volkszaehler.org

the open smartmetering platform

volkszaehler.org community, Steffen Vogel

Zusammenfassung

volkszaehler.org ist ein freier Smart Meter im Selbstbau. Die anfallenden Stromprofile bleiben dabei unter der Kontrolle des Nutzers.

Seit dem 1.1.2010 müssen Stromversorger ihren Kunden für Neubauten so genannte "intelligente Stromzähler" (Smart Meter) anbieten. Der Kunde soll dadurch seinen Stromverbrauch analysieren und optimieren können.

Das dabei anfallende Stromverbrauchsprofil erlaubt einen sehr detallierten Einblick in den Tagesablauf des Nutzers (wann steht er auf? wann geht er in's Bett? wann kocht er? wie oft verwendet er seine Spülmaschine? verändert sich sein Verhalten? ...).

Darum sollten diese Daten ausschließlich für den Nutzer selbst zur Verfügung stehen - und das geht nur, wenn man sich den Smart Meter selbst baut. Mit einem Materialeinsatz von ca. 100 €, etwas Geschick und Zeit lässt sich das mit Hilfe eines Standard-µC-Moduls aufbauen.

Inhaltsverzeichnis

1 Einführung

In diesem Artikel werden wir das Projekt vorstellen, einen generellen Überblick über die einzellnen Kompontenten geben und auf die Implementierung eingehen. Ich möchte damit den Einstieg in das Projekt erleichtern und neue Entwickler motivieren sich zu beteiligen.

Einzellheiten zur Installation/Konfiguration eines eigenen volkszaehler.org Setups finden Sie unsere Mailingliste[1] und im Wiki[2].

Neben dem **Schutz der Privatsphäre**, hat das Projekt auch noch einige andere Ziele:

- eine kostenfreie Alternative gegenüber den kommerziellen Messstellenbetreibern anbieten
- dem Nutzer ein Bewusstsein über seinen Verbrauch/Nutzungsverhalten aufzeigen
- Energie intelligenter zu nutzen
- die Breite Masse erreichen[3]
- offene Protokolle und Standards zu vorrantreiben

Wir haben es uns zur Aufgabe gemacht diese Ziele durch eine lückenlose Kette von quelloffener Soft- und Hardware zu erreichen. Der Nutzer soll die Möglichkeit haben jeden einzellnen Schritt nachvollziehen zu können. Daraus folgt nicht, dass wir jede Zeile Code selbst geschrieben haben. Wir nutzen eine Reihe anderer Software, die aber wiederum selbst quelloffen ist.

2 Was ist ein intelligenter Zähler, Smart Meter oder elektronischer Heimzähler

Ein intelligenter Zähler ist ein elektronischer Strom-, Gas-, Wasser- oder Wärmemengenzähler. Er soll dem Nutzer seinen Energieverbrauch transparent machen. Durch eine zeitnahe Rückmeldung soll ein sparsamer Umgang mit Energie gefördert werden. Detaillierte Lastprofile erlauben dem Verbraucher eine gezielte Optimierung seines Energieverbrauchs. Sein Lastprofil gibt ihm Antworten auf die folgenden Fragen:

- Wie hoch ist die eigene Grundlast?
- Lohnt sich (aus ökonomischer oder ökologischer Sicht) der Ersatz eines Altgerätes durch ein effizienteres Neugerät?

[1] volkszaehler-dev@lists.volkszaehler.org
[2] http://wiki.volkszaehler.org
[3] Daher stammt auch der Name **volks**zaehler.org

- Wie hoch ist der Anteil von Geräten mit hoher Leistung, die nur für kurze Zeit eingesetzt werden (bei Stromverbrauchern z.B. Fön, Staubsauger, Mikrowelle), am Gesamtverbrauch?
- Gibt es hier Optimierungspotenzial?
- Lassen sich Lastspitzen in Zeiten günstigerer, tageszeitabhängiger Tarife verschieben?

