Ebenen Modell

Beschreibung und Definition der drei Ebenen (Statisch, Flexibilität, Netz), die Grundlage der Architektur von Casa Corrently sind. Die Nutzung dieses Ebenenmodells hilft die einzelnen Komponente (Zähler, Sensoren, Prozesse) einfacher zu strukturieren. Jede Ebene ist untergliedert in die Kernelemente: Energiebilanz, Vorhersage und GrünstromIndex (der Ebene)

Einleitung

ebenen_in_nodered.png
Image not found or type unknown

Innerhalb von Node-Red sind die ebenen als einzelne Flows angelegt und fassen Schritte im Energiemanagement zusammen. Jede Ebene bietet die Möglichkeit eine eigene Erzeugung und Verbrauch zu definieren, welche zur Bildung einer Bilanz genutzt wird.

Flow Daten einer Ebene

ebene_0_flowdaten.png
Image not found or type unknown

Die einzelnen Komponenten (zum Beispiel Stromzähler) fügen ihren individuellen Wert in Watt zur entsprechenden Variable des Flows hinzu.

wirkleistungs_knoten.png
Image not found or type unknown

```
// msg.payload =  msg.payload;
const subSUM='Consumption';



let sum = flow.get(subSUM) * 1;
if(isNaN(sum)) sum = 0;

let previous = context.get("previous") * 1;
if(isNaN(previous)) previous = 0;

if(flow.get("SaldoID") != context.get("SaldoID")) {
    context.set("SaldoID",flow.get("SaldoID"));
} else {
    sum -= previous;
}
sum += msg.payload;

context.set("previous",msg.payload);

flow.set(subSUM,sum);
node.status({text:msg.payload +" W"});
return msg;
```

Damit auch bei asynchronem Empfang der Daten immer eine Bilanz gebildet werden kann, wird mit einer `SaldoId` gearbeitet, welche als Indikator für die einzelnen Komponenten dient, ob der Leistungswert bereits zur Summe (`Consumption` oder `Production`) hinzugefügt wurde.

In der Referenzimplementierung merkt sich der Node im eigenen Context, welche Leistung zur Summe hinzugefügt wurde (Variable `previous`) sowie die `SaldoId`. Sollte beim folgenden Messwert die `SaldoId` gleich sein, so wird der vorherigen Messung zunächst von der Summe abgezogen (Zeile 14) und im Anschluss der aktuelle Wert hinzugefügt (Zeile 16).

Konzepte

Die Konzepte im Ebenenmodell dienen als Design-Pattern und Nomenklatur. Sie geben Beispiel für die Implementierung und Abbildung auf eine individuelle Problemstellung

SaldoId

Eines der Ziele jeder Ebene ist die Bildung einer Energiebilanz. Damit auch asynchron mit Sensordaten gearbeitet werden kann, wird eine SaldoId verwendet.

saldo_id_trigger.png
Image not found or type unknown

Auf jeder Ebene wird die SaldoId mit Hilfe eines *Inject* einmal zum Start des Flows und dann in regelmäßigen Intervallen neu definiert und als Flow Variable den Komponenten der jeweiligen Ebene (Flow) zur Verfügung gestellt. Gleichzeitig werden die Summen des Flows zurückgesetzt.

```
flow.set("Consumption",0);
flow.set("Generation",0);
flow.set("SaldoID",new Date().toLocaleString());
return msg;
```

Hinweis

Die SadoId sollte nicht zu häufig aktualisiert werden, damit bei der Bildung der Bilanz für die Ebene alle Messwerte berücksichtigt werden.

Bilanzierung einer Ebene (Saldo)

Jede Ebene bildet für sich eine Bilanz (Saldo). Hierbei wird in der Einheit Watt gerechnet (nicht WattStunde!), da die Sensoren so oft wie möglich einen Messwert liefern sollen.

saldo_ebene0.png
Image not found or type unknown

```
msg.payload = flow.get('Consumption') - flow.get('Generation');
flow.set('Saldo',msg.payload);
node.status({text:"P: "+msg.payload+" W"});
return msg;
```

Konzepte

Zweistufige Prognoseerstellung

Die Vorhersagen in den Ebenen werden in zwei Schritten durchgeführt. Zunächst wird eine generische Abbildung auf Abhängige Werte vorgenommen und im Anschluss in einen Graphen (=konkrete Zeitreihe) überführt.

