interface die vanaf de werkplek te bereiken is.

Aansluiting voertuig

Het is niet essentieel voor de berekeningen van het algoritme, maar wel voor het correct aansturen van de verschillende auto's. De identificatie van dit laadpunt is het eenvoudigst te implementeren via een interface die gebruikt kan worden wanneer de gebruiker zijn auto aansluit op een laadpaal.

Identificatie gebruiker

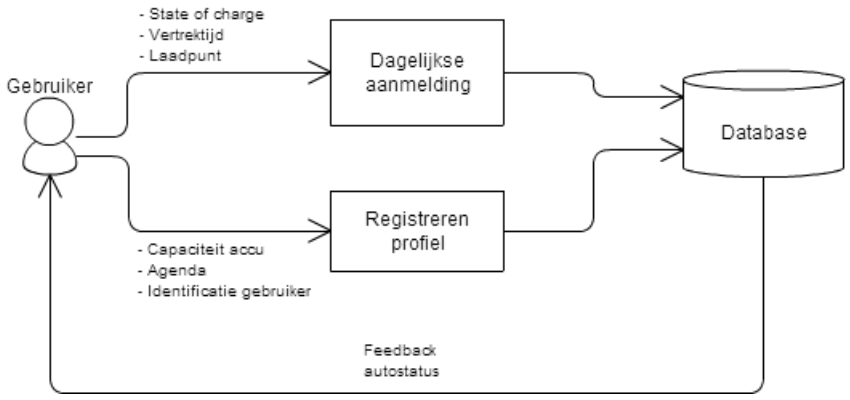
Identificatie van de gebruiker en dus het voertuig is belangrijk bij het geven van feedback naar deze persoon. Al is het om een notificatie te sturen als zijn auto opgeladen is of het weergeven van het oplaadschema via een bepaalde interface. Om dat soort zaken mogelijk te maken, is het nodig om de gebruiker te herkennen bij het aanmelden. Het verhoogt de gebruiksvriendelijkheid als de gebruiker niet bij iedere aanmelding weer zichzelf moet identificeren, maar dat dit herkend wordt. Dit kan aan de hand van een eenmalige registratie via een bepaalde interface die vanaf de werkplek te bereiken is. Dit hoort dan gelinkt te worden aan de dagelijkse aanmelding.

Samenvatting

Om nog even samen te vatten welke informatie er precies nodig is van de gebruiker en wat voor een soort interface daarbij hoort:

1. State of charge
   (a) Dagelijkse aanmelding
   (b) Interface die bruikbaar is in de buurt van laadpunt
2. Vertrektijd
   (a) Dagelijkse aanmelding (i) en/of registreren agenda (ii)
   (b) Interface die bruikbaar is in de buurt van laadpunt (i) en/of interface die toegankelijk is met een PC (ii)
3. Nog benodigde hoeveelheid energie
   (a) Registreren capaciteit van de accu
   (b) Interface die toegankelijk is met een PC
4. Aansluiting voertuig
   (a) Dagelijkse aanmelding
   (b) Interface die bruikbaar is in de buurt van laadpunt
5. Identificatie gebruiker
   (a) Registreren gebruiker
   (b) Interface die toegankelijk is met een PC

Deze informatie moet op een bepaalde manier verkregen worden. Dit is verdeeld onder informatie die ingevoerd wordt bij een dagelijkse aanmelding en informatie die eenmalig geregistreerd dient te worden. Dit is schematisch weergegeven in figuur 5.2.



Figuur 5.2: Overzicht van hoe de informatie verkregen wordt van de gebruiker

5.2. Interviews

Om een beeld te kunnen vormen van wat de gebruiker precies wilt, zijn er drie elektrische rijders geïnterviewd. Iedere geïnterviewde heeft zijn eigen mening omtrent het onderwerp waar wij conclusies van maken. Elk interview is opgedeeld in vier onderdelen: beoordeling van de huidige situatie, opties voor een interface, mogelijke vormen van feedback en de mogelijke risico's die kunnen voorkomen bij verschillende vormen van interfaces. De samenvatting van ieder interview is terug te vinden in de appendix C. De resultaten van de interviews zijn hieronder samengevat.

Conclusie

Huidige situatie Alle rijders zijn het er over eens dat de huidige situatie niet goed werkt en dat er verandering nodig is. De manier waarop verschilt echter nogal per gebruiker.

Interface De één wilt liever een interface aan de laadpaal zelf terwijl de ander liever een applicatie zou zien. Maar als voor een applicatie gekozen zou worden, dan moet deze compact zijn.

Feedback Een notificatie kan zowel per mail als per sms en deze hoort verstuurd te worden wanneer een auto klaar is met opladen terwijl iemand anders in de wachtstrij staat. Deze notificatie moet gestuurd worden naar zowel de persoon in de wachtrij als de persoon die zijn auto moet verplaatsen van de laadpalen naar een andere parkeerplaats. Een dashboard vindt men een leuke optie, maar vooral om het laadgedrag van hun eigen auto te zien.

Risico Het eerlijk invullen en de mogelijke extra drempel die een applicatie zou vormen

5.3. Mogelijkheden

Verschillende mogelijkheden voor de vormgeving van de interface worden opgenoemd en zullen vervolgens tegen elkaar afgewogen worden.

Windows programma

Met een Windows programma wordt een programma bedoeld zoals Outlook, dat geïnstalleerd kan worden op elke computer dat Windows gebruikt. Hierdoor worden de andere besturingssystemen uitgesloten. Voor de implementatie binnen het systeem van Priva zelf zou dit niet tot problemen leiden aangezien alle computers van Windows gebruik maken. Echter is het niet wenselijk als dit systeem toegepast zou worden in een andere instelling die niet exclusief gebruik maakt van Windows. Het is lastig programmeerbaar.

Web interface

Een web interface creëert een webpagina die toegankelijk is met de PC en zelfs met je telefoon, het laatste is echter niet gebruiksvriendelijk omdat webpagina's meestal niet ontwikkeld zijn voor kleine schermen. Daardoor kan deze optie niet gebruikt worden in de buurt van een laadpaal. Interactie (invoeren van data) met een webpagina is eenvoudig te implementeren en deze informatie kan eenvoudig doorgestuurd worden naar een database.

Applicatie

Een applicatie kan geïnstalleerd worden op een smartphone. Omdat een mobiele telefoon zo goed als altijd bij de hand is, kan de applicatie gebruikt worden bij het aanmelden in de buurt van een laadpaal. Uit de interviews is gebleken dat een applicatie compact hoort te zijn en het niet wenselijk is om feedback te krijgen via de applicatie over bijvoorbeeld de autostatus. De nadruk wordt hier gelegd op het zo compact mogelijk houden.

Interface aan laadpaal

Een interface aan een laadpaal kost het meeste tijd om te implementeren. Hiervoor is een fysieke interface nodig die geprogrammeerd moet worden en aangesloten hoort te worden met een netwerk die data kan doorsturen naar de database die weer op zijn beurt communiceert met het algoritme. Het is wel het meest gebruikvriendelijk bij het aanmelden bij de laadpaal.

Afweging

In tabel 5.2 is een afweging gemaakt van de verschillende mogelijkheden aan de hand van een toetsing. De eerste twee mogelijkheden uit de tabel worden getoetst op de gebruiksvriendelijkheid (+ of -), de volgende drie op de haalbaarheid en de gebruiksvriendelijkheid (aan beide voldoen +, aan één eis voldoen ± of aan beide niet -) en de laatste op de haalbaarheid (+ of -). Het totaal beoordeelt of het gemiddeld meer neigt naar een + of een -. Het is uit de tabel af te leiden dat om een interface te verkrijgen die alle functionaliteiten heeft, er een combinatie van twee mogelijkheden nodig zal zijn.

Een combinatie van een web interface en een applicatie. Een interface aan de laadpalen is een goed alternatief voor de applicatie, het is echter veel meer werk om deze functionerend te krijgen. Vooral om deze reden is gekozen om niet een interface te maken aan de laadpaal, het wordt wel aangeraden voor verder onderzoek.

Tabel 5.2: Afweging van de verschillende mogelijkheden. De eerste twee mogelijkheden uit de tabel worden getoetst op de gebruiksvriendelijkheid (+ of -), de volgende drie op de haalbaarheid en de gebruiksvriendelijkheid (aan beide voldoen +, aan één eis voldoen ± of aan beide niet -) en de laatste op de haalbaarheid (+ of -).

	Windows programma	**Web interface**	**Applicatie**	**Interface aan laadpaal**
Dagelijkse aanmelding	-	-	+	+
Registreren profiel	+	+	-	-
Bruikbaar in de buurt van laadpaal	-	±	+	+
Toegankelijk vanaf werkplek	+	+	±	-
Feedback laadgedrag	±	+	±	-
Communicatiemogelijkheid met algoritme	-	+	+	-
Totaal	**-**	**+**	**+**	**-**

5.4. Specificaties

Nu dat er besloten is welke twee interfaces ontwikkeld gaan worden, kunnen er duidelijke eisen opgesteld worden aan deze interfaces.

Applicatie

De applicatie zal gebruikt worden bij de dagelijkse aanmelding en hoort volgens de gebruikers compact te zijn. Dit houdt in dat alleen essentiële informatie opgevraagd mag worden. Het oplaadpunt, de aankomst- en vertrektijd en de state of charge zal ingevuld moeten worden door de gebruiker waarna het mogelijk zal zijn om zijn auto op te laden. Dit is namelijk niet op te vragen via de hardware en de informatie verschilt per aanmelding, dus is het niet nauwkeurig te schatten. Verder is het wenselijk dat de applicatie werkt op zoveel mogelijk mobiele telefoons.

