Every *Thing* Under the Sun: How Web of Things and Semantic Data Brings Benefit to Small-Scale Photovoltaic Installations

Ganesh Ramanathan
ganesh.ramanathan@student.unisg.ch
University of St.Gallen
Siemens AG
Switzerland

Srinivas Marella
Energy Services Technology Enabler
Hyderabad, India
srinivask.marella@gmail.com

Abstract

Small-scale photovoltaic (PV) systems of up to a few kilowatts capacities are becoming increasingly available and affordable for off-grid installations. However, in our experience with using PV energy in farming in India, we found that many installations had faults or were lying underutilized. Though integrating such systems into IoT applications is now practical, analyzing the system's performance and utilization requires knowledge of the components and the system design. Off-band infusion of this knowledge into the software applications leads to tight coupling and vertical silos. To address this challenge, we have developed an ontology to describe small-scale PV installations, which enables us to represent subsystems and their components in the form of Web of Things (WoT) Thing Descriptions. We show that our approach results in technical and semantic integration of the PV system into IoT applications, allowing the development of reusable fault detection and optimization programs. This reduces the cost of developing solutions to monitor and optimize the usage of PV systems, thereby bringing benefits to the farming community by improving their livelihood.

Keywords

Renewable Energy, Photovoltaics, Semantic Data, Web of Things

ACM Reference Format:
Ganesh Ramanathan and Srinivas Marella. 2022. Every *Thing* Under the Sun: How Web of Things and Semantic Data Brings Benefit to Small-Scale Photovoltaic Installations. In *Proceedings of the 12th International Conference on the Internet of Things (IoT '22), November 7–10, 2022, Delft, Netherlands.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3567445.3571113

1 Introduction

The growing use of Photovoltaic (PV) technologies to meet energy requirements is facilitated by declining infrastructure costs and easy availability of components worldwide [10, 14]. As a result, small-scale PV systems are available as off-the-shelf kits or components that can be installed and commissioned by semi-skilled technicians. This trend not only has a high social impact (especially in developing economies) but also promotes the migration towards an environment-friendly alternative for energy needs [13]. However, the author in [14] has also pointed out that the lack of insufficient technical support, maintenance, and long cost-recovery time has posed significant barriers to the more widespread adoption of PV systems.

PV systems are prone to faults and degradation of components during their lifetime, which results in sub-optimal performance. Such faults are challenging to detect and diagnose because PV systems contain multiple subsystems and components like PV arrays, charge controllers, batteries, and load-side controls. A fault in any of these parts can lie undetected while causing inefficient operation or reducing the life of the components. For example, poor electrical contacts or malfunctioning of the charge controller can result in energy loss as high as 18% [1]. In an extensive study that evaluated the use of small-scale PV systems in developing economies [19] it was pointed out that premature degradation of battery as a result of wrong installation or configuration of components was one of the common causes of failures. The authors also have pointed out that in the absence of monitoring capability, the system performance evaluation becomes difficult (e.g., understanding the charge controller's performance requires measurement of input and output voltages and currents). One of the authors of this paper, who is a field practitioner helping small farming communities in India adopt PV, confirms this as a prevalent problem.

Optimum usage of PV systems requires knowledge about generation characteristics, storage capacities, and the expected load profile. Often, users cannot decide on scheduling loads because they have insufficient information about the state of the PV system and the energy profile of the loads. In this regard, research into methods for optimizing PV energy usage (e.g., [11]) shows encouraging results for using data integrated through IoT infrastructure.

However, the widespread application of such fault detection and optimization techniques to real-life installations is yet to be seen. From our experience, several factors contribute to this challenge. Firstly, the heterogeneity of subsystem and component configurations and specifications makes it challenging to make the knowledge available to software agents responsible for fault detection or optimization. Secondly, such software agents require not only the components' specifications but also a description of the system topology. Due to the absence of an ontology to describe both these aspects, existing methods rely on off-band integration of system design knowledge (either hard-coded or based on proprietary formats). Finally, IoT *data lakes*, which contain measurement data, need to be linked to the semantic description of the system, and current approaches are lacking in suggesting a mechanism to enable this.

IoT '22, November 7–10, 2022, Delft, Netherlands
© 2022 Copyright held by the owner/author(s).
ACM ISBN 978-1-4503-9665-3/22/11.
https://doi.org/10.1145/3567445.3571113

To address this challenge, we developed an ontology [1] based on our study of small-scale PV systems in various configurations (in contrast to the domains such as smart grid or distributed energy management systems were more resources and research is available). The practical effectiveness such a knowledge-driven integration of PV systems into IoT infrastructure was evaluated by developing several fault detection and load management programs. Our primary non-functional requirement for these programs was to make them easily applicable to different small-scale PV installations (each with a different configuration and component specifications). In addition, we aimed to keep the programs agnostic of the IoT infrastructure-specific interfaces and data models. The infrastructure-agnostic nature of our approach avoids the need to develop or customize fault detection and optimization programs for each installation. From our implementation and evaluation, we could confirm that using semantic technologies to describe the PV system and couple it with the machine-understandable description of the components using WoT Thing Descriptions enabled us to create reusable software programs.

