The Semantically Enabled Smart Grid

Andrew Crapo (crapo@research.ge.com)

GE Global Research, Niskayuna, NY

Xiaofeng Wang (wang.Xiaofeng@ge.com)

GE Energy, Melbourne, FL

John Lizzi (lizzi@research.ge.com)

GE Global Research, Niskayuna, NY

Ron Larson (ron.larson@ge.com)

GE Energy, Melbourne, FL

Keywords: Smart Grid, Semantic Web, Architecture, Model Driven

Abstract

To fully achieve the benefits of smart grid, a range of new software applications, components, and improvements to business processes will rely on information emanating from existing and new systems and data sources. These new smart software components will need to interpret business semantics in a common way in order to ensure that data can be exchanged and shared, and that business intelligent activities can be carried out in an efficient and cost effective manner. An open, and shared information model provides such common semantics. An architecture driven by this information model will allow for reduced integration costs, increased development efficiency, and increased overall system flexibility. It also allows for new application functionality not possible given traditional architectural approaches.

Application of semantic web technology presents several interesting questions: How can smart grid information models be made accessible to domain experts? How can smart grid applications leverage Semantic Web technology to reason in ways not possible with traditional information modeling techniques? What are the key challenges facing those looking to leverage Semantic Web technology in the smart grid domain? We present an integrated set of tools and technologies aimed at addressing such questions. To make semantic modeling accessible to domain experts, we have developed the Semantic Application Design Language (SADL), a controlled-English language with an associated authoring environment for building rich formal models and adding layers of domain-specific rules. Models are translated to OWL (Web Ontology Language) and application rules are translated to the Semantic Web Rule Language (SWRL) or to Jena Rules. Smart grid reasoners are then able to draw inferences both from the logical structure of the model and from the domain rules. The result is "documents that think"- documents that are computable. When situation-specific data is combined with the model, the output is the implications of the document for the situation.

1. INTRODUCTION

The focus of this paper is to highlight how open, shared, and semantic information models can be used to enable new smart grid capabilities. Not only will an open, shared, and semantic information model enable semantic interoperability among diverse smart grid components and systems, it will allow for the application of new capabilities not possible given traditional systems engineering approaches.

This paper is broken into two sections. The first section provides background information regarding the smart grid, associated standards, and Semantic Web technologies. The second section describes GE's vision for the smart grid architecture, its information model, and how that information model can enable new use cases and applications. We introduce the Semantic Application Design Language (SADL) and its associated authoring and execution environment, which can be used by subject matter experts (SMEs) to develop smart grid models and analytics based upon Semantic Web languages, standards, and open source tools.

2. BACKGROUND

2.1. Smart Grid Challenges

According to EPRI [1], the smart grid "refers to a modernization of the electricity delivery system so it monitors, protects and automatically optimizes the operation of its interconnected elements—from the central and distributed generator through the high-voltage transmission network and the distribution system, to industrial users and building automation systems, to energy storage installations and to end-use consumers and their thermostats, electric vehicles, appliances and other household devices." Realization of this vision will require "a two-way flow of electricity and information to create an automated, widely distributed energy delivery network. It incorporates into the grid the benefits of distributed computing and communications to deliver real-time information and enable the near-instantaneous balance of supply and demand at the device level." [Ibid] In other words, what separates the smart grid from today's grid is that the flow of information and its meaning—communication—will be as ubiquitous as is the flow of electricity in the current grid.

Today, the system that manages the transmission and distribution of power is subdivided into discrete subsystems, each one managing a subset of the overall solution. On the whole, these subsystems are treated individually, often from multiple vendors. A degree of point-to-point integration exists between some of these, typically to automate key business tasks such as data propagation, synchronization, planning, and configuration. In addition, as established Grid Operators reach retirement age, the replacement workforce is younger and less experienced in managing this somewhat disconnected system, and demand a more "intelligent" solution for managing the power system as a whole.

In order to fulfill the vision and meet the challenges described above, the various systems and subsystems of the smart grid need to be able to seamlessly interoperate. Interoperability and communication, whether it is between humans or software systems, begin with a shared, common language. A system's information architecture and associated information models provide system components with this shared, common, language. Not only does this common language enable information exchange, it also provides a foundation upon which new capabilities can be built.