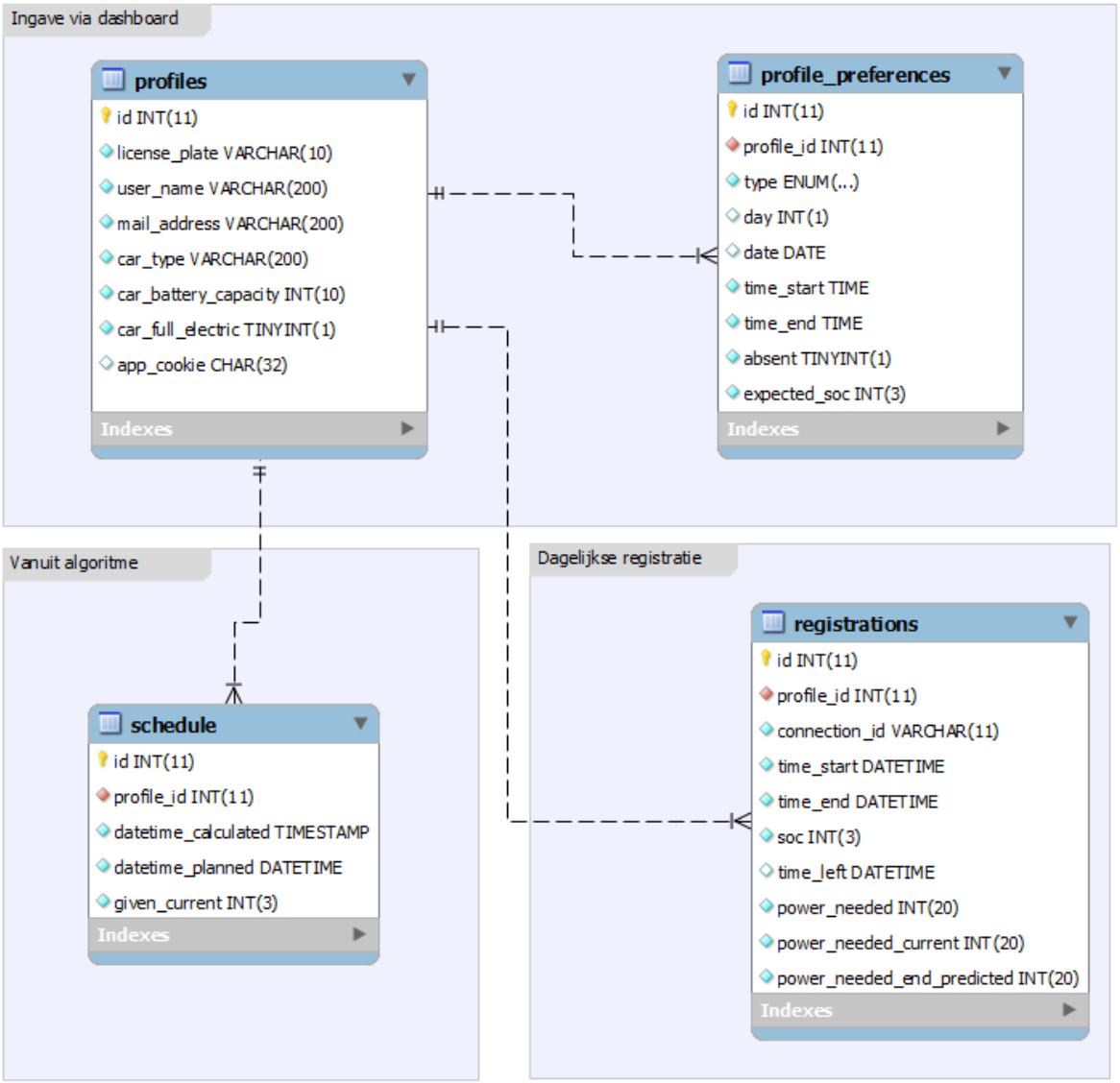
Web interface

De web interface zal het mogelijk maken om een profiel aan te maken, zodat de gegevens die vrijwel nooit veranderen eenmalig geregistreerd kunnen worden. Gegevens zoals de naam van de gebruiker, het kenteken van zijn auto, zijn type elektrische voertuig, de capaciteit van de accu, met hoeveel stroom de auto maximaal op kan laden en of de auto volledig elektrisch is. Met deze informatie moet de gebruiker en zijn auto herkend kunnen worden als het nodig zal zijn om feedback terug te koppelen naar de gebruiker en hoort de informatie waarde toe te voegen aan het prioriteringsalgoritme. Verder moet het mogelijk zijn om een agenda voor minstens de aankomende week in te voeren via deze interface. De feedback naar de gebruiker toe moet vormgegeven worden door inzicht in het oplaadschema. Zo zijn de gebruikers altijd op de hoogte van de laadstatus van hun auto's. Dan is het minder kritisch dat een notificatie via sms of e-mail verstuurd wordt naar de gebruiker als zijn auto volledig opgeladen is en omgewisseld dient te worden met een gebruiker die in de wachtrij staat.

5.5. Implementatie

De gehele interface draait om de profielen. Deze worden eenmalig geregistreerd en bevatten de variabelen die zichtbaar zijn in figuur 5.3. Van deze profielen zijn enkele instanties afhankelijk. Deze instanties nemen de identificatievariabelen over van de profielen. Met deze instanties worden de

agendavoorkeur van ieder profiel, de dagelijkse aanmeldingen en de terugkoppeling vanuit het algoritme bedoeld.



Figuur 5.3: Datamodel van de interface

Applicatie

Zoals al eerder vermeld is, moet de applicatie compact zijn zodat het geen extra drempel vormt voor de gebruikers en het moet voor zo veel mogelijk gebruikers beschikbaar zijn.

Reikwijdte

Het is momenteel zo dat 96% van de smartphones het besturingssysteem iOS of Android gebruiken [27]. Het is om deze reden voldoende om de applicatie beschikbaar te stellen voor alleen deze besturingssystemen. Het is tijdrovend om deze code afzonderlijk voor beide besturingssystemen te ontwikkelen. Daarom is een platform zoals Phonegap ideaal [28]. Hier kan in HTML, CSS en JavaScript geprogrammeerd worden en een applicatie ontwikkeld worden voor beide besturingssystemen. Er is dus geen kennis nodig van de gehele werking van beide besturingssystemen, slechts kennis van de talen waarmee websites ontwikkeld kunnen worden die toch al vereist is om het dashboard te maken. Dit bespaart enorm veel werk.

Compacte applicatie

De informatie die hier verzameld wordt, is essentiële informatie die opgevraagd moet worden om het algoritme een eerlijk prioteringssysteem op te laten stellen en is informatie die niet op een andere manier opgevraagd kan worden zonder dat de gebruiker iets hoeft in te voeren. De informatie die nodig is voor het maken van het laadschema zijn, zoals eerder vermeld, de state of charge, de aankomst- en vertrektijd, de identificatie van de gebruiker en het laadpunt waar de auto aan aangesloten is. Als al deze informatie iedere keer ingevuld dient te worden, kost het redelijk wat tijd. De applicatie maakt daarom gebruik van een cookie die na de eerste keer gebruiken van de applicatie zal onthouden welke gebruiker dit is. De aankomsttijd is verder ook iets dat overbodig is om in te voeren, aangezien dit opgehaald kan worden uit de interne klok van een smartphone op het moment dat de aanmelding gestuurd wordt.
Als er weinig getypt moet worden in de applicatie, kost het minder tijd om deze in te vullen en is het daarom gebruiksvriendelijker. Met deze gedachte is er voor gekozen om het laadpunt waar de auto wordt aangesloten aan te geven met een drop-down menu en wordt de state of charge ingevoerd met een slider die van 0 tot 100% varieert.

Web interface

De web interface heeft drie functionaliteiten. Het registreren van profielen, het invullen van een agenda en het terugkoppelen van het laadschema naar de gebruiker.

Registreren van een profiel

De informatie die nodig zal zijn voor het prioriteren van bepaalde auto's en die eenmalig ingevoerd dienen te worden zal in dit onderdeel behandeld worden. Volgens de specificaties moet deze informatie toevoegen aan het prioriteringalgoritme. Dit wordt nagegaan in tabel 5.3.

Tabel 5.3: De toevoeging van de informatie die ingevoerd moet worden bij het registreren van een profiel en de voornaamste reden waarom deze informatie via de web interface ingevoerd wordt

	Toevoeging	**Reden**
Naam gebruiker	Herkenning	Liever eenmalig invoeren dan dagelijks
Kenteken	Herkenning	Liever eenmalig invoeren dan dagelijks
Type elektrisch voertuig	Herkenning	Liever eenmalig invoeren dan dagelijks
Capaciteit accu	Gebruikt voor het berekenen van het nog op te laden vermogen van de auto	Kan niet uitgelezen worden
Fases gebruikt bij opladen	Ander laadgedrag	Kan via de hardware herkend worden, maar is fijner om op te anticiperen
Maximale stroom opladen	belangrijk voor het algoritme, bescherming van de auto	Kan niet uitgelezen worden
Volledig elektrisch	Heeft hogere prioriteit	Kan niet uitgelezen worden

Agenda

De agenda is een middel om te voorspellen wanneer de gebruikers de komende tijd zullen gaan opladen. Dit middel moet eenvoudig in te vullen zijn en een goed overzicht geven van de komende tijd. Een goed overzicht van de komende tijd krijg je door voor een langere tijd een agenda vast te leggen. Daarom wordt gekozen voor een agenda van een maand in plaats van een week. Om de agenda eenvoudig in te kunnen vullen, is het nodig dat voor zoveel mogelijk dagen zo snel mogelijk een bepaald patroon ingevoerd kan worden. Dit komt op verschillende manieren tot stand. Het is mogelijk om een default aankomst- en vertrektijd in te voeren die dan automatisch doorgevoerd wordt voor iedere dag in de

komende maand. Daarnaast kan voor een dag in de week (bijvoorbeeld maandag) een vast aankomst- en vertrektijd ingevoerd worden voor de hele maand. Vrije dagen worden ingevoerd door het vakje absent aan te vinken. Als laatste is het natuurlijk ook mogelijk om voor iedere dag afzonderlijk wat in te voeren.

Laadschema

Het dashboard oftewel de terugkoppeling op de web interface zal informatie bevatten over wie er op het moment precies aangesloten zijn, wat hun huidige state of charge is en wat het laadschema is voor de komende tijd. Deze informatie over het laadschema wordt verzameld vanuit het algoritme, zie figuur 5.3. Wie er precies aangesloten zijn en wat hun state of charge is, wordt doorgegeven door de dagelijkse aanmeldingen.

Notificatie naar de gebruiker

Uit de conclusie van de interviews bleek dat men een notificatie via e-mail of sms wilt ontvangen wanneer zijn auto volledig opgeladen is en een andere gebruiker in de wachtrij staat. De gebruiker wiens auto volledig opgeladen is krijgt een melding dat de auto gewisseld moet worden met een auto die in de wachtrij staat. Een wachtrij is niet geïmplementeerd, dus zal dat niet mogelijk zijn. Dit wordt aanbevolen voor verder onderzoek. Het is wel mogelijk om een notificatie te sturen naar de gebruiker als zijn auto volledig opgeladen is. Dit zal gebeuren via e-mail en niet sms. Er wordt gekozen voor e-mail omdat dit makkelijker te versturen is via de webserver. De Compri HX (zie sectie 6.1) krijgt de status van de auto binnen, als deze volledig opgeladen is, zal het verbindingscript communiceren met de interface dat een auto volledig opgeladen is. Het oplaadpunt wordt gekoppeld aan de gebruiker en de web interface zal dan een email versturen naar de gebruiker.