2 Related Work

In general, having machine-understandable descriptions of engineering systems allows intelligent software agents to reason about the system's construction, and its working [25] [20]. Therefore, the potential advantage of semantic descriptions to data in IoT was realized quite early on, for example, with the development of the SSN/SOSA ontologies [12]. However, software agents which must reason about the system's functioning need knowledge not only about low-level sensors and actuators but also about components and their inter-relationships in the physical processes. Ontologies that describe technical systems where PV installations are used, such as in homes and buildings [4], do not contain a detailed description of PV systems. On the other hand, ontologies such as the one from Open Energy Platform [2] are suitable to list the components of a PV system but not oriented towards describing the construction and the configuration of the system. OntoPowSys [8] assists in specifying the relationship between the electrical components and can be potentially reused in conjunction with an ontology that covers PV system-specific terms. An ontology for modeling the performance of PV systems has been proposed by [15], which is valuable for reasoning about the system's efficacy. However, it lacks terms and relationships to model the PV system installation. It is also focused on modeling the performance of large installations that use the maximum power point tracking (MPPT) algorithm for charge control. The U.S Department of Energy has provided an extensive glossary of terms used in PV systems [3], which contains terms that are adequate for describing small-scale installations. However, this glossary is available only in human-readable form.

At the component level, the BattInfo ontology [7] provides support to describe the battery characteristics (which is one of the critical components in a PV system). However, ontologies to describe other critical components, like the PV modules and charge controllers, are missing. For PV modules, the NEC 690.53 and EN50380 standards define the essential electrical and mechanical characteristics which should be available in documentation and labeling. Again, the specification and the contained terms are not available in a machine-readable (and machine-understandable) format.

Table 1 summarizes the features offered by the individual ontologies as seen from the perspective of small-scale PV systems. Here we see that the available ontologies offer sufficient taxonomic support to describe the components, but lack the feature to describe the system construction and operational characterstics (both of the PV system and the load).

Several researchers have shown the benefit of connecting PV systems to the internet so that they can be monitored remotely by software programs [17], and in some cases using the connectivity also to change operational parameters of the system. However, the cost of incorporating IoT technologies in the PV system can be economically unviable. This challenge is sparking research and the maker community to explore low-cost hardware, and open-source IoT stacks [6]. Some PV charge controller products offer built-in connectivity using Bluetooth Low Energy so that the user can use their smartphone to configure and monitor their installation. Also, the possibility to use low-cost, long-range wireless technology like LoRaWAN allows PV systems in small clusters, for example, in villages, to be integrated into a local network and then to services on the internet via an edge device [22]. This plays a significant role in reducing the cost of hardware and internet connectivity.

As in most IoT implementations [2], PV monitoring solutions also use protocols like MQTT, HTTP, or Web Sockets to connect the devices to services on the internet. The information models are, however, proprietary and product-specific (or determined by the IoT stack being used). Therefore, a key challenge in IoT integration concerns semantic and technical interoperability. The approach behind W3C's Web of Things architecture [23] and the method of describing things [24] offers the potential to avoid vertical silos in the IoT stack.

Data from PV systems is valuable for fault detection since early detection helps avoid unnecessary energy loss and deterioration of components. In [1], a survey of various fault detection methods for PV systems is presented. Since most current methods are data-oriented, the authors in [9] show how IoT technologies can be applied effectively to gather measurement data to detect faults of various kinds. In [26], the decision tree model has been used to process voltage-current data to detect faults caused by shading whereas [21] recommends using simulation models. However, in both data- and model-based approaches, the fault detection algorithm and the data pre-processing step require knowledge about the component characteristics and the system design. Also, reusing such programs outside the IoT ecosystem in which they were created is often difficult. This is because they tend to be tightly coupled to the data sources due to the use of proprietary rules to recognize the semantics of the data points and knowing about the interactions to read or write values. To tackle this challenge, an approach adopted in the automation industry is tagging data points with domain terms to explain their context in the process. For example, Haystack tags [4] are being used in the building automation domain. Tagging of data points offers ease of understanding during the engineering of

[1]https://raw.githubusercontent.com/Interactions-HSG/SemanticSolar/main/ontology/pvont.ttl
[2]https://openenergy-platform.org/ontology/
[3]https://www.energy.gov/eere/solar/solar-energy-glossary

[4]https://project-haystack.org/

	Brick	SAREF	BOT	SSN	OEP	OntoPowSys	QUDT
Components	Partly	No	No	No	Yes	Partly	No
Electrical connections	No	No	No	No	Partly	Yes	No
Location and geometry	Yes	No	Yes	No	No	No	No
Measurements	Partly	Partly	No	Partly	No	Partly	Yes
Control functions	No	No	No	No	Partly	Partly	No

Table 1: Overview of available ontologies and their coverage of concepts required to model small-scale PV systems

the system and can be used to automatically generate metadata in the context of broader system knowledge [3].

