Automated and real-time situational knowledge acquisition and representation in smart grids

Markus Stocker

University of Eastern Finland,
P.O. Box 1627, 70211 Kuopio, Finland
markus.stocker@uef.fi

Abstract. Knowledge engineering and automated sensor data processing and analysis are of key importance to smart grids. First, smart grids present a conceptual domain. Concepts, relations among them, and instances can be formalized both for knowledge about a grid, such as its elements and structure, and for knowledge sensed by its smart components, such as the optimal time to recharge a vehicle. Second, smart grids present a domain that is rich in sensors and sensor data. The multitude of sensors generate a large amount of heterogeneous data. In this paper I discuss a system that has both knowledge engineering and sensor data processing and analysis as its core aims. More specifically, the system aims at automated, near real-time, acquisition of situational knowledge from heterogeneous sensor data and its automated and formal representation in ontology. We present the overall architecture of the system and discuss the materials and methods that are necessary for its functioning in details. To showcase the application of the system we draw two domain use cases and discuss how each may benefit from the application of the presented system.

Key words: Smart grids; knowledge engineering; sensor data; knowledge acquisition; knowledge representation; time series

1 Introduction

In the past decade, sensor measurement has seen considerable advancement in low-cost, low-power, small-size, wireless technology as well as communication protocols, algorithms, and programming models [1, 7, 47, 51]. Similarly, sensor data management, processing, and query has received attention with the development of database systems optimized for streamed data [9, 8, 27, 28, 18, 11]. In addition, semantic technologies have been adopted for the semantic description of sensors, sensor networks, and sensor data [40, 13, 2, 15, 26, 35]. However, despite these advancements, managing, processing, and making sense of sensor data is an ongoing challenge [5, 21, 48, 4].

In this paper, I focus on the challenge of "making sense" of sensor data, specifically for the domain of smart grids. The concept of SmartGrids was developed in 2006 by the European Technology Platform (ETP) for Smart Grids.[1]

[1] http://www.smartgrids.eu/web/node/81

According to the SmartGrids ETP, a smart grid concerns "an electricity network that can intelligently integrate the actions of all users connected to it – generators, consumers and those that do both – in order to efficiently deliver sustainable, economic and secure electricity supplies." To do so, "a smart grid employs innovative products and services together with intelligent monitoring, control, communication, and self-healing technologies in order to: (1) Better facilitate the connection and operation of generators of all sizes and technologies; (2) Allow consumers to play a part in optimising the operation of the system; (3) Provide consumers with greater information and options for choice of supply; (4) Significantly reduce the environmental impact of the whole electricity supply system; (5) Maintain or even improve the existing high levels of system reliability, quality and security of supply; (6) Maintain and improve the existing services efficiently; (7) Foster market integration towards an European integrated market."

Sensors will be of great importance to smart grids [24]. Indeed, due to the large amount of sensor data, the domain is such that the data has to be processed in a continuous manner, ideally in near real-time. The aim of such processing is the "detection/estimation of some events of interest" [24]. I will present and discuss a system that supports the real-time representation of situational knowledge, in particular events of interest, in smart grids. The challenge of such a system is determined by the considerable gap that exists between low-level sensor measurement data and high-level conceptual terminology humans employ to describe such events of interest. The presented system aims at reducing this gap. Hence, it has the potential to contribute, at least, towards the points 2 and 3 in the above-listed aims, given that consumers require information in high-level domain terminology.

A main component of the presented system are semantic technologies, in particular ontology (knowledge base). An ontology, defined as an explicit specification of a conceptualization [19], allows for formal representation of domain knowledge, meaning the concepts of some area of interest and relations that hold among them. As such, ontologies seem to be ideal technologies to represent domain knowledge acquired from sensor data. Using ontology, the semantics of domain terminology is formally and explicitly represented outside any one system, and its particular implementation. Thus, such terminology becomes reusable and the interoperability of heterogeneous systems that commit to ontologies is improved [33, 49]. For instance, according to [30] "one of the problems common to the management of central control facilities is the fact that any equipment changes to a substation or power plant must be described and entered manually into the central computer system's database." Self-describing equipment, where the description is in the form of metadata that uses domain terminology defined in ontology, could significantly simplify the automatic registration of such heterogeneous systems, and increase the interoperability among its parts.

The presented architecture also tackles the problem of "too much data and not enough knowledge," [40] specifically for the domain of smart grids. Given the number of sensing devices that are expected in a smart grid, the data such a grid

generates is expected to be massive. We, thus, may likely run into a situation of "too much data" and yet the data may provide little explicit knowledge, in an automated manner. We underscore the importance of automation of both knowledge acquisition *and* representation because the scale of the data severely limits manual knowledge acquisition and because of the requirement that knowledge should be available in near real-time. The presented system aims at abstraction from sensor data to provide more near real-time situational knowledge that is automatically acquired and represented, for the domain of smart grids.