Gegevens opslaan

Uiteindelijk zijn de interfaces instanties waarmee informatie doorgevoerd kan worden en niet in opgeslagen worden. Het is noodzakelijk dat deze interfaces deze informatie gaat doorspelen naar een database. Deze database zal de informatie over de verschillende registraties, met bijbehorend profiel, gereed hebben voor het algoritme. Het algoritme verwerkt deze informatie en creëert een laadschema die vervolgens teruggestuurd wordt naar de database, die deze opslaat en gereed heeft voor de interface. Voor de database wordt gebruik gemaakt van MySQL. MySQL is de meest gangbare database voor web interfaces mede doordat het opensource is. Het is daarom een logische stap om voor MySQL te kiezen. De data wordt in MySQL opgeslagen in tabellen die eenvoudig te wijzigen zijn en uit te lezen zijn in Matlab. Aangezien het algoritme gebruik maakt van Matlab, is dit een goede optie.

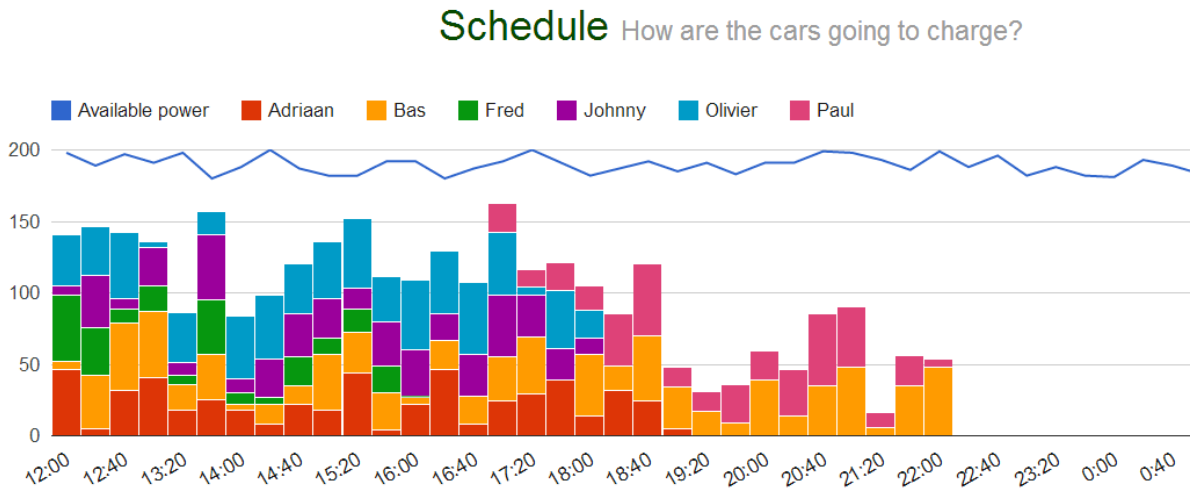
5.6. Realisatie

Beide interfaces worden getest door informatie in te voeren en te controleren of deze informatie vervolgens daadwerkelijk gewijzigd wordt in de database. Als dit het geval is, kan er gesteld worden dat de interfaces gereed zijn voor integratie in het geheel.

Web interface

Om de werking van de web interface te testen zijn verschillende werkende onderdelen nodig. Een laadschema moet gegenereerd kunnen worden en de applicatie moet een registratie kunnen doorsturen naar de database. Een willekeurige test is gegenereerd en weergegeven in figuur 5.4. Het is mogelijk om deze data door te sturen van het algoritme via de database naar de web interface.

Een registratie van een profiel ziet er uit als in figuur A.1 in appendix A en wordt verwerkt in de database. Ook dagelijkse aanmeldingen via de applicatie worden verwerkt in de database en zijn doorgestuurd naar de web interface zoals te zien is in figuur A.2 in appendix A.



Figuur 5.4: Weergeven van het oplaadschema op de dashboard

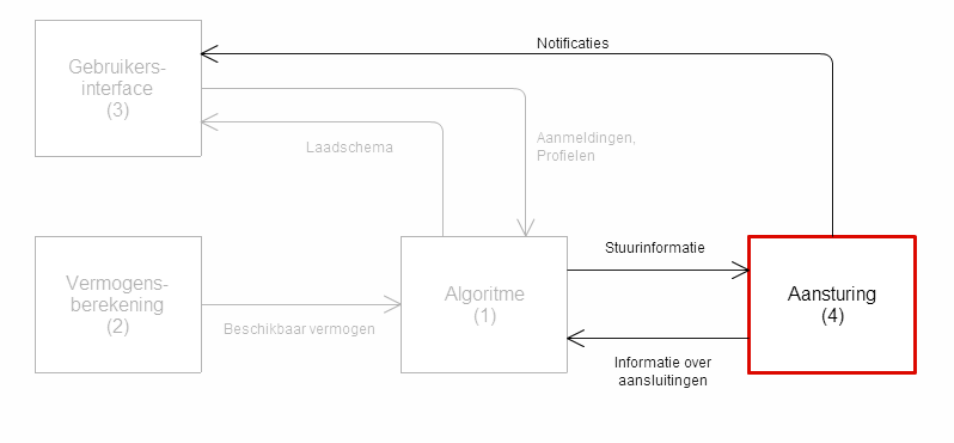
Applicatie

In appendix B is het zichtbaar hoe de vormgeving van de applicatie er uit ziet. Wijzigingen in deze applicatie zijn op een smartphone met het besturingssysteem Android getest. De informatie wordt doorgegeven naar de database en is uit te lezen in Matlab.

6

Aansturing

De aansturing van het systeem omvat de communicatie van alle gegevens van en naar de hardware. In figuur 6.1 is te zien hoe de aansturing zich bevindt in het gehele softwaresysteem.



Figuur 6.1: Uitgebreid overzicht van het product

In het hoofdstuk over het algoritme (sectie 3.4 en 3.5) is reeds uiteengezet welke informatie de aansturing zou moeten verwerken. Deze vastgestelde informatieset is samengevat in tabel 6.1. De vorm van de waardes is vastgesteld door de hardware [1].

Tabel 6.1: Alle variabelen van een aansluiting, die door de aansturing gecommuniceerd moeten worden

Richting	Informatie	Vorm
Van algoritme naar hardware	Toebedeelde stroom	Waarde van 0 of tussen 3 en 32 op twee decimalen nauwkeurig
Van hardware naar algoritme	Stroomlimiet 16 of 32 ampère	Eén bit
	Geleverde energie sinds aankoppeling	Waarde van 0 tot zo hoog mogelijk
	Status van de auto	Vier discrete waardes die de volgende toestanden representeren: 1. Geen verbinding 2. Auto verbonden 3. Aan het opladen (evt. met koelingverzoek) 4. Fout in het systeem

Het algoritme stuurt voor alle aansluitingen aan hoeveel stroom er gebruikt mag worden om op te laden. Door vóór het draaien van het algoritme de stroomlimiet en geleverde energie per aansluiting uit te lezen kan het algoritme respectievelijk bepalen hoeveel stroom de aansluiting fysiek aan zou kunnen en hoeveel de aangesloten auto al is opgeladen. Als een status van een auto verandert, verandert dus de situatie van het oplaadsysteem. Het algoritme moet dan opnieuw uitgevoerd worden, om het oplaadschema voor de nieuwe situatie te berekenen.

6.1. Mogelijkheden voor realisatie

Zoals in het programma van eisen al gesteld is, is vanuit Priva de wens om het systeem te integreren in hun software en hardware. Priva heeft een eigen productlijn van software en hardware voor grootschalige communicatie en digitale en analoge aansturing. Priva past dit voornamelijk toe op het gebied van gebouwbeheer, maar hun systemen zijn algemeen toepasbaar.

Uiteraard zouden er ook andere opties voor communicatie zijn. Omdat de informatie die verstuurd moet worden echter ook waardes zijn, is er al snel een protocol nodig om de communicatie te kunnen realiseren. Interpretatie van een protocol ontwikkelen gaat vaak veel tijd in zitten, laat staan zelf een protocol ontwikkelen. Door een systeem van Priva te gebruiken, is er weinig tijd nodig om de communicatie in werking te stellen. Hierdoor kon de focus van het project gelegd worden op het algoritme. Bovendien zijn er geen zware eisen wat betreft minimale communicatiesnelheid of maximaal bandbreedtegebruik, dus vrijwel elke optie is voldoende.

Priva-systeem

Keuze dus voor de Priva hardware en software. Deze keuze voor implementatie heeft bovendien de beste prijs-kwaliteitverhouding, aangezien in dit onderzoek gratis gebruik gemaakt kan worden van de onderdelen die normaal honderden euro's kosten. De hardware en software is wel intellectueel eigendom van Priva, waardoor specifieke verslaglegging van de implementatie slecht mogelijk is.

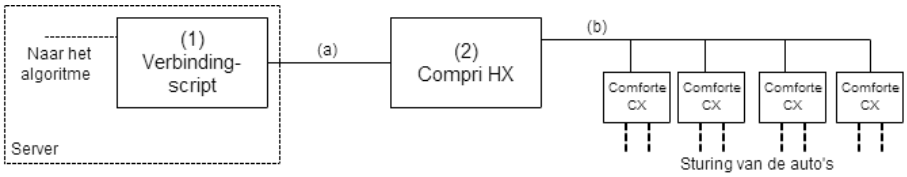
Een systeem van Priva is altijd opgebouwd uit een regelunit, genaamd Compri HX, en waar nodig naregelunits (Comforte CX genoemd). Beide type units kunnen vanuit speciale software van Priva geconfigureerd en ingesteld worden. Dit programma is een zeer high-level configuratieprogramma, waarbij er alleen aandacht besteed hoeft te worden aan de functionaliteit. Implementatie op het niveau van protocollen wordt automatisch geregeld.

Figuur 6.2: De Compri HX van Priva

De Compri HX is een minicomputer met een CPU, RAM-geheugen en digitale en analoge IO. Er bestaan verschillende uitvoeringen van, waarbij duurdere uitvoeringen sneller zijn en meer IO-mogelijkheden hebben. Vanwegen redenen die in sectie 6.2 uitgelegd zijn, is de keuze gevallen op de Compri HX 6E. Deze beschikt namelijk over een ethernetaansluiting, welke essentieel is voor de implementatie van . Tevens beschikt deze uitvoering over een 32-bit microprocessor met 8 MB DRAM en 1 MB SRAM.

6.2. Vormgeving

Door de aansturing te realiseren met hardware van Priva, ziet de aansturing eruit als weergeven in figuur 6.3. Voor de aansturing van het oplaadsysteem zal één regelunit voldoende zijn. Eén Compri HX kan namelijk communiceren met maximaal 75 Comforte CX-en. Dat aantal is meer dan genoeg voor het systeem.