Dem Energieversorger dienen die genaueren Verbrauchszahlen im Falle eines intelligenten Stromzählers dazu, die Bereitstellung von Strom im Netz zu optimieren und so den risikobehafteten kurzfristigen Zukaufbedarf an der Strombörse zu minimieren und die Kraftwerkeinsatzplanung so zu optimieren, dass die bereitgestellte Strommenge die Nachfrage nur minimal übersteigt. Um dies zu erreichen sind die intelligenten Zähler in der Regel aus der Ferne vom Versorger auslesbar.

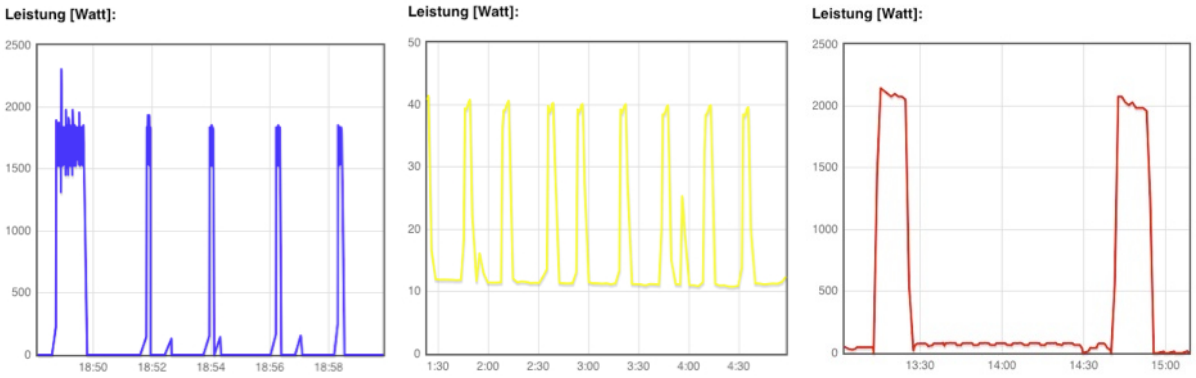
Wikipedia schreibt zum Thema intelligente Zähler:

> Ein intelligenter Zähler (auch Smart Meter genannt) ist ein elektronischer Stromzähler, der dem Energieversorgungsunternehmen über eingebaute Zusatzfunktionen oder nachträgliche Module ermöglicht, die erfassten Zählerstände über die Ferne auszulesen.
> [...] Neben dem reinen Einsatz von intelligenten Zählern zur Messung des Stromverbrauchs ist es auch möglich, in einem Haushalt den Wasser-, Gas-, und Wärmeverbrauch intelligent zu überwachen.
> [...] Der Zweck intelligenter Zähler ist vornehmlich, variable Leistungsentgelte in Abhängigkeit von der Gesamtnachfrage und Netzauslastung erheben zu können. Damit erhoffen die Energieversorger, das Netz und die vorhandene Kraftwerkinfrastruktur besser auszunutzen sowie Investitionen für Spitzenlastausbau zu vermeiden oder zumindest zurückzustellen.[4]

Bei den konventionellen Ferrariszählern wird in der Regel einmal im Jahr der Zählerstand abgelesen. Dadurch hat weder der Versorger noch der Kunde einen Überblick darüber, wann genau er wieviel Strom verbraucht. Bei den intelligenten Stromzählern wird der Verbrauch in deutlich kürzeren Intervallen erfasst.

Anhand der gemessenen Werte lassen sich Verbrauchsdiagramme erstellen. Hier seht Ihr als Beispiel ein paar Diagramme typischer Verbraucher eines Haushaltes:

[4] Quelle: http://de.wikipedia.org/wiki/Intelligenter_Zähler



Tatsächlich birgt die verbraucherfreundliche Lösung einer automatischen und feingranularen Verbrauchserfassung durch den Stromzähler die Möglichkeit zur Erstellung eines präzisen Lastprofiles. Der Lastgang eines Haushalts verrät viel über die Gewohnheiten der Personen, die in diesem Haushalt leben. Anhand der gemessenen Daten lassen sich Nutzungsprofile erstellen. Es sind im Prinzip Rückschlüsse auf die Lebensgewohnheiten des Haushaltes möglich:

- Wann stehen die Bewohner auf?
- Wann gehen Sie ins Bett?
- Wird jeden Tag gekocht? mit einer oder mit mehreren Herdplatten? oder gibts häufig Pizza? :-)
- Ändern sich die Gewohnheiten (Besuch oder Nachwuchs)?
- Muss ein Bewohner Nachts häufiger auf Toilette?