2.2. Standards

The smart grid covers a wide range of business domains and functionalities. Standards provide common protocols, syntax, and data models that can be used by the various elements of the smart grid to work together. Leveraging standards also enables smart grid participants to drive commoditization of the components and/or of the component interfaces that are standardized. This in turn enables smart grid participants to focus on innovative applications, analytics, decision support facilities, etc., that create value for various customer segments.

The smart grid standards space spans multiple domains from electric power generation to information technology and emanate from a number of organizations: NERC, FERC, IEC, CIGRE, EPRI, W3C, NIST, and others. In the last decade or so, the power industry has made great strides in creating a common information model (CIM) to resolve semantic inconsistency issues. Today, the CIM is widely accepted by both vendors and customers globally. IEC 61970 [2] and IEC 61968 [3] series standards define data exchange specifications based on the CIM so that the interoperability between various systems and applications can be achieved. Using CIM as a semantic model to drive interface and data exchange design has been a key step in the standards space to better enable smart grid interoperability.

However, as the saying goes, the wonderful thing about standards is that there are so many of them! NIST originally identified 16 key standards or specifications for smart grid interoperability [4]. After public comment, the list increased to 31. EPRI's report to NIST increased the list to 77 [1]. NIST's conclusion is that "hundreds of standards will be required to build a safe, secure Smart Grid that is interoperable, end to end" [ibid]. End-to-end interoperability will require that these standards themselves are interoperable.

2.3. The Semantic Web

The Semantic Web envisions the future World Wide Web as a universal medium for the exchange of data, information, and knowledge. Where as HTML and XML provide shared syntaxes for information, the Semantic Web demands that shared semantics be achieved. To this end, the World Wide Web Consortium (W3C) has developed formal specifications such as RDF [5], RDFS [6], the Web Ontology Language (OWL) [7], and the Semantic Web Rule Language (SWRL) [8]. These enable a formal description of the concepts, terms, and relationships within a knowledge domain. These shared models, or ontologies, are the foundation upon which semantic search, communication, interoperability, and intelligence (reasoning) are built. Each of the Semantic Web standards mentioned above, in the order listed, provide a successively more expressive and powerful modeling capability. RDF and RDFS provide a weak ontology language. OWL provides a much more expressive language. SWRL allows domain-specific or business logic to be captured in the domain vocabulary defined by the underlying ontology.

Semantic technology is born of research and lessons learned in the past. Expectations were very high during the 1980's

and early 1990's that expert systems would revolutionize decision science. These expectations were largely unrealized, due in part to the inability of most expert system languages and systems to adequately model the domain of interest. During the same time period, object-oriented programming promised modularity and reusability at unprecedented levels. These promises were also not entirely realized. A third concurrent activity was in the area of classification and reasoning in a field called Decision Logics (DL) [9]. DL sought to build knowledge representations that were subsets of first or first and second order logic and which were provably computable.

The 1990 advent of the Web and its subsequent phenomenal success changed the way people access, exchange, and even use information. Today the size of the Web is estimated at 50 billion pages with over a trillion unique URLs. Tim Berners-Lee, credited with the invention of the Web, directs the W3C and leads that organization's efforts to realize the Semantic Web. Research into how to create and use semantic content is progressing rapidly and is worldwide. Semantic Web technology builds upon the past. For example, one flavor of OWL is OWL-DL [10], which incorporates Decision Logics to offer an expressivity that is also computable. The convergence and synergizing of these technologies in many ways overcomes the weaknesses found in each technology by itself. While there are many challenges, semantic technology and the Semantic Web offer great promise in ways that may be of enormous consequence for the smart grid.

In particular, interoperability and reasoning, both logical and domain-specific, are critically important to achieving the vision of a smart grid and are enabled by semantic models. Given two different standards or information schemas, each will have its own terminology or metadata (conceptual model, often implicit). By formalizing a conceptual model of the domain that encompasses the conceptual commitments of both standards or schemas, it becomes possible to explicitly capture the mapping between the two in terms of the shared conceptual model. This in turn permits information to flow bi-directionally between the two with mappings from one to the shared model and from the shared model to the second happening in an automated fashion. Additional standards or information schemas may likewise interoperate as long as their concepts are encompassed by the semantic model and the mapping of the new metadata to the shared model is understood and captured. An example of the kind of reasoning envisioned is described in Section 3.2.