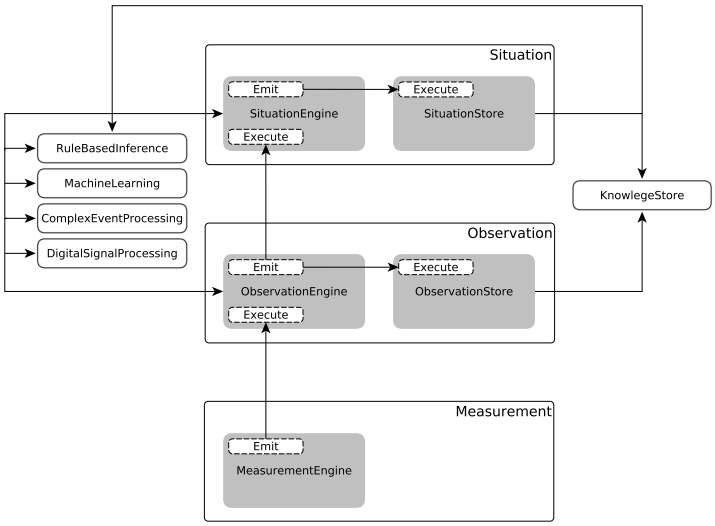

Fig. 1. System architecture for automated (near) real-time acquisition and representation of situational knowledge acquired from sensor data showing the three layers of measurement, observation, and situation as well as the main components and modules, and their interactions.

2 Architecture

In the following I present an architecture of a system for automated and (near) real-time situational knowledge acquisition and representation in smart grids. Figure 1 provides an overview of the system architecture.

The architecture builds on top of a distributed real-time computation system, namely Storm.[2] Storm supports real-time processing of streams of data. It allows for the building of so-called topologies, i.e. graphs of computation. Nodes of such a graph consist of processing logic. Edges, connecting nodes, indicate the flow of

[2] http://storm-project.net

data, i.e. streams (sequence) of tuples. Storm topology nodes can either be so-called "spouts" and "bolts." Nodes implement application logic, and spouts are sources of streams. Sensing devices can, thus, be understood as Storm spouts in a topology. In contrast, a bolt "consumes any number of input streams, does some processing, and possibly emits new streams." Bolts are, thus, Storm elements implementing the components in the layers of the presented architecture. Overall, a topology transforms streams into other streams. Specifically, in the presented system, Storm transforms streams of tuples for measurement data into streams of tuples for the information elements of knowledge about situations.

The architecture consists of three layers, namely the measurement, the observation, and the situation layers. Layers build on top of each other. Each layer consists of components. Components form a Storm topology; indeed, they correspond to nodes of a topology. Components may hold associations to modules which perform certain computations. Examples of components are measurement engine, observation engine, observation store, situation engine, and situation store. Examples of modules are digital signal processing, machine learning, complex event processing, or (rule-based) inference. Hence, modules perform the work that is coordinated by components, which are nodes of a Storm topology that communicate their input and output over edges of the topology graph.

3 Materials and methods

In this section, I will present the key materials and methods that are relevant to the architecture of the discussed system. Broadly speaking, we assume the existence of a sensor network and computer infrastructure, including hardware and software, to process sensor data and to represent acquire knowledge.

A smart grid consists of a large number of devices, each of which performs a specific function. Examples for sensing devices include Automated Meter Infrastructure (AMI, also known as "smart meter"), sensors for basic measurements (e.g. voltage, current, storage, and continuity sensing), atmospheric conditions (e.g. temperature, wind, moisture, solar radiation), distributed generation sensors (for load balancing). Also household appliances, electric vehicles, solar panels, batteries, etc. are devices of a smart grid. Naturally, computer infrastructure including hardware, communication links, as well as software, are also elements of a smart grid. Hence, the heterogeneity of devices, software, and systems is huge.

Several computer language specifications and software are employed in this work. In particular, the presented architecture adopts the Web Ontology Language (OWL) [50] and Resource Description Framework (RDF) [29] Schema ontology languages; the Semantic Web Rule Language (SWRL) [22]; software implementations of knowledge bases and reasoning engines such as Pellet [34] and HermiT [39]; RDF databases and APIs such as Apache Jena [10] and Star-

dog[3]; RDF and OWL authoring tools such as Protégé.[4] The architecture builds on Storm.[5] The system is implemented in Java.

Data retrieval and parsing. Sensing devices support a vast array of protocols for the exchange and communication of data. For instance, ANSI C12.18, IEC 61107, IEC 62056, Universal Metering Interface (UMI), and the Open Smart Grid Protocol (OSGP) are protocols used in data exchange with meters. The type of data exchanged and the communication protocols vary. For instance IEC 61107 exchanges ASCII encoded data using a serial port. TCP/IP technology is growing in relevance.[6] Reviewing data exchange protocols implemented in smart meters and data communication protocols such as TCP/IP is beyond the scope of this paper. The core message is that there exist a wide array of protocols smart grid sensing devices use to encode and communicate data.

The landscape of protocols means that computer systems that build on smart grid sensing devices are required to face considerable heterogeneity with respect to retrieval and parsing of data from such devices. While tedious, it is, however, a mere engineering problem to develop interfaces between systems, including in a system for real-time situational knowledge acquisition and representation discussed here. Such a system is required to implement interfaces to other systems such as sensing devices, databases, or stream processing systems. Implementations for the interfaces need to include the logic necessary to continuously retrieve and parse data, and translate it to meet the system's format.

In our architecture data retrieval and parsing is performed by the measurement engine component of the measurement layer. The component may make use of specific modules to retrieve and parse data. A measurement engine component emits measurements onto one or multiple streams.

Digital signal processing. Digital signal processing (DSP) techniques [37] may be of use to enhance the signal of the measured environmental property or to translate the signal into a pattern. There exists an array of DSP techniques, for instance filters and the Fourier transform. A bandpass filter can be used to enhance frequencies withing a certain band, and to suppress frequencies outside the specified band. Such a filter can be used to enhance the signal and suppress noise, for instance if it is known that the signal of interest lies within a certain band frequency. Fourier transform is an algorithm that translates a signal from its time-domain to its frequency-domain. The algorithm has many purposes. For instance, the frequency-domain pattern of a signal can be classified using machine learning.