Figuur 6.3: Vormgeving van de aansturing

De Comforte CX is in staat om digitale en analoge in- en output uit te lezen, te verwerken en aan te sturen. De scheiding van het software- en het hardwaresysteem is dan ook bepaald tussen de Compri HX en de Comforte CX. De configuratie van de Comforte CX-en wordt behandeld in de thesis over de hardware [1]. Zij gebruiken de Comforte CX om hun oplaadhardware te realiseren. De in- en output van de Comforte CX aan de kant van de Compri HX is de uiteindelijke verbinding tussen de software en de hardware. Deze in- en outputs zijn als volgt vormgegeven door de hardware: er zijn twee voertuigaansluitingen per Comforte CX. Dat betekent dat per Comforte CX twee sets van de informatieset beschreven in tabel 6.1 gecommuniceerd moeten worden.

Omdat het algoritme op een aparte computer - de server - draait, moet er een script (1) komen dat kan communiceren met het algoritme en de centrale unit van de Privahardware, de Compri HX (2). Door het script te draaien op dezelfde server als waar het algoritme draait, kan dit script via command prompt communiceren met het algoritme. De communicatie met de Compri HX moet dan nog vormgegeven worden (a). De Compri HX communiceert op zijn beurt via (b) met de Comforte CX-en.

Communicatie verbindingscript en Compri HX (a)

Het verbindingscript zal draaien op de server, een computer. Een computer kan op enkele manieren met een Compri HX verbonden worden: via een RS232-poort, een RS485-poort en een ethernet-poort. Een verbinding over ethernet (10 Mb/s) is sneller dan een seriele verbinding (19,2 kb/s). Bovendien zit niet op elke pc meer een seriele poort, en is een ethernetverbinding flexibeler. Een logische keuze

dus voor een ethernetverbinding.

Communicatie met de Compri HX gaat doorgaans via speciale software van Priva. Echter, omdat het algoritme zo vrij mogelijk ontwikkeld moest kunnen worden, is dit geschreven in Matlab. Een directe communicatieverbinding tussen Matlab en de Compri HX zou dus de meest ideale situatie zijn. Hiervoor ondersteunt de Compri HX de universele koppeling XML. E**x**tensible **M**arkup **L**anguage is een standaard voor het beschrijven van data. Met een XML-verbinding met een Compri HX is het mogelijk om zowel gegevens (zowel losse bits als 32-bit waardes) uit te lezen, als zulke gegevens te versturen. Dit gaat allemaal via TCP/IP.

Dus om de specificaties van de communicatie tussen het script en de Compri HX samen te vatten:

- Verbinding via ethernet
- Communicatie via TCP/IP
- Syntax aan de hand van XML

Communicatie Compri HX en Comforte CX-en (b)

Voor de communicatie tussen de Compri HX en de Comforte CX-en is eigenlijk geen keuze. De enige manier waarop deze twee elementen kunnen communiceren is met BACnet via een RS485-verbinding. BACnet is een gerenommeerde open standaard voor datacommunicatie in gebouwbeheersystemen.

Voor elke aangesloten Comforte CX moeten er, zoals eerder beschreven, twee sets aansluitinginformatie doorgestuurd kunnen worden, een set zoals beschreven in tabel 6.1. Om in het algoritme de informatie van een aansluiting te kunnen matchen met de informatie van een voertuig, moet de gebruiker (zoals beschreven in het hoofdstuk *Interface*) bij zijn aanmelding aangeven aan welke aansluiting hij staat. De identifier voor de aansluiting die dan opgegeven wordt, moet dan ook in de communicatie tussen de Compri HX en een Comforte CX gedefiniëerd zijn. Dat komt erop neer dat de softwarematige verbindingen genummerd moeten worden. De aansluitingen van de eerste gekoppelde Comforte CX zijn dan nummer 1 en nummer 2, de aansluitingen van de tweede Comforte CX dan nummer 3 en 4, enzovoort. Deze nummers zullen voor de eindgebruikers ook duidelijk bij de aansluiting te zien moeten zijn.

Met speciale software van Priva kan zeer eenvoudig worden geconfigureerd welke bits en waardes via het BACnet-protocol gecommuniceerd moeten worden tussen de Compri HX en de Comforte CX-en. Hoe de configuratie exact werkt kan niet specifieker beschreven worden, omdat die methode intellectueel eigendom is van Priva. De te communiceren waardes kunnen in ieder geval high-level geconfigureerd worden, waarna software van Priva alle benodigdheden ter uitvoering van het BACnet-protocol opzet.

6.3. Configuratie

Verbindingscript (1)

De Compri HX heeft geen mogelijkheid om via de ethernetverbinding communicatie met de server te starten. De communicatie tussen het verbindingscript en de Compri HX zal dus altijd gestart moeten worden door het verbindingscript.

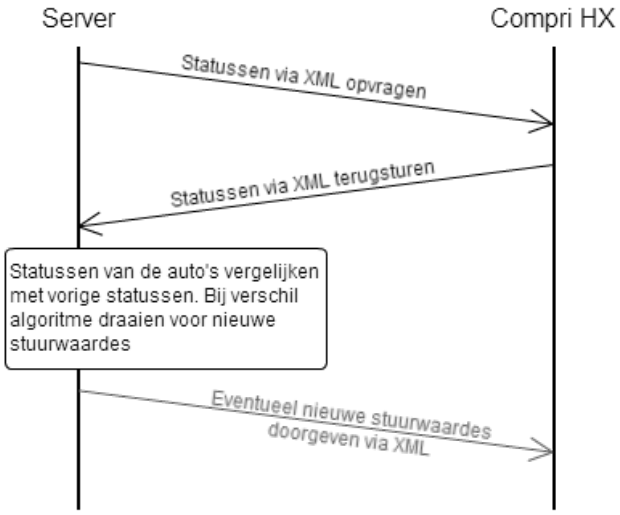
Communicatie bij uitvoeren algoritme

Als het algoritme uitgevoerd wordt, is er enkele informatie nodig vanuit de aansturing. Het algoritme kan dan aan het verbindingscript vragen om communicatie op te zetten met de Compri HX om de benodigde waardes op te halen. Met een HTTP-request aan de Compri HX wordt een XML-pagina met alle gegevens teruggestuurd. Zodra deze gegevens dan zijn opgehaald kan het algoritme verder.

Aan het einde van het algoritme, als de stuursignalen bekend zijn, kan het algoritme weer aan het verbindingscript vragen om communicatie te starten met de Compri HX. De stuursignalen kunnen in een HTTP-request doorgegeven worden aan de Compri HX.

Communicatie bij statusverandering auto

Als er een statusverandering is bij een auto, dan is de Compri HX daarvan op de hoogte, maar de server niet. De Compri HX kan echter geen communicatie starten. Om het algoritme toch te kunnen starten aan de hand van een statusverandering, zal de server dus de Compri HX moeten pollen. Deze polling is schematisch weergegeven in figuur 6.4.



Figuur 6.4: Polling door de server van de Compri HX

Het verbindingscript moet dus een component krijgen dat periodiek gestart wordt, waarbij actuele statussen uitgelezen worden uit de Compri HX. De actuele statussen worden vervolgens vergeleken met statussen van de vorige keer, en als er verschil is, moet er waarschijnlijk gehandeld worden. Elke statusverandering betekent uiteraard wat anders. Er moet dus ook verschillend gereageerd worden op verschillende veranderingen. Alle mogelijke statusveranderingen en hun betekenissen zijn uiteengezet in tabel 6.2. Tevens is er per statusverandering bepaald wat de gewenste vervolghandeling is.

Er zijn dus enkele acties die het verbindingscript moet kunnen uitvoeren of aansturen. De handelingen zullen op verschillende manieren uitgevoerd kunnen worden. Dit kan samengevat worden in onderstaande lijst.

1. Algoritme starten. Dit kan door het Matlab-script van het algoritme aan te roepen.

2. Foutmelding doorgeven aan systeembeheer. Dit kan in de vorm van een e-mail, via de *Interface* uit te voeren zoals in sectie 5.5 beschreven.

3. Auto afmelden in systeem. Dit kan door de actieve registratie in de database van de betreffende aansluiting af te melden.

4. Bericht sturen naar gebruiker. Dit kan net zoals de foutmelding in de vorm van een e-mail.

Informatiecontrole

Bovendien kan het verbindingscript ook eventueel verbinding maken met de database van de interface, om te controleren of gegevens die via de app of het dashboard ingegeven zijn overeen komen met de statussen van de aansluitingen. Als er bijvoorbeeld iemand aangemeld is op een paal waar volgens de hardware niets is aangesloten, kan het zijn dat de hardware defect is. Hier zou dan bericht over naar de systeembeheerder gestuurd kunnen worden.

Tabel 6.2: Betekenis van statusveranderingen

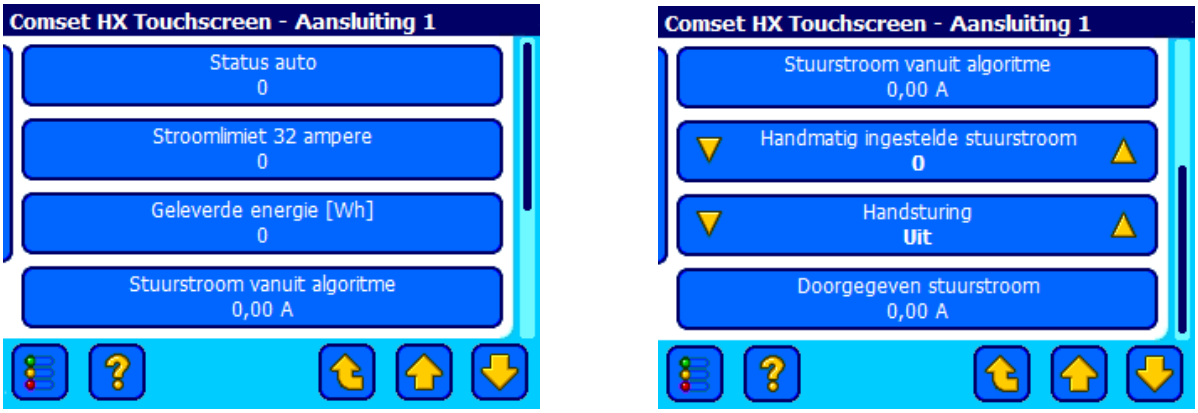
Van	Naar	Betekenis	Actie
Niet verbonden	Verbonden	Er is een auto aangesloten	Algoritme opnieuw draaien, zodat nieuwe auto meegenomen wordt
Niet verbonden	Opladen	Zou niet voor moeten komen. Dit betekent dat de hardware een auto laat opladen zonder aansturing vanuit de software	Foutmelding doorgeven aan systeembeheer
Verbonden	Niet verbonden	Een auto die gereed was, of niet aan het opladen was wordt losgekoppeld	Auto afmelden in het systeem
Verbonden	Opladen	Een auto begint met opladen	Geen speciale actie nodig
Opladen	Verbonden	Auto is klaar met opladen	Algoritme opnieuw draaien, bericht sturen naar gebruiker
Opladen	Niet verbonden	Een auto die aan het opladen was is losgekoppeld	Algoritme opnieuw draaien, auto afmelden in het systeem
Elke status	Foutmelding	Er is een error met de verbinding	Foutmelding doorgeven aan systeembeheer

Compri HX (2)

De Compri HX is een krachtige minicomputer, maar zoals deze in figuur 6.3 opgesteld is, is zijn enige taak om BACnet-informatie om te zetten naar een XML-pagina en andersom.