3. FRAMEWORK

Smart grid is an initiative to apply innovative information technologies within the Transmission and Distribution (T&D) domain to solve key challenges and opportunities of the energy industry. GE has defined a system architecture and developed system requirements for the overall power system that enables the subsystems to operate in a coordinated, efficient, and reliable manner. This architecture will improve overall operational efficiency and address issues with aging assets, retiring workforce, and escalating reliability and efficiency (green) expectations. The power system will be capable of handling emergency conditions with "self-healing" actions, which will allow the utility industry to be more responsive to the energy market and overall utility needs.

The smart grid system capitalizes on advances in information and communications technologies, and will enable:

- Better grid performance
- Better support to the utility business processes
- Improved service delivery
- Improved customer service
- Self-healing to correct problems early
- Interactivity with consumers and market
- Optimization of resources
- Predictive capability to prevent emergencies
- Security

While smart grid will result in these benefits to nation-wide utility services, it will also allow the flexibility necessary to allow operational standards, protocols, and best practices to be adopted and implemented to meet unique local circumstances and needs. For example, not all smart grid customers will be starting from the same point. Many of them will be at different implementation points, will have different drivers, paths, and deployment rates as well as varying current technology sub-systems. Likewise, it is not necessary for smart grid to have identical technological capabilities nation-wide. Availability of some technologies (e.g., CAD, GIS mapping) will be determined based on local circumstances and needs. Where this paper refers to accepted standards and best practices, the intent is to refer to accepted standards and best practices employed by the supporting jurisdiction. It is expected that a variation in operational and technical practices and capabilities will exist across jurisdictions and smart grid will allow for the flexibility of integrating these various solutions via providing interface requirements and standards.

To fully achieve the benefits of smart grid, different applications, systems, and components need to interpret business semantics in a common way so that data can be exchanged and shared, and business intelligent activities can

be carried out. The purpose of the information architecture is to provide a common data and information definition that can be used throughout the smart grid system. The definition must support both integration and business intelligence. In addition, such a definition can be leveraged to enforce data integrity and business rules at the business service level.

3.1. Smart Grid Information Model

The smart grid information model makes use of the capabilities of Semantic Web technology to make the model modular and extensible. Concepts important to all model users are captured in the base model. This base model will be extended in stages, with each stage adding detail useful for a particular constituency and each extension narrowing the scope of applicability to a more specialized constituency. Extended models explicitly include models upon which they depend and are the contextual models used by various applications.

The information model draws heavily from CIM and other industry standards and allows mappings between concepts from different existing schemas or systems to be captured explicitly as part of the model. The approach also facilitates the intersection of multiple domains such as electrical distribution and communication devices since a contextual model at this intersection can import models at the correct level of granularity from both domains. As some standards are currently captured in UML, we will attempt to highlight similarities with and differences between UML models and ontologies in OWL or RDF/RDFS. RDF and RDFS do not have the same expressivity as UML. For example, cardinality restrictions are not possible as part of property definition in RDF/RDFS. Even with the expressive capability of OWL, direct translation between UML and OWL is not necessarily possible, especially if one desires to remain in OWL-DL [11].

Critical to successful capture of large, distributed models is the concept of an XML namespace [12]. Concept names need only be unique within a single namespace, allowing the same name to be used with different meanings or different names to be used for the same concept in different namespaces. For example, suppose that two models, the first in namespace *ns1* and the second in namespace *ns2*, define *Node* and *ConnectivityNode*, respectively, to mean exactly the same thing. Then we can say in OWL that *ns1:Node* is equivalent to *ns2:ConnectivityNode*. Similar constructs allow us to state that two classes are disjoint or that individual instances of things are known to be the same or are known to be different. This expressivity of OWL allows a formalization of the mapping from the concepts of one standard or system to those of another.

Using namespaces to identify sub models, OWL makes it easy to take an existing model and extend it for a specific purpose. For example, a core upper-level model of electrical distribution grids might be extended with additional concepts suitable for detailed network connectivity analysis. To do this, the more detailed model need only "import" the core upper-level model. A second model might import the same core model and extend it to support business decision processes. OWL permits explicit capture of version dependencies for imported models, making the model extension more robust than similar concepts in procedural languages such as Java.

As highlighted above, the purpose of the smart grid information model is to achieve the semantic consistency between applications and systems. Some concept definitions will be governed by business semantics while others will be governed by engineering and scientific principles. The smart grid information model captures both, combining in a modular and compatible way business concepts, equipment behaviors and structures, relationships, and rules. Existing industry standards such as CIM provide a starting point for smart grid conceptual models, but other concepts will be found to be important as the varying views of different stakeholders and different existing applications become part of the supported community. OWL permits these various extensions to be made without impacting either the shared core models or the other extension models.