However, beyond the semantics of the data points, understanding engineering systems (in general) for purposes like fault detection and optimization requires the description of the components and the physical processes they play a role in [16]. The seminal work of Borst [5] was instrumental in inspiring ontologies to describe several engineering systems [16].

Therefore, our motivation behind this work is to enable intelligent software agents to reason about the PV installation by making machine-understandable knowledge of the system available to them.

3 Approach

To make the system knowledge accessible, we developed an ontology for small-scale PV systems by putting together existing terminologies and adding concepts to support the description of control functions, electrical characteristics, and loads. A key feature of our approach is that we consider semantically describing the system installation as well as the process data delivered through the IoT infrastructure. In other words, we recognize that in real-life deployments of IoT infrastructure, system knowledge and run-time data are not necessarily transported on the same information channel. Therefore, our ontology seeks to establish links between these two at the source.

The effectiveness of the ontology was evaluated by applying it to several real-life installations, showing that our approach enables experts to create fault detection and optimization programs based solely on domain knowledge. Further, we scrutinized whether such programs could be applied to different installations without having to adapt them.

3.1 Ontology Design

To identify concepts required for our ontology, we studied several real-life deployments of PV systems. A bare-minimum configuration of such a system consists of a PV module, a charge controller, and a battery. In contrast, in a more complex setup, multiple PV modules (called an array) may be connected in parallel or series (or a combination of them), and the system may have multiple batteries and charge controllers.

We then analyzed the system knowledge and run-time data which will be required to evaluate fault rules and determine optimum load scheduling. Based on this, our ontology design considered three main aspects for modeling (Figure 3(a)): 1. Description of the components and inter-relationships (i.e., electrical topology), 2. Description of the charge control and load management strategy along with its configuration parameters, and 3. Semantic tags to annotate process data measured during run time. While 1. and 2. provide knowledge to software programs to reason about the system state, 3. enables the programs to find the appropriate data points in the IoT stack.

We modeled our ontology using the Web Ontology Language (OWL), which provides means to describe terms and relationships formally. The formalization, based on Description Logic (DL), enables software reasoners to draw inferences using facts stated in the ontology. In the following section, the ontology is described and then followed by an explaination of how it enables the extraction of knowledge which is then coupled to W3C WoT Thing Descriptions.

3.1.1 Components and Inter-relationships Since existing ontologies cover only part of the taxonomy required to describe PV systems, our ontology extends them with terms in the glossary from the U.S Department of Energy. At the highest abstraction level, we classified the components according to their role in the electrical circuit – i.e., as active or passive devices. Each component has electrical terminals that can be annotated with specifications like polarity, current limit, etc., and the terminals can be linked to form a circuit. This enables the application of rules (e.g., using SWRL as the one show in listing **??**) to infer circuit configuration (e.g., if the panels are wired in parallel or series) for the purpose of fault detection and diagnostics. In addition, the ontology offers extensive data properties to specify the PV module and the battery, as these are crucial for fault detection and optimization purposes. Another novel feature of the ontology is that it provides classes to specify electrical characteristic curves like voltage-current, power-current, and load profile. The availability of such characteristic curves in RDF enables the determination of optimal operation points using SPARQL queries instead of relying on the syntactical representation of mathematical functions.

```
{swrlexample}
    PVModule(?m1) ^ PVModule(?m2)
    ^ hasTerminal(?m1, ?t1)
    ^ hasTerminal(?m2, ?t2)
    ^ connectedTo(?t1, ?t2)
    ^ hasPolarity(?t1, positive)
    ^ hasPolarity(?t2, negative)
    -> inParallelTo(?m1, ?m2)
```

```
{sparqlexample}
SELECT * WHERE {
    ?bat a  pvont:LeadAcidBattery.
    ?bat pvont:connectedTo ?con.
    ?con a pvont:PWMController.
    ?bat pvont:batteryVoltageNominal ?v.
    FILTER(?v > 12.0) }
```

3.1.2 Control Functions One of the key factors determining the efficiency of a PV system is the choice of battery charging and load management strategies. Our ontology lists the commonly available

```
SELECT * WHERE {
    ?bat a  pvont:LeadAcidBattery.
    ?bat pvont:connectedTo ?con.
    ?con a pvont:PWMController.
    ?bat pvont:batteryVoltageNominal ?v.
    FILTER(?v > 12.0) }
```

Figure 1: An example SPARQL query

```
    PVModule(?m1) ^ PVModule(?m2)
    ^ hasTerminal(?m1, ?t1)
    ^ hasTerminal(?m2, ?t2)
    ^ connectedTo(?t1, ?t2)
    ^ hasPolarity(?t1, positive)
    ^ hasPolarity(?t2, negative)
    -> inParallelTo(?m1, ?m2)
```

Figure 2: An example rule in SWRL

strategies and their required inputs, outputs, interlocks, and configuration parameters. This knowledge is essential, especially in IoT integration of PV systems which allows the remote update of the parameters. Some of the control functions have the potential to be implemented in cloud services – in this case, the ontology will help to describe the function and enable its matching to an installation.

