# SolvisSmartHomeServer / Fhem Client

## Einführung

Die SolvisRemote bietet einen Zugriff über mehrere http-Seiten an. Über diese kann der aktuelle Status der Anlage untersucht werden und Einstellungen der Anlage verändert werden.

Dieser Zugriff eigen sich jedoch nicht direkt zur Integration in ein SmartHome-System wie Fhem, OpenHAB u.a..

Bisher existierte für das SmartHome-System FHEM ein Modul, das die Messwerte der Anlage in FHEM zugänglich machen konnte sowie den Anlagenmodus (Tag/Nacht/Standby/Timer) verändern konnte. Letzteres funktionierte nicht ganz zuverlaässig.

Ziel des vorliegenden Projektes war, auch weitere Anlagenparameter von einem SmartHome-System einstellen zu können und möglichst eine Insel-Lösung nur für das Fhem-SmartHome-System zu vermeiden.

Bei dem neuen Modul handelt es sich daher um eine Server-Client-Lösung.

Der Server dient zum Auslesen der Messwerte und Auslesen/Einstellen der Anlagenparameter. Der Server ist in Java geschrieben, da aus meiner Sicht für größere Projekte eine Script-Sprache (wie Perl/JavaScript u.a.) wenig geeignet ist und ich in den letzten Jahren beruflich viel in Java programmiert habe.

Mit diesem Server können sich mehrere SmartHomeClients gleichzeitig verbinden. Auf diese Weise kann man von verschiedenen SmartHome-Systemen den Server ansprechen. Die etwas CPU-zeitintensive Verarbeitung erfolgt nur an einer Stelle, dem Server.

Auf der SmartHome-Seite ist nur noch ein relativ einfacher an das verwendete System angepasster Client zu realisieren.

Der Datentransfer zwischen Server und Client erfolgt über das JSON-Format, welche recht einfach über eine Library eingelesen/erstellt werden können. Das Format im Einzelnen ist im vorliegenden Dokument im Anhang beschrieben.

## Voraussetzungen

### Solvis Anlage, SolvisRemote

Grundvoraussetzung zur Verwendung der vorliegenden Lösung ist natürlich eine **Solvis-Anlage** sein, welche die **SolvisControl 2**(seit 9/2007) verwendet (SolvisMax, SolvisBen), die **SolvisRemote** muss noch zusätzlich vorhanden sein.

### Server

Der Server kann auf verschieden Systemen laufen, für die es ein Java-Run-Time-Environment gibt. Entwickelt habe ich es auf einem Windows-System mit Oracle-JDK 8. Im Einsatz habe ich es auf einen Raspberry Pi 3 Modell B, getestet auf einem Raspberry Pi 2 Modell B, jeweils mit OpenJDK 9. Auf einem Raspberry Pi 2 kostet es etwa 1% der CPU-Zeit, höchst wahrscheinlich wird es auch noch auf einem Raspberry der ersten Generation lauffähig seine, was ich aber nicht getestet habe.

Im Betrieb laut „top“ benötigt das System (einschl. OpenJDK) ca. 60 MByte Speicher (af dem Raspberry Pi 2).

### SmartHome-System

Zusätzlich ist natürlich ein SmartHome-System mit einem entsprechenden Client notwendig.

Aktuell existiert nur für das SmartHome-System FHEM ein entsprechendes Modul.

Da ich mit dem Fhem-SmartHome-System nicht wirklich zufrieden bin (ich mag einfach kein Perl), spiele ich mit dem Gedanken auf OpenHAB umzusteigen.

Auf dem Server kann natürlich auch das SmartHome-System laufen, muss aber nicht.

## Verwendete Schnittstellen der Solvis-Anlage

Die Messwerte der Anlage, welche im Anlagenschema angezeigt werden, können noch recht gut unter der folgenden Adresse als Hex-String verpackt einem vereinfachten XML-Rahmen ausgelesen werden.

Dies kann man unter der folgenden Adresse auslesen:

http://<tcp-ip-Adresse der Solvis-Anlage>/sc2\_val.xml?

Dieser String wurde bisher durch das Fhem-Modul „73\_SolvisMax.pm“ ausgewertet um die Daten auf der FHEM-Oberfläche darstellen zu können.

Über diesen Weg lassen sich jedoch nicht die Anlagenparameter – wie Tag-/Nacht-Temperatur, Raumeinfluss etc. – verändern. Das geht nur über die SolvisControl, welche unter folgender Adresse zugänglich ist:

http://<tcp-ip-Adresse der Solvis-Anlage>/remote.html

Das bisherige FHEM-Modul „73\_SolvisMax.pm“ ließ hier nur sehr rudimentäre Zugriffe auf die SolvisConrol zu, es waren nur die Anlagenmodus Tag/Nacht/Timer/Standby wählbar. Ab und zu erkannte die SovisControl einer dieser Betätigungen nicht, so dass man es wiederholen musste, für ein zuverlässiges SmartHomeSystem nicht wirklich verwendbar.