Modularity and extensibility with formal mapping will enable development of different kinds of models while maintaining a shared semantics between them. We consider two kinds of models in more detail and then look at the implications of semantics for messages in a messaging environment.

Contextual Models

Information exchange is often seen in a particular business context that may represent a business process or a portion of a business process. The information exchanged in such a context will be at a particular level of granularity. For example, some contexts may not need as much detail as other contexts. A particular contextual model can import a model of the appropriate granularity. While the semantics of the model subset is consistent with the overall information model, the context may also have tighter restrictions on certain concepts (i.e. to convey business rules and data integrity existing in the applied business context) and so may need its own extensions. Contextual models can be seen as extensions of shared smart grid models using the OWL import capability described above. Applications that operate in a restricted context will utilize these context specific extension models, which in turn will build on concepts shared with other contexts. The concept of contextual models is well accepted and adopted by standard bodies like UN/CEFACT [13] and IEC [2, 3]. However,

implementation of contextual models in an ontological modeling environment may take a somewhat different form.

Implementation Model

Information models represented in UML are usually translated into a procedural language such as C++ or Java to become executable. The resulting code may be termed an implementation model. In contrast, an ontological model in OWL is usually loaded directly into a reasoner for computation. The addition of rules (SWRL) can further enhance the fidelity of an OWL model whereas such logic is normally added to a UML-derived model by adding additional code and/or by interfacing with a separate rule engine. Translation of OWL models into "compiled code" may be necessary for particular models used for particular purposes that may require enhanced performance, but means of performance enhancement other than translation, such as restricting the scope of the model or the kinds of reasoning performed, are generally preferable.

Models of physical equipment that operate at a fine-grained level of detail may lead to considerably more complexity than may be needed in other contexts such as business intelligence. As described above, the additional complexity needed at the physical level is added by extending shared models with additional semantic detail. Because the detailed model is based on the shared model, results from low-level model computations may be captured in higher-level constructs that can be immediately meaningful in other contexts that share the imported semantics.

Message Instances

Message instances are the actual data exchanged between applications or systems. When elements of a message are specified in terms of the concepts of a semantic model, the meaning encoded in a message by the sender becomes decipherable by a receiver. A message can be meaningfully exchanged between parties that share a model at some level of specificity as described above. Currently the standard for message format is XML conforming to a particular XML schema. However, the use of ontologies enables a more flexible messaging capability.

Generally speaking, two message categories are identified within the scope of smart grid.

Document-Oriented Messages

Document-oriented messages are designed for a specific business topic based on the smart grid information model. Business topics usually reflect a particular data exchange within business processes. For example, a trouble call ticket is generated when an outage is identified and needs to be fixed. The trouble call ticket contains all related information, which may involve multiple instances of business concepts and their relationships. Another example is meter-reading data, which represents meter data and reading type for a particular timeframe. A document-oriented message contains multiple instances of business concepts and the relationships between them related to a business topic. A document-oriented message uses the semantic concepts from a particular contextual model but may impose additional schematic or structural requirements upon the message. XML schema is one way of specifying these additional requirements.

Using meter reading as an example, Figure 1 (next page) illustrates how a document-oriented message is represented based on the smart grid information model. All classes and relationships are referenced from the smart grid information model, which together provide a view of meter reading messages.

The structural requirements of a document-oriented message can be specified in many ways depending on the implementation technologies. Using XML Schema as an example, the meter reading messages are represented as shown in Figure 2 (next page).

It is important to notice that business concept instances are explicitly represented with corresponding classes and relationships. In the implementation model, the business class and relationship names are used as tags to provide meaning for meter reading data.

Document-oriented messages are defined during the design phase. Their creation is informed by the relevant business topic, related business concepts and their relationships. The smart grid information model provides the metadata tags to design document-oriented message schemas.

Object-Oriented Messages

The smart grid information model provides an implementation-independent view of grid planning, operation, control, and management. With the smart grid model, it is possible for applications to access the information based on definitions of business concepts and be independent of application implementation details. This capability is implicit in the choice of OWL/RDF as the semantic modeling language. A model, including the instance data of a particular realization of a conceptual model, is a mathematical graph. Almost any desired sub-graph of the overall graph (model) can be extracted by matching a graph pattern specified in a graph query language. SPARQL is a W3C graph query language appropriate for sub-graph extraction [14].