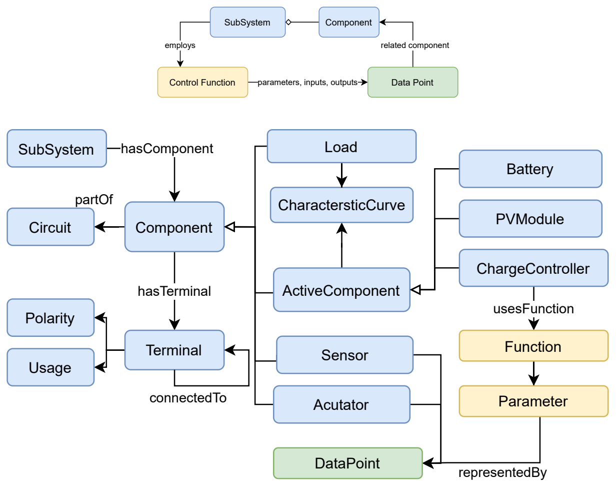


Figure 3: Overview of our PV systems ontology and the relationship between the concepts

3.1.3 Extracting Knowledge Given an ontology to describe the PV system, the next challenge was to obtain information from the PV installations. For this purpose, we designed a simple Web application that allows the user to list the components and their specification. Alternatively, for translating information contained in existing legacy data sources (e.g., text files), we explored the use of RDF Mapping Language (RML) for declarative specification of the translation. In addition, since many of the PV module manufacturers provide labels conforming to the EN50380 standard, we also experimented with optical character recognition to read the label content and translate it to RDF and this showed promising results since the terms used in the labels are standardized. The result of such knowledge extraction is that we have a RDF graph which describes the subsystems and component (Figure 3(b) illustrates this with an example). An alternative or an augmenting approach is to use the measurement data along with semantic rules to fill in the missing information. For example, this can be used to determine the PV module and battery voltages or capacities (here, we believe there is a potential for applying machine learning techniques to recognize configuration and specification from data).

3.2 From IoT to WoT

So far, we have described how system knowledge about PV systems can be modeled and obtained from installations so that they can be used by software programs to detect faults or optimize performance. Furthermore, such software programs need access to run-time data (i.e., measurements). We now describe our approach to linking system knowledge to the run-time data available through the IoT infrastructure.

3.2.1 Run-time Data Monitoring a typical small-scale PV installation requires only a modest amount of data points, such as voltages and currents at the PV module, battery, and load. In addition, data related to weather (outdoor temperature and solar irradiation in particular) and battery temperature help understand the system's state. There are several off-the-self IoT devices available which have flexible input-output (IO) ports where sensors and actuators from the PV system can be attached [5]. Many of these solutions allow the user to specify labels and additional descriptions while configuring the IO data points. However, in the absence of a standard terminology or naming convention, the metadata attached to IO data points require custom interpretation in code on the consumer side. To address this, we used syntactic tags offered in our ontology to annotate the data points. For example, an analog input annotated using the tags "panel, current" can be inferred using the system knowledge as the input current to the charge controller. Therefore, adopting the approach of tagging data points for later linking to the semantic description of the components enables our solution to be agnostic of the IoT infrastructure.

```
{tdmodule.jsonld}
{"title": "PV Module 1",
  "@id": "urn:pvmodule1",
  "@type": "pvont:PVModule",
  "pvont:pvRatedPower":40,
  "pvont:connectedTo": "urn:controller1",
  "properties":[{
     "outputVoltage": {
     "@type": "qudt:Voltage",
     "@id": "urn:pvmodule1_voltage",
     "pvont:hasTag": "pvont:output",
     "type": "number",
     "forms": [{
```

[5]We also found that several open source solutions offered by the IoT *maker* community are robust and more open to customization

```
      "op": "readproperty",
      "href": "http://server/mod1/voltage"
  }]}}}
{tdcontr.jsonld}
{"title": "Charge Controller",
 "@type": "pvont:PWMController",
 "@id": "urn:controller1",
 "pvont:pvRatedCurrent":50,
 "pvont:hasInput": "urn:module1",
 "pvont:hasControlFunction": "pvont:CCCV",
 "properties":[{
    "inputVoltage": {
    "@type": "qudt:Voltage",
    "pvont:hasTag": "pvont:input",
    "link": {"href":"urn:"urn:pvmodule1_voltage",
            "rel": "pvont:source"}
  }}}
```

Listing 1: Partial view of a TD for a PV module showing examples of metadata, links to other components, and property interaction

```
{"title": "PV Module 1",
 "@id": "urn:pvmodule1",
 "@type": "pvont:PVModule",
 "pvont:pvRatedPower":40,
 "pvont:connectedTo": "urn:controller1",
 "properties":[{
    "outputVoltage": {
    "@type": "qudt:Voltage",
    "@id": "urn:pvmodule1_voltage",
    "pvont:hasTag": "pvont:output",
    "type": "number",
    "forms": [{
      "op": "readproperty",
      "href": "http://server/mod1/voltage"
   }]}}}
```