Das vorliegende neue Modul nutzt auch die obigen beiden Wege, erweitert den Weg über die SolvisControl um die Interpretation des Bildschirminhaltes um in Abhängigkeit vom Bildschirminhalt die Buttons der SolvisControl passend zum einzustellenden Wert bedienen zu können. Der dann eingestellt Wert wird immer verifiziert, so dass verloren gegangene Button-Betätigungen erkannt werden.

Der Bildschirminhalt der SolvisControl wird folgender Http-Zugriff gelesen:

http://<tcp-ip-Adresse der Solvis-Anlage>/ display.bmp?

Der Ursprung des Koordiantensystems ist wie bei Bildern üblich oben links.

Die Betätigung der << Buttons wird folgender Http-Zugriff verwendet:

http://<tcp-ip-Adresse der Solvis-Anlage>/Taster.CGI?taste=links

Zum Betätigen eines angezeigten Buttons auf dem Bildschirm erfolgt über folgenden Http-Zugriff:

http://<tcp-ip-Adresse der Solvis-Anlage>/Touch.CGI?x=<x>&y=<y>

Hierbei sind <x> und <y> die Koordinaten des Buttons aus dem obigen Bild multipliziert mit zwei (auf der http-Seite wird das Bild der SolvisControl um den Faktor 2 vergrößerst dargestellt.

## Interne Komponenten des SolvisSmartHomeServer

Der SolvisSmartHomeServer besteht aus 3 Funktionseinheiten

### Server

Der eigentliche Server stellt die Schnittstelle nach außen dar. Er nimmt Verbindungen von bis max. 50 Clients entgegen, interpretiert deren Befehle und sendet die Solvis-Daten an die Clients.

### Messwerte-Erfassung

Dieser fragt regelmäßig (default alle 10s) den Solvis-Hex-String mit den Messwerten ab, interpretiert den Hexstring und sendet bei einer Änderung dem Client die gemessenen Werte. Für bestimmte Daten (Temperaturen) erfolgt eine Mittelwertbildung über eine Messwerte-Reihe (Default: 12, entspricht über einen Zeitraum von 2 Minuten). Erkennt dabei das Modul einen Wert, der vom Mittelwert stärker abweicht als die normale Schwankungsbereich des Sensors, wird der Wert bei der Mittelwertbildung doppelt gewichtet, so dass der vom Modul gelieferte Wert trotz Mittelwert-Bildung dem wirklichen Wert bei größeren Änderungen besser folgt (z.B. Aufheizung des Kesselwassers durch laufenden Brenner).

### Auswertung und Steuerung über die SolvisControl-Bildschirme

Zur Interpretation des Bildschirminhalts wurde ein abgespecktes OCR realisiert, das folgende Zeichen der SolvisControl erkennen kann:

+ - 0 1 2 3 4 5 6 7 8 9 ° C [ ] : . / h %

Zur Identifikation der einzelnen Screens werden gezielt bestimmte rechteckige Flächen der Screen untersucht. Zusätzlich kann auch der Identifikation das OCR herangezogen werden. So wird für Screens der Heizkreise nur die Überschrift herangezogen, welche einzelne Screen gerade angezeigt wird, wird durch die Detektion der

## Ablauf des Programms nach dem Start

Das Programm durchläuft nach dem Start drei verschiedene Phasen. Erst in der dritten Phase sind sämtliche Messwerte eingelesen und das Modul ist zur Steuerung der Solvis-Anlage bereit.

### Phase 1: Learning (nur beim ersten Start)

Beim ersten Start des Programmes und nach Änderung des „control.xml“-Files müssen die Grafiken angelernt werden, die zur Identifikation der Bildschirme und Status-Buttons benötigt werden, angelernt werden. Das macht das Modul großteils vollautomatisch, in dem es durch die verschiedenen Bildschirme der SolvisControl geht und sich dabei die zur Identifikation notwendigen Bildschirmbereiche merkt. Diese werden in die Datei „graficData.xml“ gespeichert, so dass die Learning-Phase nur beim ersten Starten des Moduls durchlaufen wird.

### Phase 2: Auslesen der aktuellen Anlageparametern

In dieser Phase wird das zyklische Auslesen der Messwerte gestartet. Der Client erhält entsprechenden Werte.

Gleichzeitig erfolgt das Auslesen der Anlageparametern von der SolvisControl.