Schritt 1: Abbildung

vorhersage_schritt1.png
Image not found or type unknown

```
// Create empty array if not exists
let hours = {};
for(let i=0;i<24;i++) {
    hours["h"+i] = {
        latest: {},
        outcome: {
            sum: 0,
            cnt: 0,
            mean: 0,
            min: 9999999,
            max:0,
            min_gsi:999999
        }
    };
}

for(let i=0;i<msg.payload.length;i++) {
   let hour = new Date(msg.payload[i].time).getHours();
   hours["h"+hour].latest.gsi = msg.payload[i].mean + 1;
}

msg.hours = hours;
return msg;
```

In diesem Schritt wird ein Array mit einem Element für jede Stunde des Tages gebildet (Zeile 3-15). Auf Basis der Werte aus der Vergangenheit wird nun ein Wert für jede Stunde prognostiziert.

Ist die Vorhersage von einem anderen Wert als die Tageszeit abhängig, wie zum Beispiel bei der Sonneneinstrahlung, so wird im Anschluss die vorhandenen Bedingungen (Beispiel: Sonneneinstrahlung) zum jeweiligen historischen Wert gemappt.

Schritt 2: Graphen

vorhersage_photovoltaik.png
Image not found or type unknown

```
let a = [];

for(let i=0;i<msg.payload.gsi.forecast.length;i++) {
    let item = {
        'x':msg.payload.gsi.forecast[i].timeStamp,
        'y':msg.payload.gsi.forecast[i].esolar
    }
    a.push(item);
}

msg.graph = [{
    'series': ['GenerationForecast'],
    'data': [a]
}];
return msg;
```

Der Graphen bildet die Zeitstempel für die kommenden 36 Stunden (=Vorhersagezeitraum) ab. Er wird bereits so aufgebaut, dass er am Ende der Verarbeitung in der InfluxDB gespeichert und mittels der Grafana Dashboards visualisiert werden kann.

Konzepte

Ebenen GrünstromIndex

ebenen gsi.png


Jede Ebene berechnet einen eigenen GrünstromIndex. Dieser liegt im Wertebereich zwischen 0 und 100, wobei ein hoher Indexwert ein Indikator für viel Strom und ein niedriger Indexwert einer für wenig Strom ist. Der GrünstromIndex wird in der regel für die kommenden 36 Stunden berechnet.

gsi_ebenen4.png


In den Ebenen 0 und 1 wird der Wert auf Basis der Prognosen erstellt. In der Ebene 3 wird der für die Postleitzahl des Objektes gültige öffentliche GrünstromIndex verwendet.

```
let saldo = flow.get("forecastSaldo");
let min = 9999999999;
let max = -999999999999;

for(let i=0; i<saldo.length;i++) {
    saldo[i].measurement = "e0gsi";
    if(saldo[i].fields.w > max) max = saldo[i].fields.w ;
    if(saldo[i].fields.w < min) min = saldo[i].fields.w ;
}

let delta = max - min;

for(let i=0; i<saldo.length;i++) {
    saldo[i].fields.gsi = 100-Math.round(((saldo[i].fields.w  - min) / delta)*100);
}



flow.set("gsi",saldo);
global.set("e0gsi",saldo);
msg.payload = saldo;

return msg;
```

Ebene 0 - StatusQuo

In dieser Ebene werden die meisten Sensoren und Stromzähler eingebunden. Sie enthält alle Komponenten, welche sich nicht direkt beeinflussbar sind.

Erzeugung und Verbrauch

ebene0_erzeugung_verbrauch.png
Image not found or type unknown

Erzeugung

Bei der Referenzimplementierung existieren zwei Quellen zur Stromerzeugung.

Eine **Aufdach-PV-Anlage**, welche bereits initial zur Eigenstromnutzung installiert wurde. Diese Anlage besitzt folgende Elemente:

- SMA Wechselrichter (Tripower)
- SMA Energy Meter für Wirkleistungsbegrenzung
- SMA HomeManager
- Discovergy Produktionszähler (Easymeter via Discovergy API)

Zur Messung der erzeugten Strommengen wird lediglich der Discovergy Zähler genutzt in der Referenzimplementierung genutzt, da die SMA Komponenten insgesamt zu ungenaue Werte lieferten. Die Werte des SMA Energy Meter lassen sich jedoch recht einfach in Node-Red über einen `UDP In` Node integrieren.