Listing 2: The snippet of TD of the controller showing its link to the PV module. Semantic description of the system has enabled to link the properties (i.e., the output voltage of the PV module is the expected input voltage at the controller)

```
{"title": "Charge Controller",
 "@type": "pvont:PWMController",
 "@id": "urn:controller1",
 "pvont:pvRatedCurrent":50,
 "pvont:hasInput": "urn:module1",
 "pvont:hasControlFunction": "pvont:CCCV",
 "properties":[{
    "inputVoltage": {
    "@type": "qudt:Voltage",
    "pvont:hasTag": "pvont:input",
    "link": {"href":"urn:"urn:pvmodule1_voltage",
            "rel": "pvont:source"}
  }}}
```

3.2.2 *From Knowledge and Data to Things*

Availability of abstractions of the real-world entities facilitates software programming – a paradigm, for example, adopted by object-oriented programming [18]. However, IoT architectures primarily focus on providing connectivity, security, data storage, and access interfaces. It is left up to the software programmers to reconstruct the abstractions of the real-world entities (or *things*) and know about the interactions afforded by them. This challenge of lack of semantic and technical interoperability is addressed by the W3C's WoT approach, which brings together the semantics of the things and describes their interactions. Therefore, in our approach, we use the Thing Description (TD) to represent the real-world entities in the PV systems – i.e., its subsystems and components. From the previous steps, we have access to both semantics (system knowledge) and the interactions (via IoT infrastructure), and this enables us to *derive* the TD of the components (e.g., see Listing lst:tdmodule). Since our ontology also describes the concept of a subsystem (e.g., for energy storage) and enumerates its interactions, it also helps us derive TDs for the subsystems.

4 Evaluation

Our evaluation aimed to confirm if semantic descriptions of PV systems and the consequent availability of the abstractions as TDs can enable the creation of reusable fault detection and optimization programs. In other words, we expect that machine-understandable descriptions of PV systems that provide a semantic representation of the components and interactions (in the form of *things* with TDs) will allow ***loose coupling*** between the analytical programs and installation- and IoT stack-specific knowledge. To confirm this, we integrated four real-life installations of PV systems into an IoT infrastructure and described the installations using our ontology. Data was gathered from these installations for a period of 18 months. Software programs to analyze faults and determine optimum load scheduling were developed by us while keeping in mind that it should not be tightly coupled to any installation-specific or IoT infrastructure-specific features. These two steps are now described in more detail.

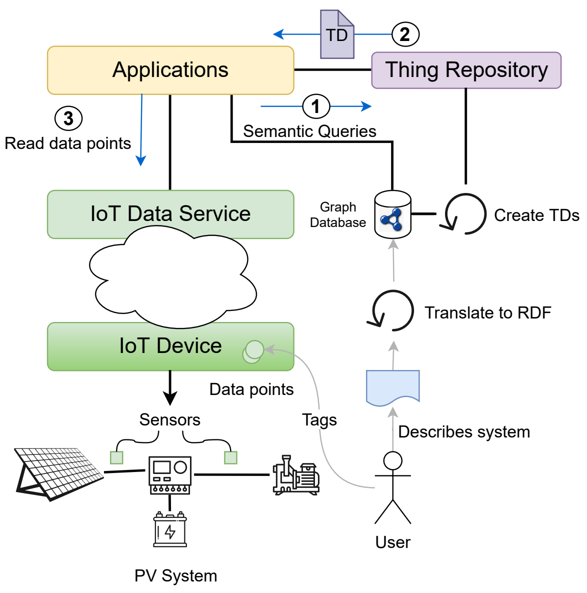
4.1 Experimental Setup

4.1.1 *IoT Infrastructure*

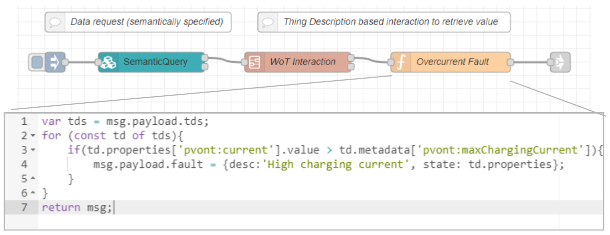
The primary IoT-related function required in our use case is to send measurement data from the PV installation periodically to an IoT infrastructure in the cloud where it can be stored. Performance monitoring of a typical small-scale PV installation requires measurement data of voltages and currents at the PV module, battery, and load. In addition, outdoor temperature, solar irradiation, and battery temperature are also helpful in understanding the system's state. To correlate the data points to the semantic description of the components, we require the data point information model to support some form of freely definable text annotation (e.g., description text). Finally, to retrieve the data from the IoT cloud service, we require an API-documented communication interface based on one of the well-known standards like HTPP(s), MQTT, CoAP, etc. Therefore, our approach and evaluation remains independent of the IoT infrastructure. Figure 4(a) gives an overview of the knowledge and data flows.