SPARQL is to models expressed in OWL/RDF as SQL is to information stored in a relational database. Like SQL, SPARQL allows information to be retrieved from a model as a table of data—named columns with rows of information, each row representing one data set matching the query. The results of such a query might be formatted in

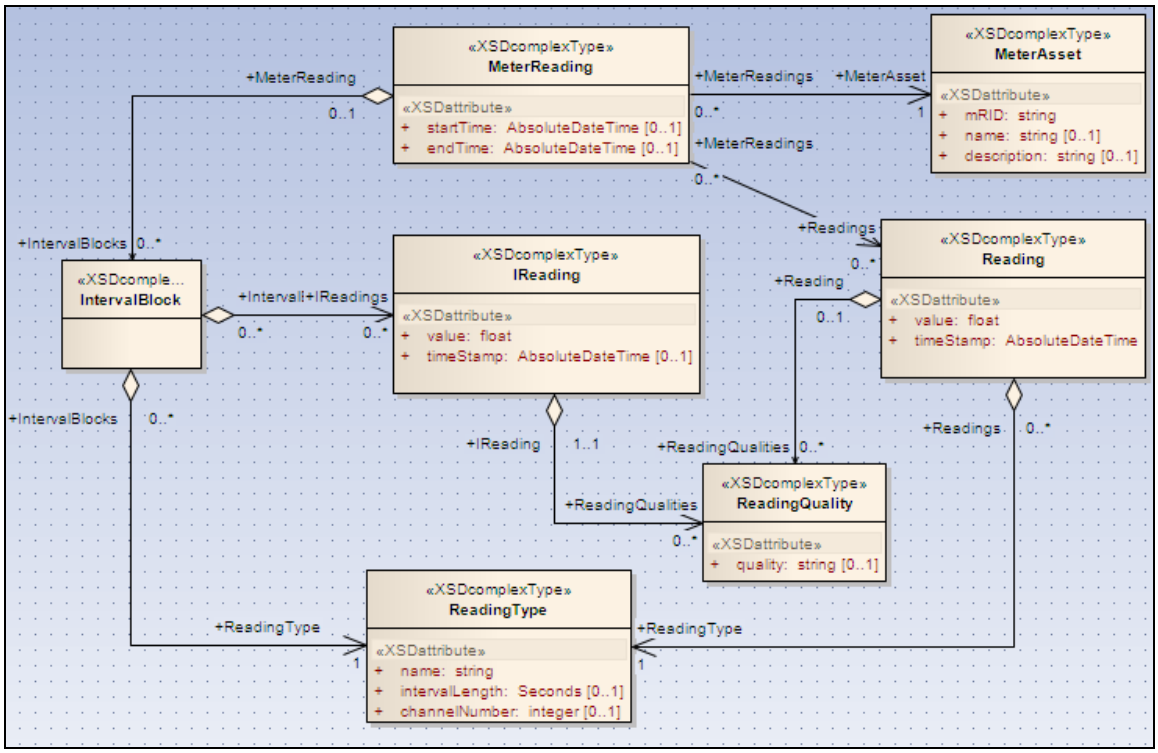

Figure 1: An Example Meter Reading Contextual Model for a Document Oriented Message