Funktion Produktionszähler

```
msg.payload = msg.payload["303fbb8ca6404ebba48c196b4dbbc176"];
return msg;
```

Funktion Wirkleistung

```
msg.payload = Math.round((msg.payload.power1 + msg.payload.power2 + msg.payload.power3)/1000)
const subSUM='Generation';

let sum = flow.get(subSUM) * 1;
if(isNaN(sum)) sum = 0;

let previous = context.get("previous") * 1;
if(isNaN(previous)) previous = 0;

if(flow.get("SaldoID") != context.get("SaldoID")) {
    context.set("SaldoID",flow.get("SaldoID"));
```

```
} else {
    sum -= previous;
}
sum += msg.payload;

context.set("previous",msg.payload);
node.status({text:msg.payload +" W"});
flow.set(subSUM,sum);
return msg;
```

Die Discovergy API liefert die Leistungswerte je Phase. Da es sich um einen 3-Phasen Wechselrichter handelt, muss aus den Einzelwerten die Summe gebildet werden (Zeile 1). Zum Schluss wird der rohe Wert der Discovergy API noch auf die innerhalb von Casa Corrently verarbeite Dimension (Watt) umgerechnet.

Als Ergänzung existiert eine kleine **Balkon-PV Anlage**. Angeschlossen ist diese Anlage mittels eines auf ZigBee auslesbaren Wechselrichters, da jedoch der Hersteller insolvent ist, können die Daten nicht direkt vom Wechselrichter ausgelesen werden. Stattdessen wird eine für diesen Betrieb modifizierte ein AVM FRITZ!DECT 200 Smart Home Stecker verwendet, welcher über die Fritz-Box API ausgelesen werden kann.

Funktion BalkonPV

```
msg.payload.ain = "087610221618"; // Hier die AIN aus der Fritz Box eintragen
return msg;
```

Funktion Wirkleistung

```
msg.payload = Math.round(msg.payload);
const subSUM='Generation';

let sum = flow.get(subSUM) * 1;
if(isNaN(sum)) sum = 0;

let previous = context.get("previous") * 1;
if(isNaN(previous)) previous = 0;

if(flow.get("SaldoID") != context.get("SaldoID")) {
    context.set("SaldoID",flow.get("SaldoID"));
} else {
    sum -= previous;
}
```

```
sum += msg.payload;

context.set("previous",msg.payload);

flow.set(subSUM,sum);
node.status({text:msg.payload +" W"});
return msg;
```

Verbrauch

Die Verbrauchsmessung für den gesamten Haushalt muss den in Ebene 1 definierten Speicher herausrechnen. Dies geschieht bei der Refeenzimplementierung durch die Betrachtung Ermittlung des Strombezugs am Netzanschluss (Ebene 2) sowie die vie MQTT verfügbare Entladeleistung des Speichers.

Funktion Wirkleistung

```
msg.payload = msg.payload["781ffa307e434529be9f747eece1b8dc"];

let power = Math.round((msg.payload.power1 + msg.payload.power2 + msg.payload.power3)/1000)

msg.payload = power;

return msg;
```

Trigger

Lediglich die Abfrage der Discovergy-API benötigt einen Trigger (Auslöser), damit regelmäßig neue Leistungsdaten vorhanden sind. Der Speicher sowie die Balkon-PV Installation benötigen keinen Trigger, da hier von den Sensoren fortlaufend Messwerte gesendet werden.

Ausgang zum Saldo / Bilanzierung

linkout_ebene0.png
Image not found or type unknown

Damit bei jeglicher Aktualisierung der Saldo ebenfalls aktualisiert wird, kommt ein "Link Out" Node zum Einsatz

Bilanzierung / Saldo

ebene0_saldo.png

In der Ebene 0 wird der Saldo über die im Flow gespeicherten Summen der Erzeugung (Generation) und des Verbrauchs (Consumption) gebildet.

Funktion Saldo bilden

```
msg.payload = flow.get('Consumption') - flow.get('Generation');
flow.set('Saldo',msg.payload);
node.status({text:"P: "+msg.payload+" W"});
return msg;
```

Die erneute Aufteilung der Einzelwerte dient lediglich der getrennten Speicherung in der InfluxDB.

Vorhersage / Prognose

ebene0_vorhersage.png

Wie in der Konzeptbeschreibung zur Prognose aufgeführt, erfolgt die Erstellung der Prognose in zwei Schritten. Zur Vorhersage der Erzeugung wird in der Referenzimplementierung der `esolar` Wert des GrünstromIndex genutzt und mit den tatsächlichen Erträgen der Anlage präzisiert.

ebene0_vorhersage_grafana.png

Für die Aktualisierung der Prognose wurde ein Intervall von 20 Minuten gewählt, da gerade für die Erzeugung lediglich alle 60 Minuten neue Werte aus dem GrünstromIndex vorliegen und somit eine häufigere Aktualisierung keine besseren Ergebnisse liefert.