4.1.2 *Semantic Descriptions*

The information about the PV installations was entered into a Microsoft Excel Worksheet which was



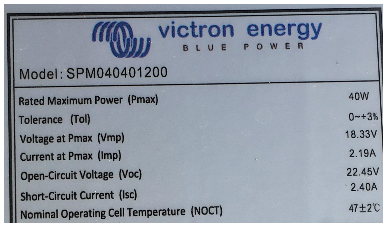
(a) Architectural overview


Data request (semantically specified)
Thing Description based interaction to retrieve value
SemanticQuery
WoT Interaction
Overcurrent Fault

```
var tds = msg.payload.tds;
for (const td of tds){
    if(td.properties['pvont:current'].value > td.metadata['pvont:maxChargingCurrent']){
        msg.payload.fault = {desc:'High charging current', state: td.properties};
    }
}
return msg;
```

(b) Node Red-based application

Figure 4: The translation of system description and creation of TDs is automated by the programs. During run time, the application program makes semantic queries (1), which then returns links to the TDs representing the relevant components/subsystems. The TDs are retrieved from the repository (2) and the interaction description in it enables the application program to read data points (3) from the IoT service.



victron energy
BLUE POWER

Model: SPM040401200

Rated Maximum Power (Pmax)	40W
Tolerance (Tol)	0~+3%
Voltage at Pmax (Vmp)	18.33V
Current at Pmax (Imp)	2.19A
Open-Circuit Voltage (Voc)	22.45V
Short-Circuit Current (Isc)	2.40A
Nominal Operating Cell Temperature (NOCT)	47±2℃

Figure 5: Label on PV panel from which data can be extracted using OCR.

then translated to RDF using RDF Mapping Language (RML [6]). Alternatively, the information could be provided using a simple Web application that exports the RDF data. As yet another alternative, optical character recognition (OCR) on images of component labels taken using a smartphone to extract product specification information was explored (Figure 5 shows an example of a label that conforms to the EN50380 standard, based on which the text from OCR can be converted to RDF).

The list of data points, along with their annotations (tags), was exported as a CSV file from the IoT cloud service. In case of some IoT devices, this list can be obtained from its configuration tool directly. For this purpose, RML was used to transform the data point list to RDF. This way, the relationship between the components and the relevant data points could be established based on the tags.

In the next step, a software program was developed to generate WoT TDs for the components. The program queries the knowledge graph to create a property interaction for each measurement data point available for the component and adds the metadata derived from the product specification. For example, the TD of a battery contained properties for voltage, current, and temperature measurements along with metadata like the nominal voltage, capacity, etc. We also created TDs to represent subsystems. For example, the storage subsystem consisting of a charge controller and batteries was represented by a single TD which contained properties like charging status, minimum, maximum, and average battery voltage, etc. WoT scripting API [7] was used to define the semantics of such aggregation. The generated TDs were stored in a repository program based on the ThingWeb Directory project [8].

4.2 Fault Detection and Optimization Programs

We used Node-Red[9] to develop several fault detection and load optimization programs (see Figure 4(b)). The low-code programming environment was chosen because its possibility to use visual composition of programs using pre-defined functions in a library lowers the software programming skills required (considering the cost-consciousness in our use case, this is highly relevant). The programs were created solely based on the domain knowledge captured in the ontology (i.e., the taxonomy and relationships). A function available in the library encapsulated the access to the knowledge graph by internally formulating the SPARQL statement based on simplified input conditions. To facilitate easy construction of the semantic query, we incorporated Spartnatural [10], a visual query builder that the domain expert can use without having to know the SPARQL syntax. Using the visual query builder, the domain expert could easily construct queries like "lead-acid batteries with 12V nominal voltage that are charged using a PWM controller". As an example of programs built based on semantic descriptions, consider the evaluation of the rule, which states that the charging current of a battery should not be outside the limits defined in the product specification. First, the program needs to find the batteries in the system and then the specification of each of the batteries. In the next step, the program needs to retrieve the historical data of the relevant current sensors logged in the IoT infrastructure, Finally, a mathematical expression is applied to evaluate the sensor data. Figure 4(b) shows the implementation of this program

[6]https://rml.io/specs/rml/

[7]https://www.w3.org/TR/wot-scripting-api/

[8]https://github.com/thingweb/thingweb-directory

[9]https://nodered.org/

[10]https://sparnatural.eu/

using just three *nodes*. The first node specializes in querying the knowledge graph for components and retrieving and returning the TDs of the components, which is then fed to the second node where the required property values are fetched. The TD contains the description of the components – in the case of our example, the TD contains the description of the batteries (including their min and max charging current) and the description of the property interaction for retrieving the charging current. In the final node, the program iterates through the TDs and accesses the metadata (allowed min and max charging current) and the run time values using property interactions.

Like the fault detection program, we developed a load optimization program that accesses the TDs of the batteries, loads, and weather data (all based on semantic queries). It then evaluates the amount of charge in the batteries. Then, considering the load characteristics and the sunshine forecast, it outputs the potential duration that each load can be operated.

We made synthetic changes in our test infrastructure to verify if the programs could be applied to installations of different system configurations and component specifications. Configuration changes included the addition/removal of PV modules to the array, changing between parallel and series circuits, addition/removal of batteries, and changing the charge control function and its parameters. Similarly, product configurations like PV module voltage and battery capacity were changed. The impact of changing the IoT infrastructure or the communication API for obtaining data point values was also examined.