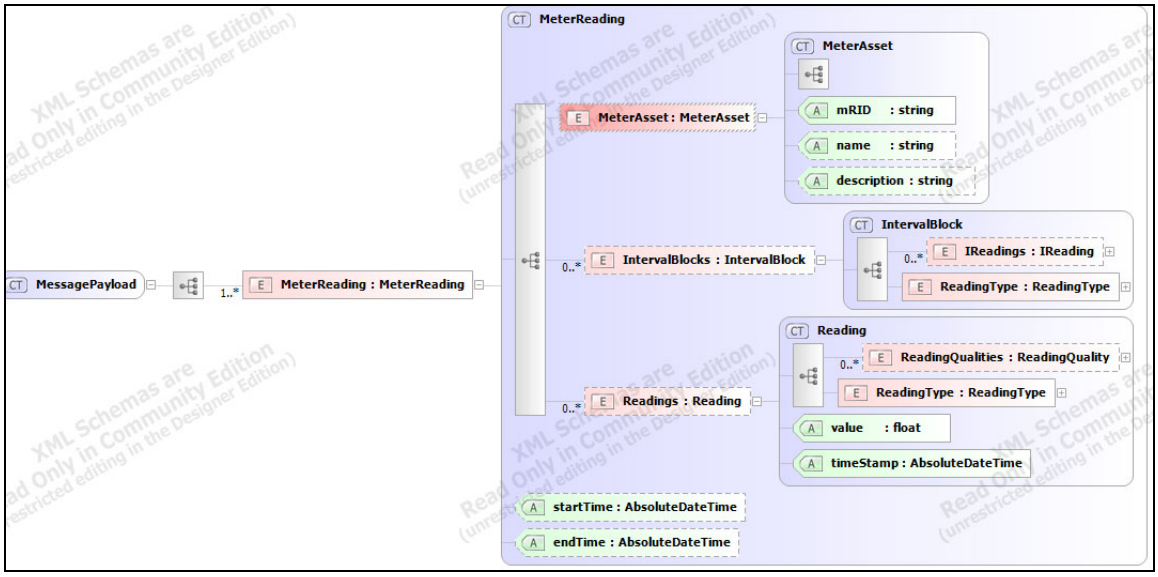

Figure 2: Example Meter Reading XML Schema

XML conforming to a particular XML schema. However, SPARQL also allows data to be returned as a graph. This is significant because the form of the data returned is. in this case. exactly the same form as the larger graph from which the data was extracted. This means that the formats for serialization will also be the same, e.g., RDF/XML, N-triple, etc. In other words, one can ask a question of an OWL model and get the answer back as an OWL model, albeit a potentially much smaller and more concise model for the desired purpose.

Such a query result is an object-oriented message. OWL graphs serve as object-oriented messages not just in response to model queries but also for transmission of

semantic information as messages between any sender and receiver. These messages can be used to:

1. Access model definition (meta-data)
2. Navigate a smart grid information model graph
3. Access business data
4. Update business data

Object-oriented messages are more dynamic than document-oriented messages since they provide flexibility to access and update business data based on the definitions of the semantic model. Extensions of to a base model or contextual model are immediately reflected in the legitimate content of object-oriented messages.

Information exchange, model query, and data and model maintenance are important parts of smart grid information management. These are well-supported by semantic technology. However, a well-defined semantic model has even more to offer. Semantic models captured in OWL can support rule-based intelligence on top of the logical inference implicit in the formal model, e.g.., inheritance. This rule-based intelligence can address model validation, regulatory compliance, model interaction, discovery of correlations between information objects, and business strategy. In the context of the eight-layer stack of the Gridwise Architecture Council (GWAC) [4], the use of semantic models enables implementations to move up from syntactic interoperability (layer 3, achievable with XML/XSD) to semantic understanding (layer 4) and above, which requires formal, logic-based models and the expression of rules in terms of model concepts.

3.2. The Semantic Application Design Language (SADL)

Semantic technology is still very young and the available tools are largely geared towards the ontologist rather than the SME. This is analogous to the early days of word processing when authoring software required a high degree of knowledge about and awareness of the formatting tags. (Who still learns to use LaTeX?) Recognizing the potential but aware that its realization would require the equivalent of WYSIWYG (What You See Is What You Get) authoring environments, we determined to try an experiment. Carefully choosing English-like phraseology to represent OWL constructs, we created a language called the Semantic Application Design Language (SADL) [15]. We used the Eclipse IDE Meta-Tooling Project (IMP) [16] both to define the language and to create an integrated development environment (IDE) for authoring documents (models) in the language. The IDE does token colorization by concept type (class, property, instance, etc.), shows error markers with explanation where an illegal or unrecognized phrasing is used, and is capable of much more including hyper linking concepts to their definitions, folding, phrase completion proposals, etc.

Once we had a functioning prototype, our litmus test was to build a small model in a domain and then put the model represented in the SADL language in front of a SME. If the expert asked what something meant we had failed, but if the expert immediately started talking about the model—whether it was correct in this point, should be extended in that area, etc.—without even thinking about the representation itself—we claimed some level of success. The results were promising and so SADL has evolved to be more complete with better editing support. Most of the functionality of SADL is released to Open Source as a project on SourceForge [15].