Ebene 0 - StatusQuo

GrünstromIndex

ebene0_gsi.png
Image not found or type unknown

Basierend auf dem prognostizierten Saldo der Ebene wird ein GrünstromIndex gebildet und zur Visualisierung mittels Grafana in die InfluxDB gespeichert.

Vergleiche: GrünstromIndex je Ebene

Funktion Ebenen GSI

```
let saldo = flow.get("forecastSaldo");
let min = 9999999999;
let max = -999999999999;

for(let i=0; i<saldo.length;i++) {
    saldo[i].measurement = "e0gsi";
    if(saldo[i].fields.w > max) max = saldo[i].fields.w ;
    if(saldo[i].fields.w < min) min = saldo[i].fields.w ;
}

let delta = max - min;

for(let i=0; i<saldo.length;i++) {
    saldo[i].fields.gsi = 100-Math.round(((saldo[i].fields.w  - min) / delta)*100);
}



flow.set("gsi",saldo);
global.set("e0gsi",saldo);
msg.payload = saldo;

return msg;
```

ebene0_gsi_prognose.png
Image not found or type unknown

Ebene 1 - Flexibilität und Speicher

In dieser Ebene werden alle Komponenten eingebunden, die für das Energiesystem eine Flexibilität darstellen. Dies können entweder verschiebbare Lasten oder ein vorhandener Stromspeicher sein.

Erzeugung und Verbrauch

Erzeugung

Bei der Referenzimplementierung von Casa Corrently existiert kein flexibler Erzeuger. Die Erzeugung ist ausschließlich Photovoltaik und somit vom Sonnenschein abhängig. An dieser Stelle können jedoch flexible Erzeuger wie ein BHK aufgenommen werden, dazu ist analog zur Einbindung der Erzeugung in Ebene 0 vorzugehen.

Die Entladung des Speichers wird, wegen des gewählten Betriebsmodus des Speichers, erst im Saldo / Bilanzierung berücksichtigt.

Funktion Wirkleistung

```
msg.payload = Math.round(msg.payload.power_mw / 1000);
node.status({text:"P:  "+msg.payload+"  W"});
const subSUM='Consumption';

let sum = flow.get(subSUM) * 1;
if(isNaN(sum)) sum = 0;

let previous = context.get("previous") * 1;
if(isNaN(previous)) previous = 0;

if(flow.get("SaldoID") != context.get("SaldoID")) {
    context.set("SaldoID",flow.get("SaldoID"));
} else {
    sum -= previous;
}
sum += msg.payload;

context.set("previous",msg.payload);
flow.set(subSUM,sum);

return msg;
```

Verbrauch

Flexible Verbräuche sind Geräte, bei denen entweder der Betriebszustand geschaltet werden kann (An/Aus) oder die Wirkleistung (Watt) begrenzt wird.

Innerhalb der Referenzimpelementierung wird hier ein TP-Link HS110 Smart-Plug als Beispiel für ein schaltbares Gerät verwendet. An der so geschalteten Steckdose hängt ein E-Scooter, der lediglich mit überschüssigem Strom geladen werden soll. Mit `Flexon` wird ein Inbound-Webhook definiert, der den SmartPlug schalten kann. Ausgelöst (gesendet) wird der Webhook hier von einem Alert im Grafana Dashboard. Selbstverständlich hätte man dieses Ziel auch direkt aus den Werten der Ebene 2 erreichen können oder zusätzliche Bedingungen/Priorisierungen mit einer anderen Flexibilität vornehmen können.

Die **Wallbox** ,zur Steuerung der KFZ-Ladung, hat ein eingebautes Management für die Ladung mit Überschussstrom. Zur Saldenbildung wird von der Ladeeinrichtung daher lediglich die momentane Leistung via MQTT abgerufen.

Bilanzierung / Saldo

ebene1_saldo.png
Image not found or type unknown

Bei der Referenzimplementierung kommt ein Speicher zum Einsatz, welcher im Betriebsmodus "*Null-Auspegelung am Netzanschlusspunkt*" betrieben wird. Dieser Modus hat den Vorteil, dass die Eigenoptimierung des Speichers vollständig erhalten bleibt und als abgeschlossenes System betrachtet werden kann.

Eine Saldierung muss damit nicht durch die Ebene vorgenommen werden, sondern kann aus den Speicherdaten ausgelesen werden.