5 Results

Variations ->	**Components**	**Design**	**Controls**
Charging fault	Yes	Mostly	Yes
Battery degradation	Yes	NA	Yes
Diode fault	Yes	Yes	NA
Ground fault	NA	Yes	NA
Shading	Yes	Partly	NA
Dust coating	Yes	Partly	NA
Maximizing usage	Yes	Partly	Yes
Battery optimization	Yes	Yes	Yes

Table 2: Results showing the adaptivity of the fault detection and optimization programs for various use cases in response to changes in component specifications, design, and control functions. In all cases, the programs adapted to change in IoT service.

The ontology which we created was able to model the deployment and configuration of the PV systems used in our evaluation. In addition, the ontology was used to model other sample installations of medium sized PV systems. The ontology will be made available open-source so that contributions can be made to cover more wider variances in PV systems.

Apart from enabling description of small scale PV installations, the ontology contains tags that can be embedded into data point descriptions in the IoT stack. These tags

The use of the knowledge graph to assist fault detection and operation optimization programs showed that the programs could refer to domain-level terms without being bound to installation-specific features. The concepts in the ontology need to support description of fault rules in a manner that can adapt to variances in components, design, and control functions. This was validated by creating fault rules for some well-known cases like charging fault or battery degradation. It was found that in most cases the ontology supported the use cases (see Table 2 for more details). As a result, faults like battery overcharging and charge controller inefficiency were detected using the same set of rules on different installation configurations. The availability of semantic descriptions of energy generation, storage, and load subsystems helped us design robust load scheduling programs. For example, using the description of the load characteristics of equipment like irrigation pumps, cold-storage refrigeration units, and lights, an optimum usage schedule could be suggested while considering the storage status and sunshine prediction.

However, in some cases, the ontology lacked concepts to model the low-level physical phenomena required to distinguish faults. For example, detecting under-performance due to shading or dust coating requires modeling power characteristics based on the time of day. Similarly, maximizing the usage of available energy requires an additional mechanism to model the temporal characteristics of the load, concepts that are missing in our ontology (we consider constant loads). Some of these shortcomings can potentially be addressed in a more scalable manner through system-aware machine-learning techniques.

The openness of our ontology is enabled through its bridging to other known ontologies. This was verified by querying using available terms in the SSN, SOSA, and Brick ontologies. This opens up the possibility to integrate concepts such as building facade orientation to determine optimum PV performance. To address the farming use case, the linking to irrigation requirements allowed us to determine the load scheduling of PV-powered pumps.

The ontology which we developed for PV systems therefore helps in describing small scale PV installations, integrating data obtained from them in IoT stack while using semantic annotations, and finally, enables creation of programs that are decoupled from installation specifics.

6 Conclusion

We have shown that by integrating domain knowledge into IoT systems using semantic Web technologies, we can improve the performance and usage of small-scale PV installations. In the case of using PV energy in farming in remote areas, this would have a direct impact on the livelihood of the farmers. Our approach emphasizes the potential and importance of the ground-up integration of system knowledge in IoT, as this would enable the development of reusable software programs. We hope that this would open up doors for other research avenues like system-aware machine learning techniques in optimizing renewable energy sources or research into IoT architecture which considers system knowledge as an essential resource in addition to process data.

References

[1] Kais AbdulMawjood, Shady S Refaat, and Walid G Morsi. 2018. Detection and prediction of faults in photovoltaic arrays: A review. In *2018 IEEE 12th International Conference on Compatibility, Power Electronics and Power Engineering (CPE-POWERENG 2018)*. IEEE, 1–8.

[2] Ala Al-Fuqaha, Mohsen Guizani, Mehdi Mohammadi, Mohammed Aledhari, and Moussa Ayyash. 2015. Internet of things: A survey on enabling technologies, protocols, and applications. *IEEE communications surveys & tutorials* 17, 4 (2015), 2347–2376.

[3] Arka A Bhattacharya, Dezhi Hong, David Culler, Jorge Ortiz, Kamin Whitehouse, and Eugene Wu. 2015. Automated metadata construction to support portable building applications. In *Proceedings of the 2nd ACM International Conference on Embedded Systems for Energy-Efficient Built Environments*. 3–12.

[4] Dario Bonino and Fulvio Corno. 2008. Dogont-ontology modeling for intelligent domotic environments. In *International Semantic Web Conference*. Springer, 790–803.

[5] Pim Borst, Hans Akkermans, and Jan Top. 1997. Engineering ontologies. *International journal of human-computer studies* 46, 2-3 (1997), 365–406.

[6] Youssef Cheddadi, Hafsa Cheddadi, Fatima Cheddadi, Fatima Errahimi, and Najia Es-sbai. 2020. Design and implementation of an intelligent low-cost IoT solution for energy monitoring of photovoltaic stations. *SN Applied Sciences* 2, 7 (2020), 1–11.