In the presented architecture, DSP is implemented in modules associated to components. Naturally, it is the knowledge acquisition task that determines whether such a module is required for a particular problem. Modules can be

[3] http://www.stardog.com
[4] http://protege.stanford.edu/
[5] http://storm-project.net/
[6] http://en.wikipedia.org/wiki/Smart_meter

associated to components of any layer. For instance, if unprocessed observations are to be persisted by the system then DSP may be applied by components at the observation or situation layer. However, if, e.g., a filtered signal is to be persisted by the observation layer or if observations are not to be persisted in first place, then DSP may be applied by components at the measurement layer. Doing so may considerably reduce the number of tuples forwarded in the Storm topology. Note that it is possible to forward (DSP) processed data as well as unprocessed data, by defining two streams.

Machine learning. Machine learning (ML) [32] is an area of artificial intelligence in computer science. Just like DSP, it comes with array of techniques. Algorithms have been developed for purposes such as data clustering, classification, and regression. Algorithms are typically categorized by whether they are supervised or unsupervised. Multilayer Perceptron (MLP) feedforward artificial neural networks [20] are an example of a supervised machine learning algorithm. A MLP network consists of a set of neurons that form the input layer, one or more hidden layers, and an output layer. MLP is trained in a supervised manner using error back-propagation learning, which consists of a forward pass and a backward pass through the layers. In the forward pass, the signal resulting from the application of an input vector is propagated through the network in a forward direction and the actual response of the network at the output layer is recorded. In the backward pass, the recorded response of the network at the output layer is subtracted from a desired response, i.e. the label, to produce an error signal, which is propagated through the network in a backward direction. In this pass the network is adjusted in order to align the actual response with the desired response. In supervised learning, a set of input vectors with associated labels is used to train (calibrate) the network. Once trained, the network can be used to classify input vectors for which the label is not known.

As for DSP, in the presented architecture, ML is implemented in modules associated to components. It is the knowledge acquisition task that determines whether such a module is required for a particular problem. Modules for machine learning tend to be associated to components at the situation layer because machine learning is typically used to acquire knowledge from observations.

Knowledge representation. Knowledge representation and reasoning [3] is an area of artificial intelligence in computer science. An ontology, defined as an explicit specification of a conceptualization [19], allows for formal representation of domain knowledge, meaning the concepts of some area of interest and relations that hold among them. The Web Ontology Language [50] is today a de-facto standard ontology language. We adopt this language in the presented architecture and in the following brief presentation of ontology and its language constructs.

A concept of interest to our domain is the class of AMI sensing devices (smart meters). A physical AMI installed at a residential house is modeled in the ontology as an individual, instance of the concept for AMI. Formally, if `ami3345` is a physical AMI installed at a residential house and `AMISD` the concept for AMI

Listing 1.1. Overview of example terminological and assertional axioms.

```
# Terminological axioms
ResidentialHouse ⊑ House
isInstalledAt ⊑ owl:ObjectProperty
hasSpatialLocation ⊑ owl:DatatypeProperty
AMISD ⊑ SensingDevice ⊓ ∃isInstalledAt.ResidentialHouse

# Assertional axioms
AMISD(ami3345)
ResidentialHouse(house1345)
isInstalledAt(ami3345, house1345)
hasSpatialLocation(ami3345, sl6578)
hasLongitude(sl6578, 502913.319)
```

sensing devices, then the concept assertion `AMISD(ami3345)` (assertional axiom) states that the individual `ami3345` is an instance of the concept `AMISD`, meaning that `ami3345` is in fact a `AMISD`.

Naturally, a physical device relates to spatial and temporal locations. We can model such knowledge in the ontology by relating an individual to other individuals or data values. Relations between individuals are called object properties. Relations between an individual and a data value are called datatype properties. Properties are also known as roles. The individual `ami3345` relates via the object property `isInstalledAt` to a physical residential house, e.g. `house1345`. If `ResidentialHouse` is the concept for the class of residential houses, then we can state via the concept assertion `ResidentialHouse(house1345)` that the individual house `house1345` is in fact a residential house. We may further state that a residential house is a type of house. This can be expressed via a subclass relation, formally `ResidentialHouse` $\sqsubseteq$ `House` (terminological axiom). We can, thus, state that the AMI `ami3345` is installed at the residential house `house1345`, formally using the role assertion `isInstalledAt(ami3345, house1345)` (assertional axiom). Furthermore, by means of the object property `hasSpatialLocation` we may relate the `ami3345` to a spatial location `sl6578`, which is an individual instance of the concept `SpatialLocation` representing the class of spatial locations. Formally, `hasSpatialLocation(ami3345, sl6578)`. A spatial location typically has values for latitude and longitude, perhaps also altitude. We can represent these by means of datatype properties that relate the spatial location with corresponding values for latitude and longitude, e.g. `hasLongitude(sl6578, 502913.319)`. Listing 1.1 provides an overview of the terminological and assertional axioms discussed here as examples. An ontology consists of terminological and assertional axioms. Terminological axioms form the so-called TBox of a knowledge base and assertional axioms form the so-called ABox of a knowledge base. Thus, a knowledge base is formed by a TBox and an ABox.