De Compri HX is echter tot veel meer in staat. Vanuit het programma van eisen wordt gevraagd om een mogelijkheid voor handmatige besturing. Indealiter is dit voor een product niet nodig, maar het is onontkoombaar dat de mens het soms beter weet dan bijvoorbeeld het algoritme uit hoofdstuk 3. Er kunnen zich altijd tijdelijke situaties voordoen die niet voorzien zijn bij het opstellen van het algoritme. Op zo'n moment moet het mogelijk zijn om de stuurwaardes van het algoritme te overrulen en andere waardes naar de hardware te sturen.

De Compri HX ondersteunt aansturing via een eenvoudig programmeerbare touchscreen. Hierop is het mogelijk om via de speciale Priva-software waardes weer te geven of in te stellen. Dat eerste kan gebruikt worden om alle informatie van de aansluitingen weer te geven, dus de status van de auto, de gebruikte energie sinds het aansluiten en of de auto zijn fase-aansluiting moet delen. Ook kan de stuurwaarde vanuit het algoritme weergegeven worden. Dit lijstje kan dan per aansluiting uitgebreid worden met een handmatig in te stellen waarde en een schakelknop, om te kiezen of de handmatige waarde of de stuurwaarde vanuit het algoritme doorgestuurd moet worden naar de hardware. Het touchscreen zou er dan uitzien als weergegeven in figuur 6.5.



(a) Deel 1 van het menu

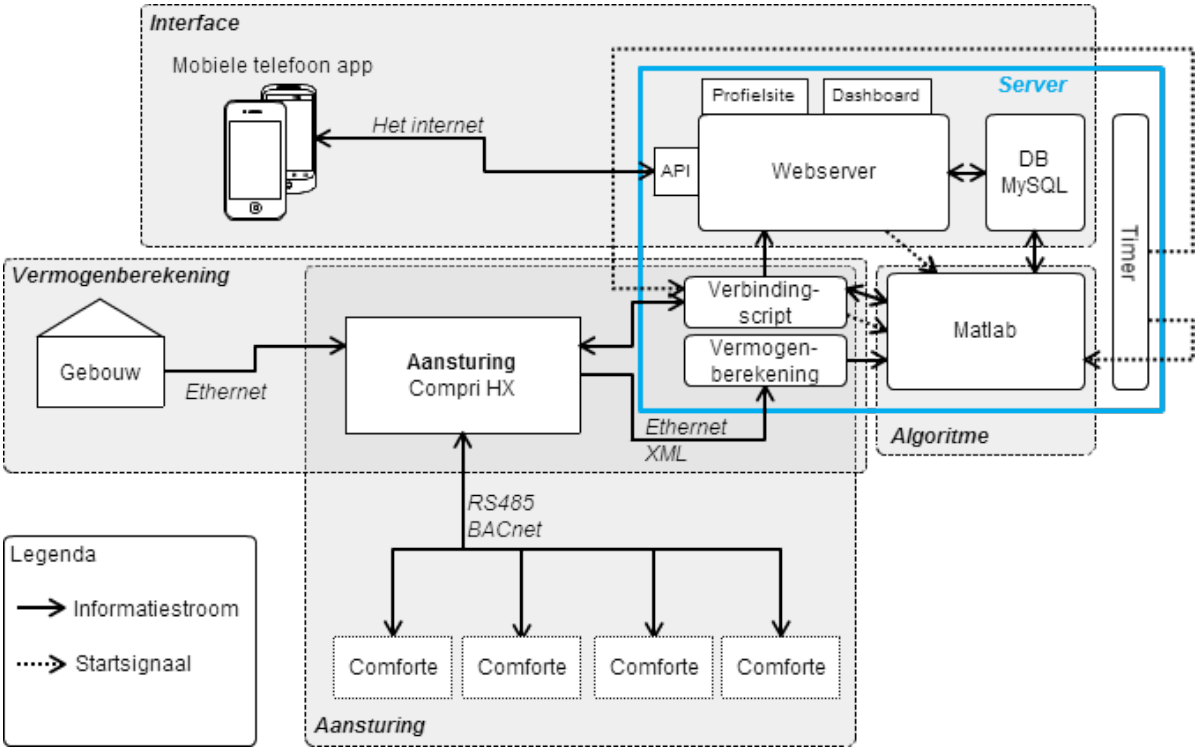
(b) Deel 2 van het menu

Figuur 6.5: De touchscreen interface voor handmatige sturing van een aansluiting

Door de handmatige overwrite te implementeren in de Compri HX, en niet in de server zelf, is er nog een zekere veiligheid aan het systeem toegevoegd. Hierdoor kunnen de oplaadpalen namelijk ook nog aangestuurd worden, als de verbinding tussen de Compri HX en de server wegvalt.

7

Integratie



Figuur 7.1: De implementatie van het gehele softwaresysteem

7.1. Het geheel

In de vorige vier hoofdstukken zijn alle onderdelen stuk voor stuk overwogen, vormgegeven en getest. Om met alle onderdelen een goed geheel te kunnen vormen, moet alles geïntegreerd worden tot één systeem. Deze integratie is schematisch weergegeven in figuur 7.1.

Wat direct opvalt is de overlap op twee plekken. Zowel de aansturing als de vermogensberekening maken gebruik van de Compri HX. Het verbindingscript, de webserver, de database en het algoritme zitten in dezelfde box: de server. Bovendien zijn er koppelingen tussen enkele onderdelen te zien: tussen het verbindingscript en de webserver, tussen het verbindingscript en het algoritme en tussen Matlab en de database. In dit hoofdstuk zullen de implementatie van de twee overlappingen als de koppelingen verder behandeld worden.

7.2. Server

Omdat alle onderdelen van het systeem minstens een stuk(je) software hebben, is de meest gemakkelijke integratie om alle stukken software op één computer te zetten. Deze noemen we de server. Er is voor gekozen om de server uit te rusten met het besturingssysteem Windows 7. Omdat de benodigde programma's en functies in elk gangbaar besturingssyteem mogelijk is, is het besturingssyteem gekozen dat het meest voor de hand lag.

7.3. Algoritme

Het algoritme is dus in Matlab ontwikkeld. Op de server moet dus ook Matlab geïnstalleerd staan om het algoritme uit te kunnen voeren.

Informatie met verbindingscript uitwisselen

Als het algoritme informatie nodig heeft van het verbindingscript, kan deze eenvoudig intern in Matlab aangeroepen worden. Het verbindingscript is namelijk ook in Matlab geschreven. Nadat het script is aangeroepen kan deze verbinding leggen met de Compri HX, de gegevens interpreteren en doorgeven aan het algoritme. Verbinding tussen de server en de Compri HX gaat via http-requests.

Aan het einde van het algoritme, als er gegevens naar de hardware gestuurd moet worden, kan in Matlab de berkende waardes doorgegeven worden aan het Matlab-verbindingscript, die de waardes in een http-request verpakt en verstuurt naar de Compri HX.

Vermogenberekening aanroepen

Het script dat de vermogenberekening uitvoert is ook geschreven in Matlab. De communicatie van het algoritme met de vermogensberekening gaat dus hetzelfde als met het verbindingscript. De vermogenberekening zal ook op dezelfde manier informatie uit de Compri HX halen als het verbindingscript. Het verschil is dat de vermogenberekening geen informatie terugkoppeld naar de Compri HX.

Informatieuitwisseling met de database

Een directe verbinding tussen het algoritme en de database is mogelijk. Matlab heeft een database toolbox, maar in een Matlab-script is het eenvoudiger en sneller om gebruik te maken van de api's van zowel MySQL als Matlab [29]. Een script dat daarvan gebruik maakt is te vinden op de Matlab-site [29] en vrij te gebruiken. Met behulp van dat script is met de functie "mysql(QUERY)" de database aan te roepen, waarbij QUERY dan de uit te voeren SQL-query is.

Getimed starten

In sectie 3.6 is bepaald dat auto's in oplaadblokken van 15 minuten ingedeeld worden. Dat betekent dat het algoritme na uiterlijk 15 minuten opnieuw uitgevoerd moet worden. Hiertoe gebruiken we de Task Scheduler van Windows. Deze Task Scheduler kan elke 15 minuten een command line script uitvoeren. Matlab beschikt namelijk over de mogelijkheid om via de command line van Windows aangeroepen te worden. Met het commando "matlab -r '*matlab-script*' -nodisplay" kan een Matlab-script gestart worden.

7.4. Interface

Voor de interface is een webserver en een MySQL-database nodig. Deze onderdelen kunnen heel eenvoudig opgezet worden met het programma "XAMPP". Dit programma bevat Apache, MySQL en PHP, respectievelijk een opensource webserver, de opensource database en een scripttaal voor webservers. Met behulp hiervan kan een website opgezet worden, waarin de profielen ingegeven en het dashboard geraadpleegd kunnen worden. Bovendien kan in PHP ook de api geschreven worden waarmee de app verbinding kan maken en wijzigingen kan doorgeven en informatie kan ontvangen. Voor de website is de volgende software gebruikt:

- Yii, een framework voor PHP [30]
- Google Charts, een framework om grafieken weer te geven [31]

- jQuery, een framework voor Javascript [32]

Algoritme starten

Als er in de mobiele applicatie een nieuwe registratie is doorgevoerd, of als er gegevens zijn gewijzigd van een aangesloten auto, dan moet het algoritme gestart worden. PHP heeft de mogelijkheid om een command line uit te laten voeren met de functie "exec()". Op die manier kan het algoritme via de interface gestart worden.

7.5. Aansturing: verbindingscript

Vanuit de aansturing-module moet het verbindingscript geïntegreerd worden in de server. Hoe het verbindingscript communiceert met het algoritme is al beschreven in sectie 7.3.

Gestart worden door timer

Net zoals voor het algoritme, is het voor het verbindingscript van belang om op gezette tijden aangeroepen te worden. Ook hiervoor is het mogelijk om de Windows Task Scheduler te gebruiken.