Im Prinzip hat man mit diesen Daten eine bessere Überwachung als mit einer Kamera!

Das Energiewirtschaftsgesetz (EnWG) schreibt den Versorgern nur vor, das auch der Kunde auf diese Daten zugreifen können muss. Die Nutzung, Weitergabe, Anonymisierung der Daten ist in dem Gesetz nicht explizit geregelt.

3 Was ist der Volkszaehler?

volkszaehler.org ist ein freier intelligenter Stromzähler im Selbstbau, bei dem die anfallenden Stromprofile unter der Kontrolle des Nutzers verbleiben. Die Daten des Volkszaehlers sind nicht duch den Versorger auslesbar. Mit einem Materialeinsatz von ca. EUR 100, etwas Geschick und Zeit lässt sich ein solcher Volkszaehler auf Basis eines Standard-µC-Moduls aufbauen.

Der Volkszaehler kann nicht als fertiges Gerät gekauft werden. Es ist ein Selfmade-Projekt. Der Volkszaehler besteht nicht aus einem einzigen Element sondern aus insgesammt 4 Modulen (Messung, Verarbeitung, Speicherung & Auswertung). Das Modul zur Messung muss in der Regel von einem Elektriker im

Schaltschrank der Hausinstallation eingebaut werden. Es gibt auch Varianten die keinen Eingriff in die Hausinstallation benötigen. Alle anderen Module lassen sich von interessierten Tüftlern selber nachbauen.

Die einzelnen Module und deren Zusammenspiel wird etwas weiter unten genauer erklärt.

4 Warum brauche ich einen Volkszaehler?

Wer seinen Energiebedarf analysieren möchte, braucht dazu genaue Messwerte. Bei heutigen Hausinstallationen mit konventionellen Drehstromzählern sind für den Stromverbrauch diese Messwerte nicht vorhanden. Es wird also ein intelligenter Zähler benötigt, der in der Lage ist den Energiebedarf über sehr kurze Zeiträume zu messen und zu speichern.

Diese Geräte werden heute schon von vielen Versorgern angeboten. Die einfachen Lösungen ermöglichen die Anzeige der aktuellen Leistung, der Verbrauchswerte des aktuellen Tages, der aktuellen Woche, des aktuellen Monats und der jeweils vorhergehenden Zeiträume. Bei den meisten Lösungen, bei denen darüber hinaus eine komfortable Auswertung möglich ist, werden die Verbrauchsdaten jedoch zum Versorger übertragen. Der Nutzer kann dann i.d.R. über eine Weboberfäche eine komfortable Analyse vornehmen.

Wie schon weiter oben geschildert besteht nun die Möglichkeit für den Versorger auf Grundlage der Daten Nutzerprofile anzulegen und diese auszuwerten.

So Lustig diese Gedankenspiele auch sind - mit einem intelligenten Zähler eines Versorgers gibt man seine Daten aus der Hand und weiß nicht was mit diesen Daten gemacht wird.