SADL supports all of the important constructs of RDF, RDFS, and OWL, including modularity. A SADL model can import and extend other models written in SADL or directly in OWL. Each model has its own namespace, and names need only be unique within that namespace. Rules are expressed in terms of defined concepts and use a formula-like syntax that is then converted to the esoteric expressions of SWRL or Jena rules. Figure 3 shows a simple SADL model defining some common shapes and a rule for computing the area of a rectangle.

```
uri "http://sadl.imp/shapes" version "$Revision: 1.2 $ Last
modified on $Date: 2009/03/06 14:37:54 $".

Shape is a top-level class, described by area with values of type
float.

Rectangle is a type of Shape, described by height with values of
type float, described by width with values of type float.

Rule AreaOfRectangle
 given
   x is any Rectangle
 then
   area of x = height of x * width of x .
```

Figure 3: Small Example of SADL

Figure 4 shows the definition of ConductionEquipment, extracted from CIM, in the SADL language. We have added a Boolean property called isolationCompliance to ConductionEquipment to simplify the conclusions of the three rules shown in Figure 5.

```
ConductingEquipment is a type of Equipment,
   described by fromConnect with values of type Terminal,
   described by toConnect with values of type Terminal,
   described by isolationCompliance with a single value of type
```

Figure 4: SADL definition of ConductingEquipment with added property isolationCompliance

```
uri "http://sadl.imp/GridInteropExample".

import "file://SelectedCim.sadl" as SelectedCim.

// Desired relationship of Breaker to Disconnector on each side:
//  Disconnector --toConnect--> Terminal --connectivityNode-->
//    ConnectivityNode --terminal-->  Terminal--fromConnect-->
//    Breaker --toConnect--> Terminal --connectivityNode -->
//    ConnectivityNode --terminal--> Terminal --fromConnect-->
//    Disconnector

Rule BreakerIsolationConforms
  given   b is a Breaker
  if      e1 is toConnect of connectivityNode of terminal
                  of fromConnect of b
          e2 is fromConnect of terminal of connectivityNode
                  of toConnect of b
          e1 is a Disconnector
          e2 is a Disconnector
  then    isolationCompliance of b is true.

Rule BreakerIsolationFromViolation
  given   b is a Breaker
  if      e is toConnect of connectivityNode of terminal
                  of fromConnect of b
          e is not a Disconnector
  then    isolationCompliance of b is false.

Rule BreakerIsolationToViolation
  given   b is a Breaker
  if      e is fromConnect of terminal of connectivityNode
                  of toConnect of b
          e is not a Disconnector
  then    isolationCompliance of b is false.
```

Figure 5: Breaker Isolation Rules in SADL

These rules use concepts imported from an extract of the CIM RDF model. The first rule defines the requirements for compliance while the following two rules define specific cases of non-compliance.

One way to illustrate the reason that SADL was developed is to compare the first rule of Figure 5 with the Jena Rule syntax into which it is converted. Note that this is very similar to SWRL syntax. The converted rule is shown in Figure 6.

```
[BreakerIsolationConforms:
(?b rdf:type SelectedCim:Breaker) ,
(?b SelectedCim:fromConnect ?var1) ,
(?var1 SelectedCim:terminal ?var2) ,
(?var2 SelectedCim:connectivityNode ?var3) ,
(?var3 SelectedCim:toConnect ?e1) ,
(?b SelectedCim:toConnect ?var4) ,
(?var4 SelectedCim:connectivityNode ?var5) ,
(?var5 SelectedCim:terminal ?var6) ,
(?var6 SelectedCim:fromConnect ?e2) ,
(?e1 rdf:type SelectedCim:Disconnector) ,
(?e2 rdf:type SelectedCim:Disconnector)
  -> (?b SelectedCim:isolationCompliance 'true'^^xsd:boolean) ]
```

Figure 6: First Rule of Figure 5 in Jena Rule Syntax

While SADL was first conceived less than two years ago, it has already found its way into use both in a GE business as a tool for capturing engineering models for equipment maintenance requirements and into research projects as a way of formally capturing the important concepts of the research domain. Experience has shown that SADL is surprisingly scalable. SMEs are currently building and maintaining applications with hundreds of concepts and tens of thousands of rules. However, the complexity of models and the size of data sets for the smart grid will certainly challenge both SADL and semantic technology in general. We look forward to identifying and researching solutions to the pressure points as the semantically enabled smart grid transitions from thought to practice.