Communicatie met Compri HX

In sectie 6.2 is de communicatie tussen het verbindingscript en de Compri HX al vormgegeven met XML over TCP/IP. De XML-pagina van de Compri HX is in Matlab op te halen met de functie "urlread(*URL*)". Vervolgens kan met de functie "xmlreadstring(*DATA*)" de XML-gegevens geïnterpreteerd worden, waarbij *DATA* de ingelezen XML-datastring is.

7.6. Compri HX

Zowel in het hoofdstuk *Vermogen* als in het hoofdstuk *Aansturing* is gekozen voor het gebruik van een Compri HX. Deze twee implementaties kunnen prima naast elkaar op de Compri HX werken, zonder elkaar in de weg te zitten. Voor de communicatie tussen de server en de Compri HX kunnen ook twee aparte XML-verbinding zijn, één voor de vermogenberekening en één voor het verbindingscript.

8

Conclusie en aanbevelingen

8.1. Reflectie onderzoeksvragen

De onderzoeksvragen, die de leidende draad door het gepresenteerde onderzoek voorstellen, worden in deze sectie kort gereflecteerd.

Welke informatie is nodig om een goed algoritme te kunnen maken? Het in dit onderzoek ontworpen algoritme heeft de volgende informatie nodig:

1. Maximaal bechikbaar vermogen op moment van berekenen en de rest van de dag
2. Informatie van voertuigen die aangesloten staan, namelijk
   (a) De maximale stroom waarmee opgeladen kan worden
   (b) Totale accu-capaciteit
   (c) Initiële state of charge
   (d) Hoezeer de auto al is opgeladen sinds deze aangesloten is
   (e) Waarschijnlijke vertrektijd
3. Informatie van voertuigen die aan zullen komen
   (a) Totale accu-capaciteit
   (b) Verwachte initiële state of charge
   (c) Waarschijnlijke aankomsttijd
   (d) Waarschijnlijke vertrektijd

Hoe kan het algoritme het opladen van de auto's zo slim mogelijk indelen? In hoofdstuk 3 is in drie stappen een algoritme ontwikkeld dat vooral peak shaving, maar ook enigszins load-balancing kan uitvoeren. Het maximale vermogen wordt nu nooit overschreden. Als er niet genoeg vermogen is om alle auto's op te laden, wordt het beschikbare vermogen zo eerlijk mogelijk verdeeld aan de hand van resterende oplaadtijd per auto, benodigde energie per auto en of de auto volledig elektrisch is. Als er wel genoeg vermogen beschikbaar is wordt het opladen van de auto's zo veel mogelijk uitgesmeerd over de tijd, voor een zo gebalanceerde belasting door de dag heen.

Hoe kan het oplaadsysteem kennis krijgen van momentele energiebeschikbaarheid? Het huidige beschikbare vermogen wordt opgevraagd van het gebouwbeheersysteem (GBS) via de Compri HX.

Hoe kan een nauwkeurige schatting gemaakt worden van het toekomstig beschikbaar vermogen? De eerste schatting van het toekomstig beschikbaar vermogen wordt gedaan met behulp van een profiel van de verbruikte energie op een gemiddelde dag. Dit profiel wordt afgetrokken van het op voorhand vastgestelde plafond. Het resultaat is de eerste schatting van het toekomstig beschikbaar vermogen. Deze schatting wordt vervolgens verschoven langs de as van het vermogen tot deze snijdt met het huidige beschikbaar vermogen. Dit verkregen model zal dienen als de uiteindelijke schatting voor het toekomstige beschikbaar vermogen.

Welke informatie moet de eindgebruiker ingeven? In tabel 8.1 is weergegeven welke informatie nodig is van de eindgebruiker en waarom.

Tabel 8.1: Informatie nodig van eindgebruiker en reden

Informatie	Reden
Naam gebruiker	Herkenning
Kenteken	Herkenning
Type elektrisch voertuig	Herkenning
Capaciteit accu	Gebruikt voor het berekenen van het nog op te laden vermogen van de auto
Fases gebruikt bij opladen	Laadgedrag is verschillend
Maximale oplaadstroom	Indelen van stroom door algoritme
Volledig elektrisch	Prioriteit
State of Charge	Prioriteit
Aankomst- en vertrektijd	Prioriteit
Laadpunt	Herkenning
E-mail	Notificatie naar gebruiker

Hoe kan de interface het beste vormgegeven worden? Het is gebleken dat de uitwisseling van deze informatie via twee interfaces zal gaan. Via een applicatie en een web interface. De applicatie behandelt de dagelijkse aanmeldingen en de web interface de eenmalige registratie en de feedback naar de gebruiker toe.

Wat is de beste manier om informatie terug te koppelen naar de eindgebruiker? De feedback is tweeledig, namelijk feedback door een e-mail te sturen als de auto volledig opgeladen is en door het laadschema inzichtelijk te maken op de web interface.

Hoe kan de aanstuurinformatie en de feedback daarvan het beste gecommuniceerd worden? Om de informatie zo volledig mogelijk gecommuniceerd te krijgen is een protocol nodig. Om de implementatie van een protocol zo eenvoudig mogelijk te maken, kan het beste gebruik worden gemaakt van de hardware en software van Priva. Doordat dit zeer uitgebreide en ontwikkelde systeem tot onze beschikking was, is met weinig werk een zeer robuust communicatiekanaal tussen de server en de hardware opgezet.

Op welke manier kan er toch aansturing plaatsvinden, als communicatie wegvalt? Door het gebruik van het systeem van Priva kan de aansturing uitgebreid uitgerust worden. Zo beschikt de aansturing nu over een touchscreen waar het mogelijk is om de aansluitingen handmatig aan te sturen. Zo kan er zonder communicatie met het algoritme toch aansturing kan plaatsvinden.

Hoe kunnen alle onderdelen het beste verbonden worden? Door alle software te bundelen in één server, is het systeem een goed samenhangend geheel. De aansturing met de Compri HX ligt uiteraard buiten de server, maar is te bereiken via XML.

8.2. Reflectie programma van eisen

Er vindt een korte reflectie plaats op de eisen die niet gehaald zijn. Dit is er één, namelijk de implementatie van een wachtrij.

Wachtrij

Het idee was dat er via de dagelijkse aanmeldingen aangegeven kan worden dat alle laadpalen bezet zijn en dat de auto in de wachtrij komt te staan. Dit wordt dan opgeslagen in de database, waar het algoritme zijn data gaat ophalen. Dit wachtrijprincipe zou dan meegenomen worden in het genereren van een laadschema. Door een auto harder te laten opladen tot deze volledig opgeladen is, kan deze gewisseld worden met een auto in de wachtrij. Naar beide gebruikers moet dan een notificatie (e-mail) verstuurd worden dat deze auto's van plaats moeten wisselen. Deze implementatie is niet volbracht en wordt daarom aanbevolen voor verder onderzoek.

Het probleem van de wachtrij kan met de huidige implementatie al redelijk verholpen worden, maar nog niet optimaal. Het is nu zo dat iedere gebruiker een melding krijgt als zijn voertuig volledig opgeladen is, zij weten dus of ze hun auto kunnen wisselen met iemand in de wachtrij. Ook kan de gebruiker die in de wachtrij staat aan de hand van het laadschema die op de web interface staat controleren wiens auto klaar is met opladen en wie de persoon in de wachtrij dus moet contacteren.

8.3. Aanbevelingen voor verder onderzoek

Relatie state of charge en benodigde energie. In het ontwerp is een lineair verband aangenomen tussen de state of charge en de hoeveelheid opgeslagen energie in de auto. Bij het ingeven van de initiële state of charge door de gebruiker wordt dit lineair doorgerekend met de accucapaciteit om de totaal benodigde hoeveelheid energie te berekenen. Hoewel er in de bronnen niet duidelijk op ingegaan werd, is deze aanname wellicht verkeerd.

Verschil gemeten SOC en theoretisch voorspelde SOC. In combinatie met eventueel verloop in de meting van geleverde stroom, kan het voorkomen dat het systeem denkt dat een auto klaar is met opladen, terwijl dit niet zo is. Er kan dus nog onderzoek gedaan worden naar mogelijk verschil en voorspelde state of charge en werkelijke state of charge. Er is in dit onderzoek aangenomen dat hier geen verschil in komt, maar dit is niet zo in de praktijk. Er is enigszins feedback-regeling toegepast met de stroommeter vanuit de hardware, maar wellicht dat er nog een uitbreiding op het algoritme te ontwerpen is dat van tevoren rekening houdt met een rendement lager dan 100%.

Veilige modus Compri HX De Compri HX is een minicomputer met vele mogelijkheden. Momenteel wordt deze alleen gebruikt voor communicatie en eventuele handmatige aansturing. Omdat de informatie over het vermogen ook via de Compri HX gespeeld wordt, is het mogelijk om een klein peak shaving aloritme in de Compri HX te programmeren. Dat kan dan gebruikt worden in het geval dat communicatie met de server wegvalt.

Prioriteringswaardes vergelijken. In het onderzoek zijn de mogelijke prioriteringswaardes in het algoritme en combinaties hiervan niet heel uitgebreid onderzocht. Een kleine afweging is gedaan in tabel 3.4, maar uiteindelijk zijn alle mogelijke prioriteringswaardes meegenomen in de uiteindelijke prioriteitsbepaling. Er kan dus meer onderzoek gedaan worden naar de meest optimale prioritering.

Notificatie versturen. Deze worden verstuurd naar iedere gebruiker wiens auto volledig opgeladen is. Uit de interviews is gebleken dat gebruikers alleen een notificatie wensen te ontvangen als ze hun auto om moeten wisselen met iemand die in een wachtrij staat. Nadat een wachtrij implementatie onderzocht is, kan dit geïmplementeerd worden.

Bepaling toekomstig beschikbaar vermogen. De methode is beschreven in de reflectie op de desbetreffende onderzoeksvraag. In deze methode wordt er niet onderzocht of schaling na verschuiving ook wenselijk is. Het profiel dat gebruikt is, moet uitgebreider onderzocht worden (alle afdelingen van het gebouw meenemen in de berekening). Het profiel kan specifieker gemaakt worden in plaats van

alleen een gemiddelde te gebruiken voor iedere dag van het jaar. Enn voorbeeld is onderscheid tussen zomer en winter, in de winter gaat er bijvoorbeeld meer energie naar het verwarmen van het gebouw.