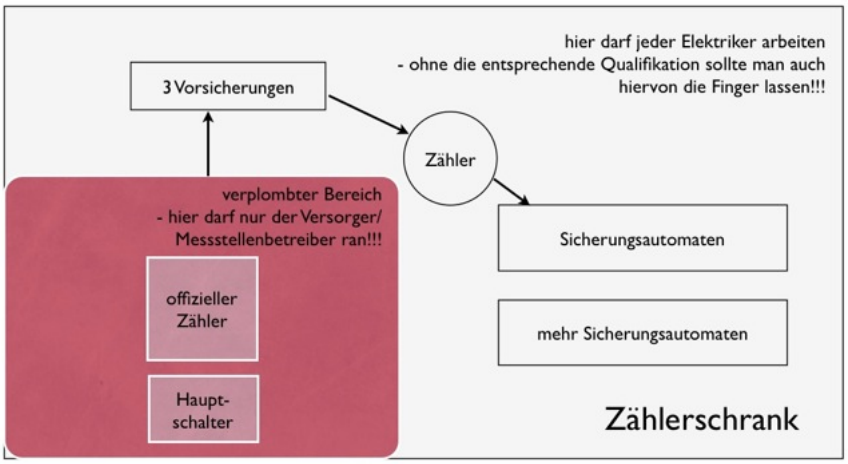
Wer dies nicht möchte, hat mit dem Volkszaehler die Möglichkeit sich einen intelligenten Zähler selbst zu bauen und die Verbrauchsdaten selbst auszuwerten. Die Daten bleiben beim Kunden und der Versorger hat keinen Zugriff auf diese Daten.

5 Wie funktioniert der Volkszaehler?

Der Volkszaehler besteht aus mehreren Modulen. Für jedes Modul gibt es unterschiedliche Ideen der Umsetzung. Es gibt also nicht den einen Volkszaehler, sondern in der technischen Ausführung verschiedene Varianten. Grundsätzlich funktionieren aber alle diese Varanten nach dem gleichen Prinzip.

Die Sensoren - in der Regel je einer pro Phase - werden im Schaltschrank der Hausinstallation zwischen den Vorsicherrungen und den normalen Sicherungsautomaten installiert. Diese Sensoren geben ihre Signale an ein µController-Board weiter, welches die Daten über eine Netzwerkverbindung an einen Datenbankserver zur speicherung weitergibt. Möchte der User nun die Daten abrufen und auswerten nutzt er dazu einen Webbrowser, mit dem er die Daten aus der Datenbank abrufen - und diese grafisch aufbereiten lassen kann.

Die folgende Zeichnung zeigt die Position der Sensoren im Schaltschrank. Das µController-Board kann, muss aber nicht, im Schaltschrank installiert werden.



6 Woraus besteht ein Volkszaehler?

Der Volkszaehler besteht aus 4 Modulen:

- Messen
- Verarbeiten
- Protokollieren/Speichern
- Auswerten

Für jedes dieser Module gibt es verschiedene Ideen diese zu realisieren. Mehr dazu findet Ihr im Wiki unter Morphologischer Kasten.

Hier möchte ich für jedes Modul eine Beispielkonfiguration vorstellen und kurz ein paar Infos dazu schreiben:

Das Projekt lässt sich in drei Bereiche aufteilen, die untereinander über eine spezifizierte API kommunizieren. Diese drei Module grenzen sich lokal, durch die verwendeten Technologien und ihre Aufgabe voneinder ab. Alle Module sollen untereinander austauschbar sein und sind in mehreren Varianten verfügbar. So kann den individuellen Bedürfnissen nach ein angepasstes Setup aufgebaut werden.

6.0.1 Messen

Ein möglicher Messsensor ist ein Wattmeter, welches auf der Hutschiene des Schaltschrankes montiert wird und zwischen den Vorsicherungen und den Sicherungsautomaten eingeschleift wird. In der Regel wird pro Phase ein Wattmeter benötigt.

hier ein Foto der Wattmeter im Schaltschrank einfügen!

Auf dem Foto sieht man unten die Stromanschlüße und oben - die etwas dünnere Kabel - die Signalleitungen die dann zum µController-Board gehen.

kurz die Funktionsweise der Wattmeter erklären

6.0.2 Verarbeiten

hier das AVR-Net-IO beschreiben

6.0.3 Speichern

hier den Webserver beschreiben

6.0.4 Auswerten

hier das Frontend beschreiben

Wer bis hierher gekommen ist und nun Lust auf mehr hat, kann sich gerne in dem Wiki umsehen ...