In the presented architecture, namely at the observation layer, we represent sensor observations using the Semantic Sensor Network (SSN) ontology [23, 12]. SSN is an "ontology for describing the capabilities of sensors, the act of sensing and the resulting observations" [12]. According to [12] the SSN ontology can be seen from the sensor, the observation, the system, and the feature and property perspectives. In this work, the SSN ontology serves as an upper ontology from which to extend to accommodate domain knowledge for what property is sensed by which sensing device, as well as observations and the result of sensing. At a more abstract level, namely at the situation layer, we employ the Situation Theory Ontology (STO) [25] to represent knowledge about real-world situations, acquired from observations. The STO captures the key aspects of the situation theory developed by Barwise and Perry [6] and extended by Devlin [14]. The theory relates to the work on situation awareness by Endsley [16, 17], as it encompasses most of the concepts discussed by Endsley [25].

The Situation Theory developed by Devlin [14], formalizes the semantics of situations by means of the expression $s \models \sigma$, read "s supports σ," meaning that the infon σ is "made factual" by the situation s. According to the definition by [14], the object $\ll R, a_i, \ldots, a_m, i \gg$ is a well-defined infon if R is an n-place relation and $a_1, \ldots, a_m$ $(m \leq n)$ are objects appropriate for the argument places $i_1, \ldots, i_m$ of R, and if the filling of argument places $i_1, \ldots, i_m$ is sufficient to satisfy the minimality conditions for R, and $i = 0, 1$ is the polarity. Minimality conditions "determine which particular groups of argument roles need to be filled in order to produce an infon" [14]. The polarity is the 'truth value' of the infon. If $i = 1$ then the objects $a_1, \ldots, a_m$ stand in the relation R; else the objects do not stand in the relation R. Parameters, denoted as $\dot{a}$, make reference to *arbitrary* objects of a given type. For instance, $\dot{l}$ and $\dot{t}$ typically denote parameters for arbitrary objects of type spatial location and temporal location, respectively. Anchors are a mechanism to assign values to parameters. Hence, the parameter $\dot{t}$ may anchor the value for the current time.

The SSN and STO are upper ontologies. They provide a base vocabulary from which we can extend to accommodate domain-specific terminology and semantics. For instance, we may extend from the SSN `SensingDevice` to accommodate the domain-specific class of AMI sensing devices, formally `AMISD` $\sqsubseteq$ `ssn:SensingDevice`. In Listing 1.1 we use a concept `SensingDevice`. We may now simply qualify it as being defined by the SSN ontology as `ssn:SensingDevice`. Further, we may state that any sensing device is a relevant individual in infons, formally `ssn:SensingDevice` $\sqsubseteq$ `sto:RelevantIndividual`. In fact, we may use a sensing device in an infon in order to express something about it, e.g. that it is in a malfunction state.

In our architecture, observations are semantically enriched measurements, consistent with the SSN ontology. Hence, the SSN ontology is used at the observation layer, more accurately by the observation engine component. Observations can be stored in a knowledge base and are typically forwarded to the situation layer. Situations are knowledge acquired from observations, for instance by means of machine learning. Rules, search algorithms, complex event process-

ing, or domain-specific heuristics may also be implemented in modules to acquire situational knowledge.

Reasoning. Ontology language constructs allow us to represent knowledge using symbols. Consisting of a terminological and an assertional box, a knowledge base relates in someways to a conventional relational database, which consists, too, of a schema and instance data. In fact, knowledge represented in a knowledge base can be represented in similar ways in a relational database. However, due to the formal and expressive semantics of the ontology language, grounded in Description Logics [3], we can perform automatic symbolic reasoning. Such reasoning is beyond the capabilities of standard relational database systems.

Reasoning can be classified in ontology reasoning and rule-based reasoning. In ontology reasoning we use a reasoner to make explicit assertions that are implicit to a knowledge base. For instance, in Listing 1.1 a reasoner will infer that `House(house1345)`, because `house1345` is explicitly stated to be a residential house and because residential houses are houses. In a complex knowledge base, such ontology reasoning can discover implicit knowledge that is unexpected to users. This is in particular true when knowledge from heterogeneous sources is integrated. In rule-based reasoning we use a reasoner to make explicit the assertions that follow from a given set of rules. Rules are of the form $p \rightarrow q$ where p is the antecedent, i.e. a conjunction of rule atoms, and q is the rule consequent. Upon the matching of a rule antecedent for some assertions of a knowledge base, the rule consequent is "fired," meaning that the rule consequent is added as an assertion to the knowledge base. A simple example for a rule is a test for a threshold. For instance, we may state a rule that triggers an alert when we have a situation in which demand is beyond some threshold. The Semantic Web Rule Language (SWRL) is a de-facto standard for the specification of rules in OWL ontologies and knowledge bases.

In our architecture, reasoning is typically performed over a knowledge base, i.e. represented observations or, more often, situations. Thus, first situations are acquired from observations and knowledge for situations is represented (persisted) in the knowledge base. Reasoning is then applied to make explicit knowledge that is implicit to observations or situations with respect to the knowledge base, i.e. its axioms or rules.

4 Use cases

In this section, I will present two use cases for real-time situational knowledge acquisition and representation in smart grids. The architecture presented here may be extended to support the discussed use cases. The first use case is adapted from a presentation by Smeaton [42]. Given data on electricity usage, e.g. from a smart meter, we use machine learning to classify the signal in order to acquire knowledge for situations of usage of different home appliances, e.g. an oven. Here we acquire knowledge for situations that are observable by smart grid sensing devices. In second use case we represent knowledge for situations of

malfunctions in a smart grid. Hence, in contrast to the first use case, here we represent knowledge for situations *about* the smart grid.