[7] Simon Clark, Francesca L Bleken, Simon Stier, Eibar Flores, Casper Welzel Andersen, Marek Marcinek, Anna Szczesna-Chrzan, Miran Gaberscek, M Rosa Palacin, Martin Uhrin, et al. 2022. Toward a unified description of battery data. *Advanced Energy Materials* 12, 17 (2022), 2102702.

[8] Aravind Devanand, Gourab Karmakar, Nenad Krdzavac, Rémy Rigo-Mariani, YS Foo Eddy, Iftekhar A Karimi, and Markus Kraft. 2020. OntoPowSys: A power system ontology for cross domain interactions in an eco industrial park. *Energy and AI* 1 (2020), 100008.

[9] Joshuva Arockia Dhanraj, Ali Mostafaeipour, Karthikeyan Velmurugan, Kuaanan Techato, Prem Kumar Chaurasiya, Jenoris Muthiya Solomon, Anitha Gopalan, and Khamphe Phoungthong. 2021. An effective evaluation on fault detection in solar panels. *Energies* 14, 22 (2021), 7770.

[10] SolarPower Europe. 2020. Global market outlook for solar power/2020–2024. *Solar Power Europe: Brussels, Belgium* (2020).

[11] Giorgos S Georgiou, Paul Christodoulides, and Soteris A Kalogirou. 2020. Optimizing the energy storage schedule of a battery in a PV grid-connected nZEB using linear programming. *Energy* 208 (2020), 118177.

[12] Armin Haller, Krzysztof Janowicz, Simon JD Cox, Maxime Lefrançois, Kerry Taylor, Danh Le Phuoc, Joshua Lieberman, Raúl García-Castro, Rob Atkinson, and Claus Stadler. 2019. The modular SSN ontology: A joint W3C and OGC standard specifying the semantics of sensors, observations, sampling, and actuation. *Semantic Web* 10, 1 (2019), 9–32.

[13] Abhishek Jain, Arunabha Ghosh, and Sanjana Chhabra. 2021. Powering Livelihoods Globally through Clean Energy. *GCF-CEEW Report* (2021).

[14] Imran Khan. 2020. Impacts of energy decentralization viewed through the lens of the energy cultures framework: Solar home systems in the developing economies. *Renewable and Sustainable Energy Reviews* 119 (2020), 109576. https://doi.org/10.1016/j.rser.2019.109576

[15] Farhad Khosrojerdi, Stéphane Gagnon, and Raul Valverde. 2021. Proposing an Ontology Model for Planning Photovoltaic Systems. *Machine Learning and Knowledge Extraction* 3, 3 (2021), 582–600.

[16] Yoshinobu Kitamura and Riichiro Mizoguchi. 2004. Ontology-based systematization of functional knowledge. *Journal of Engineering design* 15, 4 (2004), 327–351.

[17] Ascension Lopez-Vargas, Manuel Fuentes, and Marta Vivar. 2018. IoT application for real-time monitoring of solar home systems based on Arduino™ with 3G connectivity. *IEEE Sensors Journal* 19, 2 (2018), 679–691.

[18] Bertrand Meyer. 1997. *Object-oriented software construction*. Vol. 2. Prentice hall Englewood Cliffs.

[19] FDJ Nieuwenhout, A Van Dijk, VAP Van Dijk, D Hirsch, PE Lasschuit, G Van Roekel, H Arriaza, M Hankins, BD Sharma, and H Wade. 2000. Monitoring and evaluation of solar home systems. *Amsterdam: Netherlands Energy Research Foundation ECN* (2000).

[20] Ganesh Ramanathan and Maria Husmann. 2022. Semantic description of equipment and its controls in building automation systems. In *European Semantic Web Conference*. Springer, 307–310.

[21] M Sabbaghpur Arani and Maryam A Hejazi. 2016. The comprehensive study of electrical faults in PV arrays. *Journal of Electrical and Computer Engineering* 2016 (2016).

[22] Mohammed Samdani Shaik, Dipam Shah, Raghuram Chetty, and Rahul R Marathe. 2020. A LoRaWAN based open source IOT solution for monitoring rural electrification policy. In *2020 International Conference on COMmunication Systems & NETworkS (COMSNETS)*. IEEE, 888–890.

[23] W3C. 2020. *Web of Things (WoT) Architecture*. Technical Report. W3C. https://www.w3.org/TR/2020/REC-wot-architecture-20200409/

[24] W3C. 2020. *Web of Things (WoT) Thing Description*. Technical Report. W3C. https://www.w3.org/TR/2020/REC-wot-thing-description-20200409/

[25] Rita Zgheib, Emmanuel Conchon, and Rémi Bastide. 2017. Engineering IoT healthcare applications: towards a semantic data driven sustainable architecture. In *eHealth 360°*. Springer, 407–418.

[26] Ye Zhao, Ling Yang, Brad Lehman, Jean-François de Palma, Jerry Mosesian, and Robert Lyons. 2012. Decision tree-based fault detection and classification in solar photovoltaic arrays. In *2012 Twenty-Seventh Annual IEEE Applied Power Electronics Conference and Exposition (APEC)*. IEEE, 93–99.