6.1 Controller

Die Aufgabe der Controller ist es Sensoren auszulesen und diese Daten direkt an das Backend zu senden. Meist sind sie direkt mit den Zählern/Sensoren verbunden. Ein typischer Ort wäre also der Sicherungskasten. Diese Controller sind dann meist in das lokale IP Netzwerk eingebunden und senden ihre Daten an ein Backend. Ausgehend von Prototyp sind im volkszaehler.org Projekt bisher die Atmel AVR/Ethersex basierenden Controller am meisten verbreitet.

Als Sensoren werden hier für das Erfassen des Verbrauchs Impulszähler eingesetzt. Diese signalisieren dem Controller durch einen elektrischen Impuls, das eine bestimmte Menge an elektrischer Energie, Wasser oder Gas verbraucht wurde. Auch skalare Messgrößen wie z.B. Temperatur, Luftdruck oder Luftfeuchtigkeit sind möglich, werden dann aber nicht mehr durch Impulse erfasst.

Ein typisches Setup für ein Einfamilienhaus könnte folgendermaßen aussehen:

- 3x 1-phasige S0-Stromzähler für den Hausanschluss[5]
- 1x S0-Wasserzähler
- 1x S0-Gaszähler
- 1x Sensor für die Außentemperatur

6.2 Frontend(s)

Die Frontends visualisieren Messwerte und können zur Verwaltung genutzt werden. Durch ein zentrales Backend haben wir die Möglichkeit die Messwerte an mehrere Frontends gleichzeitig zu verteilen. Verschiedene Frontends, können die Daten dann unterschiedlich darzustellen.

Das Web-basierte Frontend für den Browser ist das bisher einzige Frontend und ist quasi eine Referenzimplementierung der API. Es unterstützt die volle

[5] jede Phase erhällt einen Zähler

Funktionalität des Backend und kann auch zur Verwaltung von Zählern genutzt werden. Es nutzt AJAX um Daten dynamisch nachzuladen.

Inspiriert vom Wattson[6] haben wir uns entschieden eine ähnliche interaktive Variante dieses Moodlights zu entwickeln. Derzeit experimentieren wir mit fnordlichtern[7].

Die möglichen Visualisierungsmethoden sind praktisch endlos. Damit auch die Anzahl der möglichen Frontends. Frontends dürfen nicht nur als abgeschlossenes System betrachtet werden. Denkbar sind auch Plugins in bestehende Systeme (Social Networks, iGoogle, Twitter).

6.3 Backend

Im Backend werden die Messwerte der Sensoren gespeichert und ausgewertet. Es handelt sich hierbei um ein PHP Skript welches auf einem Webserver läuft und somit die Schnittstelle zwischen Controller und Frontend ist. Daraus folgend muss es von den Controllern sowie von den Frontends via HTTP erreichbar sein. Ob der Backendserver nur über das lokale Netzwerk oder auch über das Internet erreichbar ist, wirkt sich nur auf die Erreichbarkeit durch die Frontends und die Controller aus. So kann es durchaus erwünscht sein die Daten nur im lokalen Netzwerk vorzuhalten um sie gegen Zugriff über das Internet zu schützen.

7 API

Die API definiert eine Schnittstelle zwischen Controllern und Backend, sowie zwischen Backend und Frontends. Die API orientiert sich am REST Entwurfsmuster. Dabei kann jeder Zähler/Sensor/Gruppe (im folgenden als “Entity” bezeichnet) durch eine weltweit eindeutige UUID referenziert werden. Diese UUID werden entsprechend RFC 4122 von Backend generiert und vergeben. Dieser 128 Bit lange Identifier sichert uns auch gleichzeitig die Privatsphäre. Die UUID ist praktisch eine Zugangskennung und sollte immer vertraulich behandelt werden. Da wir keinerlei nutzerbezogene Daten speichern kann diese UUID nur durch den Zugriff eines Nutzers zugeordnet werden. Daher empfiehlt sich der Einsatz von verschlüsseltem HTTPS.

Als Ausgabeformat nutzen wir das leichtgewichtige und menschenlesbare JSON.