4.1 Home appliances

This first use case is adapted from a presentation by Smeaton [42]. The author briefly discusses the analysis of electrical power usage to classify household appliance usage. Home appliance signatures can be identified in power usage time series and can be classified accordingly to household appliance usage. The author mentions the use of machine learning, specifically Support Vector Machines (SVM), to build classifiers for devices. Classified devices include shower, vacuum cleaner, kettle, microwave, toaster, electric oven. The author also suggests that such classification can be used in smart bills or behaviour analysis. In the following I will briefly describe how an implementation of the presented architecture can be developed for such a use case.

Given a continuous stream of data for electrical power usage in a home, acquired through e.g. a smart meter, an implementation for the presented architecture would require extensions at the measurement and situation layer. Specifically, a measurement engine should be developed which continuously transforms the sensor data into measurements, in this case univariate values for electrical power usage over time. Measurements are then semantically enriched to observations. Observations may be persisted in the knowledge base. Other processing is possible at this stage, for instance we may flag an observation to be erroneous by using a rule, e.g. a faulty sensor reading. Observations are then forwarded to the situation layer where the situation engine component acquires and represents knowledge for situations of household appliance usage.

The situation engine component orchestrates various modules for this purpose. At a minimum we need a machine learning module, if the methods suggested by Smeaton are to be applied. We might also need some digital signal processing, for instance to detect the starting and ending of an event. Further, given a classified situation we will need to use a situation store component to persist the situation in the knowledge base.

Given these extensions to the presented architecture, we can thus deploy a distributed system for the acquisition and representation of knowledge about situations of household appliance usage, acquired from electrical power usage measurement. Such represented knowledge can be used, among other things, also for the aims suggested by Smeaton, namely in smart bills or behaviour analysis. In fact, the knowledge base can now be queried at the granularity of the symbolic knowledge that is stored. For instance, we may ask for situations in which the electric oven is used during the night or for longer than 15 minutes.

4.2 Fault detection

This second use case is adapted from a discussion over a few emails exchanged between myself and Ilkka Nikander (ABB) in August 2012. The aim here is to identify the location of faults in an electrical network based on Automated Meter

Reading (AMR) measurements. For this purpose, we use a (CIM) model of the electrical network and alarm messages generated by AMR meters. The location of a fault corresponds to the location of a fuse that is common to alarming meters in the network. Once a fault is located it is possible to dispatch a work group by using information for the location of such groups.

In our architecture, the (CIM) model of the electrical network is knowledge persisted in the knowledge base. Although CIM is maintained as a UML model there exist tools which can automatically generate an RDF Schema from the original CIM UML model [31]. Hence, it is possible to persist a CIM electrical network model in a standard RDF/OWL knowledge base. Alarms generated by AMR meters are XML/SOAP messages and these need to be processed at the measurement layer in our architecture. Naturally, it is debatable whether an alarm is strictly speaking a measurement. There are at least two alternatives here. If an alarm is understood to be a measurement then processing it at the measurement layer into observations and situations is appropriate. Alternatively, alarms may be understood as detected situations. Hence, they could be forwarded directly to the situation layer. A third variant could be to process actual AMR voltage measurements at the measurement layer and let the system trigger alarms, e.g. by implementing a rule or complex event processing at the observation layer. Observations could be persisted in the knowledge base and alarms are forwarded to the situation layer. At the situation layer, alarms become situations, i.e. events in time which represent an alarm state in a electrical network. Situations can be represented with infons, e.g. for the location, using the technologies above.

Once situations for faults are represented and persisted in the knowledge base higher level applications can make use of the acquired and represented symbolic knowledge. As suggested above, we could make use of knowledge about the location, and operating region, of work groups, which could be available in the same or different knowledge base. By integrating knowledge from one or more knowledge bases, it is possible to locate the optimal (i.e. closest available) work group to be dispatched for the detected fault. There are at least two techniques that allow us to infer the optimal work group: one is based on quantitative and the other on qualitative spatial reasoning. Quantitative spatial reasoning is essentially based on a GIS. In this case, GIS data for the work groups is available and the location of the fault is given by spatial coordinates. Inference of the optimal work group is, thus, a quantitative spatial reasoning problem, solvable in standard GIS. Such an approach requires a GIS system and cannot be directly executed by standard semantic technologies, given that these cannot reason over quantitative spatial data. The second approach is based on qualitative spatial reasoning. Here we may adopt the Region Connection Calculus [38] and a semantic reasoner with support for qualitative spatial reasoning, e.g. [46]. With this approach, the operating area of work groups is represented as RCC regions, a mere symbol in a knowledge base. The spatial information is represented in the knowledge base by means of qualitative relationships among regions, e.g. regions that are externally connected, overlapping, or proper parts (a region including

another region). Situations generated at the situation layer in our architecture will have to relate the location of the detected fault to a region representing the location in the knowledge base. This region relates to regions of work groups via qualitative spatial relationships. A semantic reasoner with qualitative spatial reasoning capabilities can now infer which work groups is optimal (i.e. closest and available) to the location of the fault in the electrical network (the region representing the location of the fault may be, e.g., a proper part of the region representing the location of the group).

5 Discussion

I have presented, with some granularity, the architecture of a system for real-time situational knowledge acquisition and representation using semantic technologies, for the special case of smart grids. The three layered architecture with measurement, observation, and situation layers building on top of each other to gradually abstract from sensor data to represented knowledge acquired from such data is generic and I showed how it can be adapted to meet the aims of knowledge acquisition tasks in a smart grid domain. While the architecture provides some functionality out of the box, such as persisting observations or situations or distributed processing, there are elements that require domain adaptations. Most notably, we need to implement adapters to smart grid sensing devices and we need to implement the knowledge acquisition logic, for instance by developing a machine learning module.