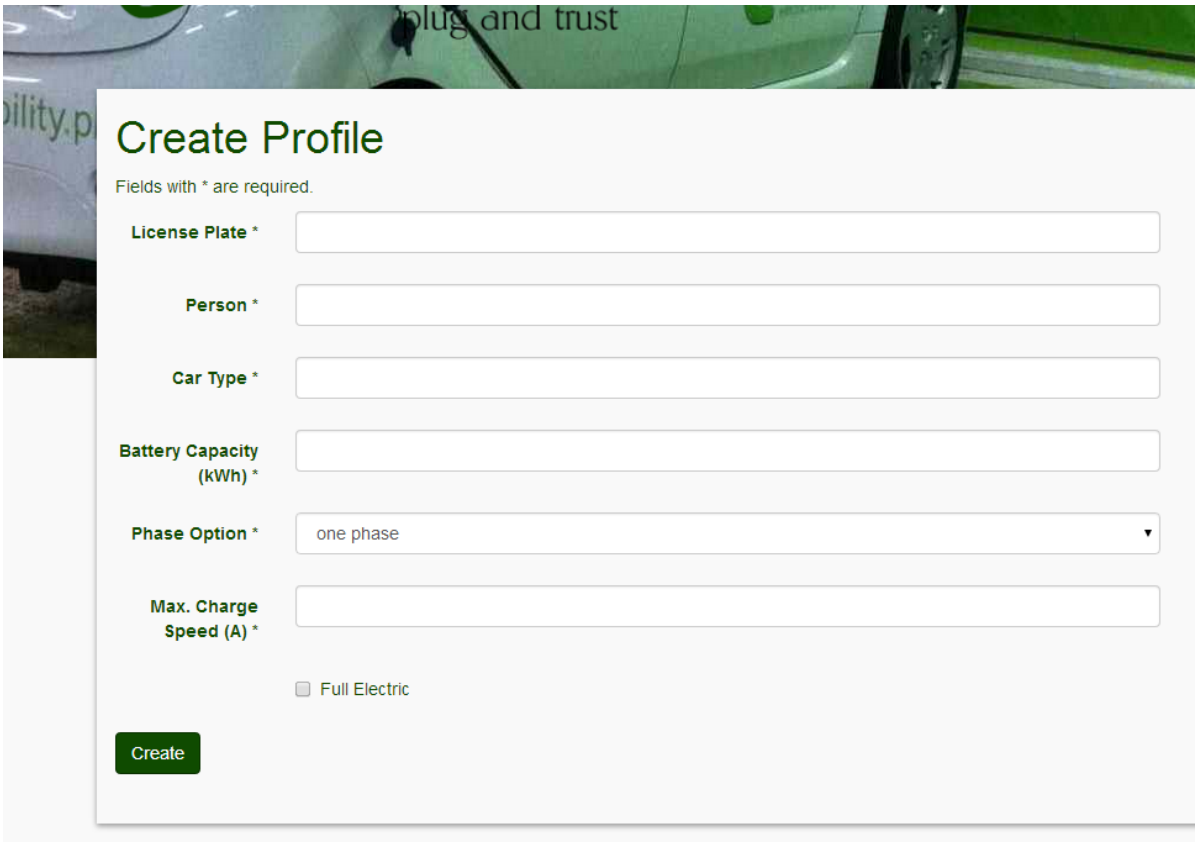
Deling van een fasedraad. Zoals in tabel 3.2 weergeven, is het afhankelijk van de (gezamenlijke) fasebelasting of één auto 32 ampère tot zijn beschikking heeft, of met een andere 32 ampère moet delen. In het tweede geval is nu de vereenvoudiging gemaakt dat de auto's ieder 16 ampère tot hun beschikking hebben. Er gaat nu eigenlijk capaciteit verloren op het moment dat één van de twee auto's minder dan 16 ampère oplaad, en de andere meer dan 16 ampère zou willen opladen. Hier zou nog iets voor ontwikkeld kunnen worden.

Interface aan laadpaal in plaats van een app. Dit is beter te veralgemenen als product. Dan is de dagelijkse aanmelding voor iedereen toegankelijk en niet alleen voor de gebruikers met een smartphone die een Android of iOS besturingsysteem hebben.

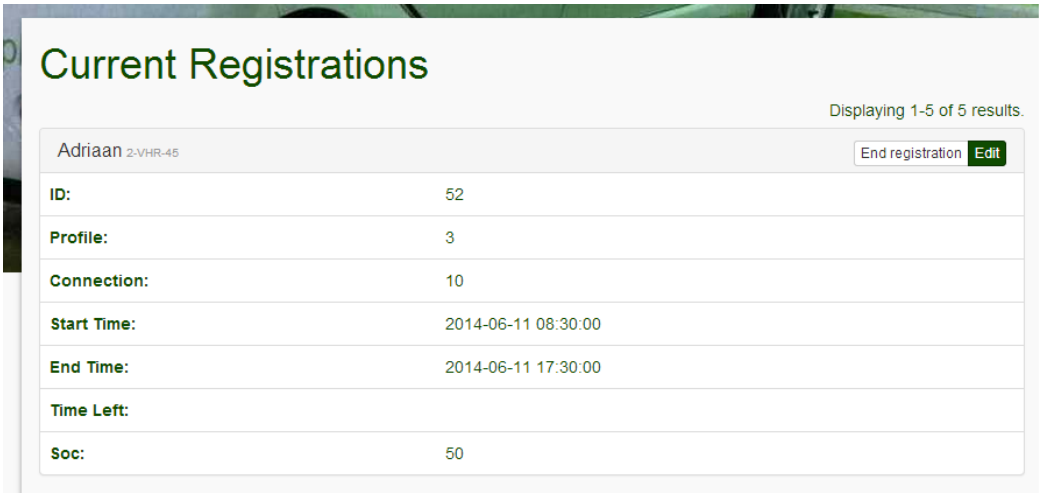
Artificial Neural Network (ANN). Dit is een zeer complex systeem en is kort toegelicht in de state-of-the-art (zie sectie 1.3.2). De mogelijkheid van de implementatie van een ANN als oplossing voor het bepalen van het toekomstig beschikbare vermogen is een heel onderzoek op zich.

A

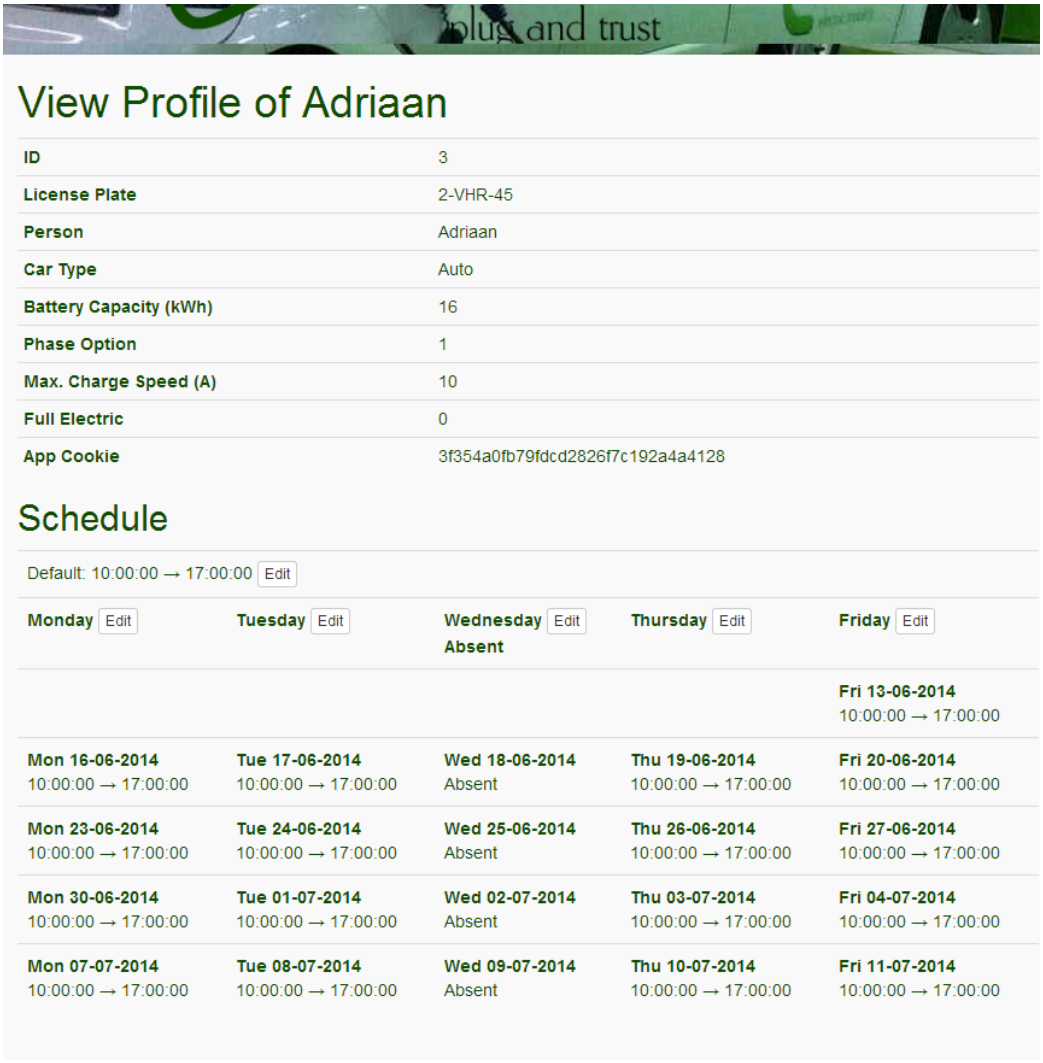
Screenshots web interface



Figuur A.1: Profiel aanmaken in de web interface



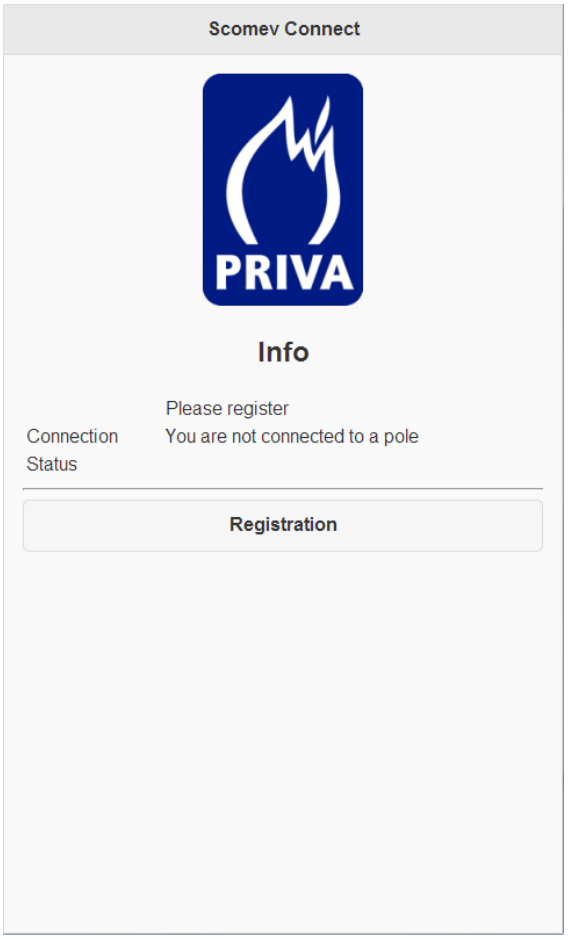
Figuur A.2: Dagelijkse aanmelding van applicatie via database doorgestuurd naar web interface