Funktion SoC kWh Umrechnen

```
msg.payload = Math.round(5500 * (msg.payload/100));
flow.set('Saldo',msg.payload);
node.status({text:"P: "+msg.payload+" W"});
return msg;
```

Die Flexibilitäten werden als Kapazität (regelbares/steuerbare Stromnutzung) als `e1flex` gespeichert. Hierbei ist zu beachten, dass der Speicher maximal mit 2.000 Watt beladen oder entladen werden kann.

Funktion Kapazität

```
if(msg.payload > 2000) msg.payload = 2000;
msg.payload += flow.get('Consumption') + global.get('FlexConsumption');
node.status({text:"P: "+msg.payload+" W"});
return msg;
```

Vorhersage / Prognose

ebene1_vorhersage.png

Die Vorhersage in Ebene 1 erfolgt durch Fortschreibung der Prognose in Ebene 0. Basierend auf den Ebene 0 Werten, wird der Speicherfüllstand vorhergesagt und die flexiblen Lasten ergänzt.

Funktion Vorhersage SoC und Flex Integration

```
let saldo = flow.get("Saldo") - flow.get("Consumption");
let options = global.get("FlexOptions");

for(let i=0; i<msg.payload.length;i++) {
    msg.payload[i].measurement = "e1forecastSaldo";
    let w = (-1) * msg.payload[i].fields.w;
    let flex = 0;

    if(i<24) {
        for(let j=0;j<options.length;j++) {
             w -= options[j][i];
             flex += options[j][i];
        }
    }



    if(w > 2000) w = 2000;
    if(w < -2000) w = -2000;

    if(saldo + w > 5500) w = 5500-saldo;
    if(saldo + w < 0) w = saldo;

    saldo += w;

    msg.payload[i].fields.w -= w;
    msg.payload[i].fields.flex = flex;
    msg.payload[i].fields.wh = saldo;
```

```
}


flow.set("forecastSaldo",msg.payload);
node.status({text:""+new Date().toLocaleString()});
return msg;
```

Optionen für Flexibilitäten werden im globalen Array `FlexOptions` gehalten (Zeile 2). Dort gespeichert werden sie durch das individuelle Fahrplanmanagement zum Gerät. Diese einzelnen Vorhersagen werden vor der Berücksichtigung des Speichers integriert (Zeile 9-14).

Eine Besonderheit hier ist, dass der Speicher der Referenzimplementierung maximal mit 2.000 beladen und entladen werden kann. Entsprechend kann der Speicherfüllstand innerhalb von einer Stunde maximal um +/- 2 kWh verändert werden (Zeile 17-18).

GrünstromIndex

ebene1_gsi.png
Image not found or type unknown

Basierend auf dem prognostizierten Saldo der Ebene wird ein GrünstromIndex gebildet und zur Visualisierung mittels Grafana in die InfluxDB gespeichert.

Vergleiche: GrünstromIndex je Ebene

Funktion Ebenen GSI

```
let saldo = flow.get("forecastSaldo");
let min = 9999999999;
let max = -999999999999;

for(let i=0; i<saldo.length;i++) {
    saldo[i].measurement = "e1gsi";
    if(saldo[i].fields.w > max) max = saldo[i].fields.w ;
    if(saldo[i].fields.w < min) min = saldo[i].fields.w ;
}

let delta = max - min;

for(let i=0; i<saldo.length;i++) {
    saldo[i].fields.gsi = 100-Math.round(((saldo[i].fields.w  - min) / delta)*100);
}



flow.set("gsi",saldo);
msg.payload = saldo;
global.set("e1gsi",saldo);
return msg;
```

ebene1_gsi_grafana.png
Image not found or type unknown

Je nach tatsächlichen Vorhandenen Flexibililtäten unterscheidet sich der GrünstromIndex der Ebene 0 und der Ebene 1 nur minimal. Wird, wie bei der Referenzimplementierung, kommt bei der Erzeugung lediglich Photovoltaik zum Einsatz, dann existieren gerade in den Wintermonaten nur

wenige Abweichungen.

Ebene 2 - Netzanschlusspunkt

In dieser Ebene befindet sich die Übergabe zum öffentlichen Stromnetz oder dem Stromanbieter.

Bilanzierung / Saldo

ebene2_saldo.png
Image not found or type unknown

Im Szenario der Referenzimplementierung ist es das Ziel die Abweichungen des Saldo auf Ebene 2 in den positiven (Einspeisung) oder negativen (Bezug) Bereich möglichst gering zu halten.

In dieser Ebene finden keine Regel oder Steuerungen statt. Entsprechend ist der Saldo der Wert des Netzübergabezählers.