The main aim of this work, and the presented system, is to argue that there is a gap between low-level sensor measurements and high-level situational knowledge, the conceptual world humans inhabit. The aim of the presented system is to bridge this gap, let computer systems process the signal of environmental properties measured by means of sensing devices to knowledge of interest, such that it is represented using terminology that "make sense" to humans, as well as machines. It is computationally relatively straightforward to access a sensor measurement in a time series or iteratively process a time series. Indeed, a time series explicitly represents sensor measurements over time. In contrast, it is computationally more challenging to uncover knowledge about the environmental property for which the signal is measured by sensing devices. This is because such knowledge is not explicitly represented in a time series. In fact, such knowledge is implicit in a time series. Thus, it is straightforward to plot a time series and more challenging to acquire and represent knowledge conveyed by the same time series. However, the value of represented knowledge conveyed by a time series is greater to humans than the value of a plot for the same time series. In real-time domains with millions or billions of measurements such automatic knowledge acquisition and representation seems to be paramount. Eventually, such a system may contribute to more accurate and timely situational awareness.

The importance of the discussed system is also motivated by the obvious limitations of persisting billions or trillions of measurement data. For the Los

Angeles Department of Water and Power, the authors of [41] highlight how information systems for smart grid networks may be unable to "centrally store and manage all the data that is collected." Similarly, in [44] the authors have used three accelerometer sensing devices sampling at 2 kHz each to measure ground vibration. Such a sensor network generates a theoretical 6000 measurements per second. While such a volume may still be within the processing capability of a workstation, a sensor network consisting of three sensors is very small. While smart meters sample at considerable lower frequency, the potential of having million of devices installed in homes will generate measurement data volumes that are a challenge to centrally store, and manage, e.g. retrieve and process. In addition to smart meters, measurement data for all kinds of sensing devices will be necessary for a grid to be smart, such as for instance weather data, voltage, or the status of storage capacity in batteries of electric vehicles. Even if it is technically possible to persist measurements, it may not be useful to do so. In the use case described by Stocker *et al.* [44] much of the ground pavement vibration measurement is not interesting as it represents background vibration of road-pavement. Of interest is only vibration induced by vehicles on the road pavement, as this is the domain for which the sensor network as deployed.

There are a number of challenges in the presented architecture. Most importantly, there is currently little support for the implementation of knowledge acquisition task. In fact, such task need to be developed by programming in Java. This is tedious, costly, and error prone. It is, however, not immediately obvious how to support the implementation of knowledge acquisition tasks in more straightforward ways, not least because of the domain specificity of such tasks. Thus, research in such methods is part of future work at which we aim.

The benefits of represented situational knowledge are numerous. As I underscored above, it amounts to a considerable data compression, as symbolic knowledge requires much less disk space than millions of measurement values, especially if much of what is measured is not of interest. Further, symbolic situational knowledge aligned with domain terminology is more intuitive and makes more sense to humans, than numerical time series. This applies also to computational services that build on top of such a system for real-time situational knowledge acquisition and representation. In fact, computational services can now interact with the conceptual layer. The benefit of using ontologies is clear in that they allow us to formally and explicitly define the semantics of domain terminology externally to any system implementation. By committing to reuse terminology defined in an ontology the interoperability of systems is greater [33, 49].

Semantic web technologies also come with a number of interesting advantages. First, the specifications for ontology, rule, and query languages are open and de-facto W3C standards. Second, there is an active community around the technologies that keeps developing systems and software which implement the specifications. For instance, there exist a number of RDF databases and OWL reasoners. OWL reasoners also come with a number of advantages, such as automated ontology and rule-based reasoning, consistency checking, and incon-

sistency explanation services. RDF databases support storage and retrieval of knowledge using de-facto standard query languages, such as SPARQL [36].

The work presented here borrows considerably from related work by the author [45, 44, 43, 46]. In fact, what I presented here is an adaptation of this work to the domain of smart grids, presenting the architecture how it would have to be adapted to meet the aims of the two discussed use cases. In future work, the aim is to actually develop one or more use cases, not necessarily those discussed here. However, in order to do so it will be of paramount importance to be able to access to real-time smart grid sensor data, in order to perform real experiments.

6 Conclusions

I have presented and discussed in details the architecture, materials, and methods relevant to a system for the automated, near real-time, representation of situational knowledge acquired from sensor data in a state-of-the-art knowledge base using de-facto standard, open, and free knowledge representation languages and related systems. The architecture builds on top of a distributed system for real-time processing of data streams. I presented the layers, components, and modules of the architecture, their role and interactions. The materials and methods required for the implementation of the architecture in software are presented and discussed for how they are used in the architecture. In order to adapt the generic architecture to the domain of smart grids I have presented two use cases that are relevant to the domain. As shown, the presented architecture can be adapted to the needs of the use cases. In future work we aim at performing real experiments by implementing the presented use cases or other use cases that are of interest to the domain of smart grids. To do so I will need real sensor data, which was not provided for this work.