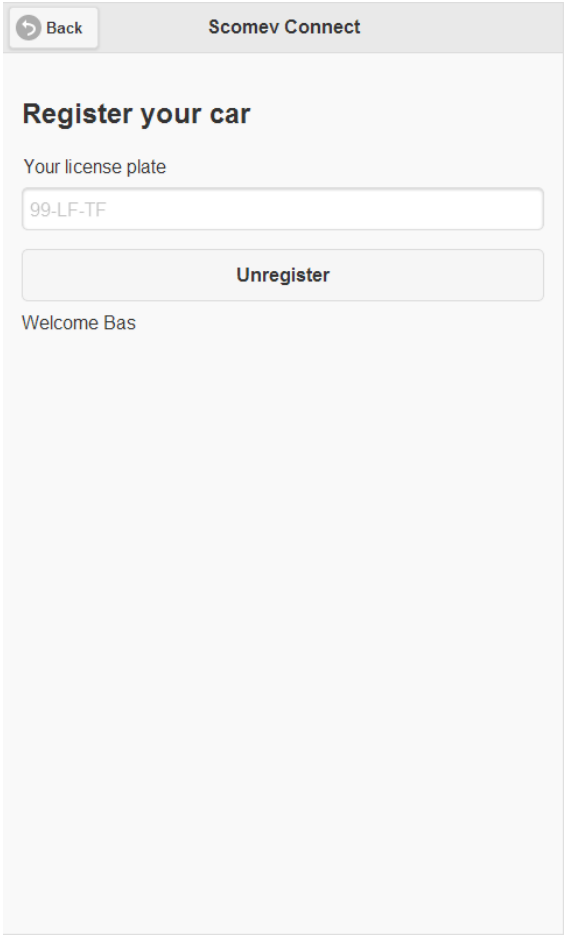
Figuur A.3: Profiel van gebruiker met bijbehorende agenda voor de aankomende maand

B

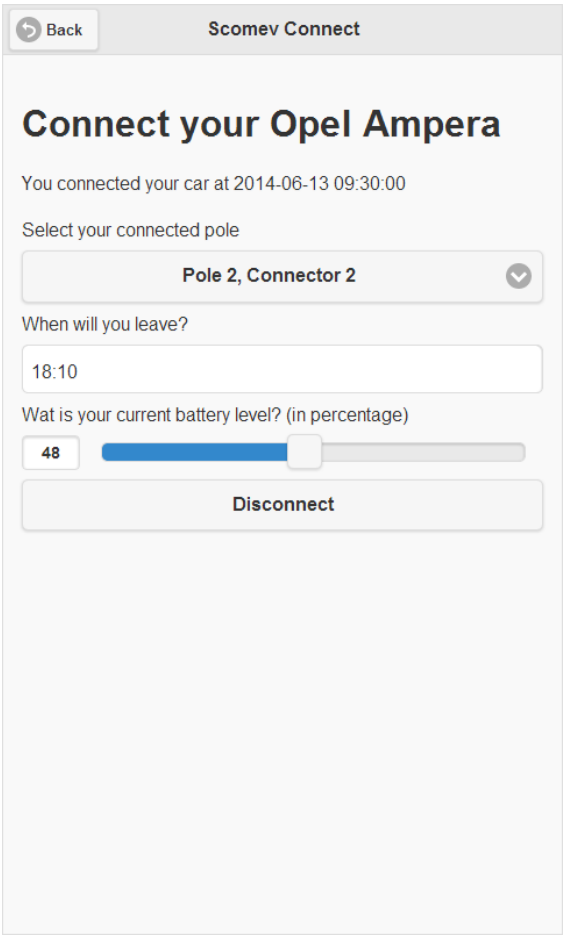
Screenshots applicatie



(a) Startscherm dagelijkse aanmelding via app



(b) Herkenning van de gebruiker



Figuur B.2: Verbinden met een laadpunt

C

Interviews

Ronald Zeelen

Huidige situatie Niet gebruiksvriendelijk. Er moet heel veel heen en weer gemaild worden over de beschikbaarheid van de laadpalen. Nu is iedereen verantwoordelijk voor elkaar. Je moet zelf weten wanneer je je auto ongeveer weg moet halen.

Interface Een interface aan de laadpaal vindt hij overbodig. Het zou wel interessant zijn om overdag te weten hoe ver je auto al opgeladen is.
Via de interface moet een bepaalde urgentie opgegeven kunnen worden, de invulling daarvan moet goed beoordeeld worden.
Aanmelden via een applicatie kan, alleen het aantal zaken dat ingevuld moeten worden beperkt houden.

Feedback Notificatie ontvangen liever via sms dan e-mail en op het internet weergeven of een andere auto al bijna opgeladen is. Een notificatie naar iemand sturen die in de wachtrij staat, is wenselijk.

Risico Hij ziet een mogelijk risico bij het eerlijk invullen van de applicatie.

Maarten Kuiper

Huidige situatie De palen zijn niet erg betrouwbaar, veel storingen. Daarnaast is er ook niet altijd plek.

Interface Vindt een applicatie invullen minder wenselijk als het weer extra tijd kost. Een agenda invullen voor heel de aankomende week is ook een optie.
Een prioriteitenschema is wenselijk, aan de hand van state of charge, een FIFO systeem.

Feedback Notificatie sturen naar iemand die in de wachtrij staat als een auto klaar is met opladen, dit kan zowel via e-mail als sms.

Risico Mogelijk risico bij het eerlijk invullen van de applicatie.

Jan van der Hoeven

Huidige situatie De huidige situatie is lastig. Er zijn afspraken gemaakt dat er tijdens de lunch gewisseld wordt, maar daar houdt iedereen zich niet aan.

Interface Zou de interface liever aan de laadpaal zien en denkt dat de app weer een extra barrière teweegbrengt, liever eenmalig aanmelden.

Feedback Notificatie kan via de mail of sms, toch liever een e-mail.
Wil in de dashboard op internet alleen de status van zijn eigen auto zien.
Met een niet volle accu naar huis rijden is een optie.

Risico Het grote risico is dat mensen geen zin hebben in een applicatie.

Bibliografie

[1] A. Taal and S. G. van Wee, *Smart Charging of Multiple Electrical Vehicles, De Hardware*, Tech. Rep. (Delft University of Technology, 2014).

[2] A. S. Masoum, S. Deilami, P. S. Moses, M. A. S. Masoum, and A.-S. A, *Smart load management of plug-in electric vehicles in distribution and residential networks with charging stations for peak shaving and loss minimisation considering voltage regulation,* IET Generation Transmission & Distribution **5**, 877 (2011).

[3] K. Nandha Kumar, P. Cheah, and B. Sivaneasan, *Electric Vehicle Charging Profile Prediction for Efficient Energy Management in Buildings*, Tech. Rep. (Nanyang Technological University, 2012).

[4] A. Rodriguez-Serrano, A. Torralba, E. Rodriguez-Valencia, and J. Tarifa-Galisteo, *A communication system from EV to EV Service Provider based on OCPP over a wireless network*, Tech. Rep. (Department of Electronic Engineering University of Seville, 2013).

[5] R. voor Ondernemend Nederland, *Aantal geregistreerde elektrische voertuigen in nederland,* (2014).

[6] P. Borkowski, M. Pawlowski, and T. Makowiecki, *Economical aspects of building management systems implementation,* in *PowerTech* (IEEE, Trondheim, 2011) pp. 1–6.

[7] W. Yang, C. Zai, and X. Huang, *Study on the influence of large-scale electric vehicles on power grid,* Advanced Materials Research **608-609**, 1566 (2013).

[8] N. Leemput, F. Geth, B. Claessens, J. Van Roy, R. Ponnette, and J. Driesen, *A case study of coordinated electric vehicle charging for peak shaving on a low voltage grid,* in *Innovative Smart Grid Technologies* (IEEE, 2012) pp. 1–7.

[9] D. Molina, C. Hubbard, C. Lu, R. Turner, and R. Harley, *Optimal ev charge-discharge schedule in smart residential buildings,* in *PowerAfrica* (IEEE, 2012) pp. 1–8.

[10] Z. Wang and W. S, *Grid power peak shaving and valley filling using vehicle-to-grid systems,* IEEE transactions on power delivery **28**, 1822 (2013).

[11] F. Kennel, D. Görges, and S. Liu, *Energy management for smart grids with electric vehicles based on hierarchical mpc,* IEEE transactions on industrial informatics **9**, 1528 (2013).

[12] B. University, *Is lithium-ion the ideal battery?* (2010).

[13] M. Mägi, *Utilization of Electric Vehicles Connected to Distribution Substations for Peak Shaving of Utility Network Loads*, Tech. Rep. (Tallinn University of Technology, 2013).

[14] BMW, *Bmw i remote app,* (2014).

[15] Mitsubishi, *Mitsubishi remote control app,* (2014).

[16] Nissan, *Nissan carwings app,* (2014).

[17] Opel, *Opel myampera app,* (2014).

[18] Chevrolet, *Chevrolet volt driver's challenge app,* (2014).

[19] Volvo, *Volvo on call app,* (2014).

[20] The New Motion, *The new motion app,* (2014).

[21] Oplaadpalen.nl, *Oplaadpalen app,* (2014).

[22] Z. Yuan, H. Xu, H. Han, and Y. Zhao, *Research of smart charging management system for electric vehicles based on wireless communication networks,* in *Information and Automation for Sustainability* (IEEE, Beijing, 2012) pp. 242–247.

[23] B & B Electronics, *The obd ii home page,* (2013).

[24] B. Oliver, *Communications protocols to promote robust grid-ev interoperability,* (2012).

[25] H. Appelrath, C. Weinhardt, and O. Terzidis, *Internet of energy,* Business & Information Systems Engineering **4**, 1 (2012).

[26] J. Fluhr and T. Lutz, *Communication with and for electric vehicles, electric vehicles "the benefits and barriers, dr. seref soylu (ed.)", chapter 9,* (2011).

[27] Z. Whittaker, *Android, ios score 96 percent of smartphone share in q4 rankings,* (2014).

[28] Phonegap, *http://phonegap.com/,* (2014).

[29] R. Almgren, *Mysql database connector from matlab file exchange,* (2005).

[30] Yii Framework, *http://www.yiiframework.com/,* (2013).

[31] Google Charts, *https://developers.google.com/chart/,* (2013).

[32] jQuery, *http://jquery.com/,* (2014).