Eine Referenz unserer API befindet sich im Wiki.

7.1 Gruppierung

Zähler und Sensoren können zusammengefasst werden. Diese Aggregatoren besitzen selbst wieder eine UUID und können dadurch beliebig tief verschachtelt werden. Hirachische Strukturen wie z.B. Mehrfamilienhäuser, Wohngemeinschaften, Hotels, Gemeinden & Städte können dadurch nachgebildet und gemeinsam ausgwertet werden.

[6] http://www.diykyoto.com/uk/wattson/about
[7] http://wiki.lochraster.org/wiki/Fnordlichtmini

8 Implementierung

Dieser Abschnitt konzentriert sich auf die Implementierung des Backends. Das Backend ist in PHP programmiert und kann auf jedem LAMP-ähnlichen System betrieben werden.

Der Code ist stark Objekt orientiert. Daher müssen wir PHP 5.3 vorraussetzen. Diese Objekt orientierte Programmierung erlaubt es uns den Code übersichtlicher und modularer aufzubauen. Zukünftige Erweiterungen, neue Zählertypen können dadurch leichter in das System integriert werden.

8.1 MVC

Das Backend wurde nach dem **M**odel **V**iew **C**ontroller Entwurfsmuster aufgebaut. Jeder Request an das Backend muss über die Datei `/htdocs/backend.php` erfolgen. Sie ist quasi der "Haupteingang"[8].

Dort wird zunächst der PHP Interpreter konfiguriert (automatisches Laden von Klassen, der Konfiguration, Fehlerbehandlung, Aufbau der Datenbankverbindung). Danach wird direkt der Router initialisiert.

8.1.1 Router

Der Router ist das Herz der MVC Struktur. Er leitet den Request an den zuständigen Kontext[9] weiter und antwortet mit dem korrekten Ausgabeformat.

Alle Anfragen an das Backend müssen einem bestimmten Schema entsprechen. Hier wird auch der zuvor angesprochene Einsatz von REST deutlich.

```
http://server:port/path/to/volkszaehler/backend.php/kontext/uuid.format?paramters=values
```

8.1.2 Kontext (Controller)

Das Backend hat verschiedene Aufgaben zu bewältigen. Für jede dieser Aufgaben gibt es einen eigenen Kontext, der praktisch die ganze Logik enthällt:

- Daten erfassen/ausgeben
- Entities erstellen/bearbeiten/abfragen
- Eigenschaften des Backends abfragen

[8] Dadurch ist sie auch das einzige PHP Skript, das über das öffentliche "htdocs" Verzeichnis zugänglich sein muss

[9] Wir nutzen hier den Begriff "Kontext" um eine Verwechslung mit dem Hardware-Controllern zu vermeiden. Entsprechend des MVC-Patterns ist hier der "Controller" gemeint.

8.1.3 Datenabstraktion (Model)

Zur Datenbankabstraktion setzen wir Doctrine 2 ein. Dieses Database Abstraction Layer (DAL) erlaubt es uns datenbankspezifische Eigenheiten, SQL-Dialekte ähnliches zu ignorieren. Doctrine unterstützt verschiede Datenbanken:

- MySQL
- SQLite
- PostgreSQL
- Oracle

Aufbauend auf das DAL kommt der Object Relational Mapper (ORM) von Doctrine zum Einsatz. Wir können mit den Daten nun als Instanzen von PHP-Objekten arbeiten. SQL Queries sind nicht mehr nötig. Für weitere Infos empfehle ich die Dokumentation des Doctrine Projekts.

8.1.4 Ausgabe (View)

Das Backend kann in verschiedenen Formaten antworten:

- JSON (Standard)
- XML
- CSV
- Klartext
- Jpeg, PNG, Gif

Dabei ist JSON das bevorzugte Format, das auch von dem Webinterface genutzt wird. Die anderen Formate sind als Alternativ für Endgeräte ohne JSON Unterstützung gedacht. Wir können nicht garantieren, dass diese Formate den gleichen Funktionsumfang wie das JSON Format besitzen.