References

1. I.F. Akyildiz, Weilian Su, Y. Sankarasubramaniam, and E. Cayirci. A survey on sensor networks. *Communications Magazine, IEEE*, 40(8):102–114, August 2002.
2. S. Avancha, C. Patel, and A. Joshi. Ontology-driven adaptive sensor networks. In *Mobile and Ubiquitous Systems: Networking and Services, 2004. MOBIQUITOUS 2004. The First Annual International Conference on*, pages 194–202, August 2004.
3. F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider. *The Description Logic Handbook: Theory, Implementation and Applications*. Cambridge University Press, 2nd Edition, 2007.
4. M. Balazinska, A. Deshpande, M.J. Franklin, P.B. Gibbons, J. Gray, S. Nath, M. Hansen, M. Liebhold, A. Szalay, and V. Tao. Data Management in the Worldwide Sensor Web. *Pervasive Computing, IEEE*, 6(2):30–40, April-June 2007.
5. P. Barnaghi, F. Ganz, C. Henson, and A. Sheth. Computing Perception from Sensor Data. Technical report, knoesis.org, 2012.
6. J. Barwise and J. Perry. *Situations and attitudes*. Bradford books. MIT Press, 1983.

7. Archana Bharathidasan, Vijay An, and Sai Ponduru. Sensor Networks: An Overview. Technical report, Department of Computer Science, University of California, Davis, 2002.
8. Philippe Bonnet, Johannes Gehrke, and Praveen Seshadri. Towards Sensor Database Systems. In *Mobile Data Management*, volume 1987 of *LNCS*, pages 3–14. Springer Berlin / Heidelberg, 2001.
9. Don Carney, Uğur Çetintemel, Mitch Cherniack, Christian Convey, Sangdon Lee, Greg Seidman, Michael Stonebraker, Nesime Tatbul, and Stan Zdonik. Monitoring streams: a new class of data management applications. In *Proceedings of the 28th international conference on Very Large Data Bases*, VLDB '02, pages 215–226. VLDB Endowment, 2002.
10. Jeremy J. Carroll, Ian Dickinson, Chris Dollin, Dave Reynolds, Andy Seaborne, and Kevin Wilkinson. Jena: Implementing the Semantic Web Recommendations. Technical Report HPL-2003-146, HP Laboratories, Bristol, UK, 2003.
11. Sirish Chandrasekaran, Owen Cooper, Amol Deshpande, Michael J. Franklin, Joseph M. Hellerstein, Wei Hong, Sailesh Krishnamurthy, Samuel Madden, Vijayshankar Raman, Frederick Reiss, and Mehul A. Shah. TelegraphCQ: Continuous Dataflow Processing for an Uncertain World. In *CIDR*, 2003.
12. Michael Compton. What Now and Where Next for the W3C Semantic Sensor Networks Incubator Group Sensor Ontology. In *Proceedings of the 4th International Workshop on Semantic Sensor Networks 2011 (SSN11)*, pages 1–8. CEUR-WS, 2011.
13. Michael Compton, Corey Henson, Holger Neuhaus, Laurent Lefort, and Amit Sheth. A Survey of the Semantic Specification of Sensors. In *2nd International Workshop on Semantic Sensor Networks, at 8th International Semantic Web Conference*, 2009.
14. K.J. Devlin. *Logic and information*. Cambridge Univ Pr, 1995.
15. M. Eid, R. Liscano, and A. El Saddik. A Novel Ontology for Sensor Networks Data. In *Computational Intelligence for Measurement Systems and Applications, Proceedings of 2006 IEEE International Conference on*, pages 75–79, july 2006.
16. M.R. Endsley. Toward a theory of situation awareness in dynamic systems. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 37(1):32–64, 1995.
17. M.R. Endsley. Theoretical underpinnings of situation awareness: A critical review. *Situation awareness analysis and measurement*, pages 3–32, 2000.
18. Lukasz Golab and M. Tamer Özsu. Issues in data stream management. *SIGMOD Rec.*, 32:5–14, June 2003.
19. T.R. Gruber. A translation approach to portable ontology specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.
20. S.S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice Hall International Editions Series. Prentice Hall, 1999.
21. David J. Hill, Yong Liu, Luigi Marini, Rob Kooper, Alejandro Rodriguez, Joe Futrelle, Barbara S. Minsker, James Myers, and Terry McLaren. A virtual sensor system for user-generated, real-time environmental data products. *Environmental Modelling & Software*, 26(12):1710–1724, 2011.
22. Ian Horrocks, Peter F. Patel-Schneider, Harold Boley, Said Tabet, Benjamin Grosof, and Mike Dean. SWRL: A Semantic Web Rule Language Combining OWL and RuleML. Technical report, W3C, May 2004.
23. K. Janowicz and M. Compton. The stimulus-sensor-observation ontology design pattern and its integration into the semantic sensor network ontology. In *The 3rd International workshop on Semantic Sensor Networks*, pages 7–11, 2010.

24. William Kao. Sensor Devices and Sensor Network Applications for the Smart Grid/Smart Cities. SensorsCon 2012, March 2012.
25. Mieczyslaw M. Kokar, Christopher J. Matheus, and Kenneth Baclawski. Ontology-based situation awareness. *Inf. Fusion*, 10(1):83–98, January 2009.
26. Jie Liu and Feng Zhao. Towards semantic services for sensor-rich information systems. In *Broadband Networks, 2005. BroadNets 2005. 2nd International Conference on*, volume 2, pages 967–974, October 2005.
27. S. Madden and M.J. Franklin. Fjording the stream: an architecture for queries over streaming sensor data. In *Data Engineering, 2002. Proceedings. 18th International Conference on*, pages 555–566, 2002.
28. Samuel R. Madden, Michael J. Franklin, Joseph M. Hellerstein, and Wei Hong. TinyDB: an acquisitional query processing system for sensor networks. *ACM Trans. Database Syst.*, 30:122–173, March 2005.
29. Frank Manola and Eric Miller. RDF Primer. Technical Report W3C Recommendation, W3C, 2004.
30. S. Massoud Amin and B.F. Wollenberg. Toward a smart grid: power delivery for the 21st century. *Power and Energy Magazine, IEEE*, 3(5):34–41, sept.-oct. 2005.
31. A.W. McMorran. An Introduction to IEC 61970-301 & 61968-11: The Common Information Model. Technical report, Institute for Energy and Environment, Department of Electronic and Electrical Engineering, University of Strathclyde, Glasgow, UK, 2007.
32. T.M. Mitchell. *Machine learning*. McGraw-Hill Boston, MA:, 1997.
33. Leo Obrst. Ontologies for semantically interoperable systems. In *Proceedings of the twelfth international conference on Information and knowledge management*, CIKM '03, pages 366–369, New York, NY, USA, 2003. ACM.
34. Bijan Parsia and Evren Sirin. Pellet: An OWL DL Reasoner. In *Proceedings of the International Workshop on Description Logics*, 2004.
35. Danh L. Phuoc and Manfred Hauswirth. Linked Open Data in Sensor Data Mashups. In David D. Kerry Taylor, editor, *Proceedings of the 2nd International Workshop on Semantic Sensor Networks (SSN09) in conjunction with ISWC 2009*, volume 522. CEUR, 2009.
36. Eric Prud'hommeaux and Andy Seaborne. SPARQL Query Language for RDF. Technical Report W3C Recommendation, W3C, 2008.
37. L.R. Rabiner and B. Gold. *Theory and application of digital signal processing*. Prentice-Hall signal processing series. Prentice-Hall, 1975.
38. D.A. Randell, Z. Cui, and A.G. Cohn. A Spatial Logic based on Regions and Connections. In *Nebel, B., Rich, C., Swartout W. (eds.): Principles of Knowledge Representation and Reasoning. Morgan Kaufmann*, pages 165–176, San Mateo, CA, USA, 1992.
39. R. Shearer, B. Motik, and I. Horrocks. HermiT: A highly-efficient OWL reasoner. In *Proceedings of the 5th International Workshop on OWL: Experiences and Directions (OWLED 2008)*, pages 26–27, 2008.
40. A. Sheth, C. Henson, and S.S. Sahoo. Semantic Sensor Web. *Internet Computing, IEEE*, 12(4):78–83, July-August 2008.
41. Y. Simmhan, S. Aman, B. Cao, M. Giakkoupis, A. Kumbhare, Q. Zhou, D. Paul, C. Fern, A. Sharma, and V. Prasanna. An Informatics Approach to Demand Response Optimization in Smart Grids. Technical report, University of Southern California, 2011.
42. Alan F. Smeaton. Sensor Data Streams from Smart Metering. CIICT: Symposium on Energy Efficiency and ICT, July 2012.

43. Markus Stocker, Mauno Rönkkö, and Mikko Kolehmainen. Making Sense of Sensor Data Using Ontology: A Discussion for Residential Building Monitoring. In Lazaros Iliadis, Ilias Maglogiannis, Harris Papadopoulos, Kostas Karatzas, and Spyros Sioutas, editors, *Artificial Intelligence Applications and Innovations*, volume 382 of *IFIP Advances in Information and Communication Technology*, pages 341–350. Springer Boston, 2012. 10.1007/978-3-642-33412-2_35.
44. Markus Stocker, Mauno Rönkkö, and Mikko Kolehmainen. Making Sense of Sensor Data Using Ontology: A Discussion for Road Vehicle Classification. In R. Seppelt, A.A. Voinov, S. Lange, and D. Bankamp, editors, *2012 International Congress on Environmental Modelling and Software: Managing Resources of a Limited Planet: Pathways and Visions under Uncertainty, Sixth Biennial Meeting*, pages 2387–2394, Leipzig, Germany, July 2012. International Environmental Modelling and Software Society (iEMSs).
45. Markus Stocker, Mauno Rönkkö, Ferdinando Villa, and Mikko Kolehmainen. The Relevance of Measurement Data in Environmental Ontology Learning. In *Environmental Software Systems. Frameworks of eEnvironment*, volume 359 of *IFIP Advances in Information and Communication Technology*, pages 445–453. Springer Boston, 2011.
46. Markus Stocker and Evren Sirin. PelletSpatial: A Hybrid RCC-8 and RDF/OWL Reasoning and Query Engine. In Rinke Hoekstra and Peter F. Patel-Schneider, editors, *Proceedings of the 6th International Workshop on OWL: Experiences and Directions (OWLED 2009)*, volume 529, Chantilly, Virginia, USA, October 2009. CEUR Workshop Proceedings.
47. Ryo Sugihara and Rajesh K. Gupta. Programming models for sensor networks: A survey. *ACM Trans. Sen. Netw.*, 4:8:1–8:29, April 2008.
48. Jeff Tollefson. US launches eco-network. *Nature*, 476(135), August 2011.
49. Michael Uschold and Michael Gruninger. Ontologies and semantics for seamless connectivity. *SIGMOD Rec.*, 33(4):58–64, December 2004.
50. W3C OWL Working Group. OWL 2 Web Ontology Language. Technical Report W3C Recommendation, W3C, 2009.
51. Jennifer Yick, Biswanath Mukherjee, and Dipak Ghosal. Wireless sensor network survey. *Comput. Netw.*, 52:2292–2330, August 2008.