4. CONCLUSION

In this paper we highlight information modeling and semantic web technologies in a smart grid context. We also introduce next generation capabilities that can be added once a semantic information model is added to the mix. Semantic web technology not only allows for smart grid software system interoperability, it also allows for next generation capabilities not possible given typical document/schema centric approaches. SADL allows non-technical users, those with depth in the grid domain, to more easily become part of the modeling team and to understand the models created. In addition, SADL provides domain users with the ability to develop and test new smart grid analytics without the use of programming languages or information technology resources. While this capability is attractive, model governance and process management become an evermore-important element of the smart grid. SADL and the SADL-IDE in Eclipse provide tight integration of version control in systems such as CVS and SVN.

5. REFERENCES

[1] "Report to NIST on the Smart Grid Interoperability Standards Roadmap", Electric Power Research Insitute (EPRI), project manager Don Von Dollen, August 10, 2009, http://www.nist.gov/smartgrid/Report%20to%20NISTlAugust10%20(2).pdf.

[2] "IEC 61970 Energy management system application program interface (EMS-API) - Part 301: Common Information Model (CIM) Base", IEC, Edition 1.0, November 2003

[3] "IEC 61968 Application integration at electric utilities - System interfaces for distribution management- Part 11: Common Information Model (CIM)", IEC Draft.

[4] "NIST Framework and Roadmap for Smart Grid Interoperability Standards" (Release 1.0, Draft), September

2009, http://www.nist.gov/public_affairs/releases/smartgrid_interoperability.pdf

[5] "RDF/XML Syntax Specification (Revised), W3C Recommendation", February 10, 2004, Dave Beckett, ed., http://www.w3.org/TR/rdf-syntax-grammar/.

[6] "RDF Vocabulary Description Language 1.0: RDF Schema, W3C Recommendation", February 10, 2004, Ramanathan V. Guha, Dan Brickley, eds., http://www.w3.org/TR/rdf-schema/.

[7] "OWL Web Ontology Language Reference", February 10, 2004, Mike Dean and Guus Schreiber, eds. http://www.w3.org/TR/owl-ref/

[8] "SWRL: A Semantic Web Rule Language Combining OWL and RuleML", May 21, 2004, Ian Horrocks et. al., http://www.w3.org/Submission/SWRL/.

[9] Fan, T., Wu-chih, H., Liau, C., "Decision Logics for Knowledge Representation in Data Mining", In Proceedings of the 25th Annual International Computer Software and Applications Conference (COMPSAC), 2001.

[10] "OWL Web Ontology Language Guide", February 10, 2004, Michael K. Smith, Chris Welty, and Deborah L. McGuinness, eds., http://www.w3.org/TR/2004/REC-owl-guide-20040210/#OwlVarieties.

[11] Lewis Hart et. al, "OWL Full and UML 2.0 Compared", March 12, 2004, http://www.omg.org/docs/ontology/04-03-01.pdf.

[12] "Namespaces in XML 1.0 (Second Edition)", August 16, 2006, Tim Bray et. al., eds., http://www.w3.org/TR/xml-names/.

[13] UN/CEFACT Home, http://www.unece.org/cefact/

[14] "SPARQL Query Language for RDF", January 15, 2008, Eric Prud'hommeaux and Andy Seaborne, eds., http://www.w3.org/TR/rdf-sparql-query/.

[15] SADL SoureForge Home, http://sadl.sourceforge.net.

[16] Eclipse IMP Home, http://www.eclipse.org/imp/.

6. BIOGRAPHY

Andrew Crapo received a B.S. in Physics from Brigham Young University in 1975, an M.S. in Energy Systems from the University of Central Florida in 1980, and a Ph.D. in Decision Sciences and Engineering Systems from Rensselaer Polytechnic Institute in 2002. He is a senior professional information scientist at the GE Global Research Center where he has worked since 1980. His work has focused on applications of information science to engineering problems including applied artificial intelligence, human-computer interactions, and information system architectures. More recently he has focused on modeling and the application of Semantic Web technologies to engineering and business problems.

Xiaofeng Wang received B.S. and M.S. degrees in electrical engineering from Tsinghua University, Beijing, China, in 1995 and 1998, respectively, and the Ph.D. degree from the Electrical and Computer Engineering Department of Michigan Technological University, Houghton, in 2001. Currently, he is a System Engineer with GE Energy. His interests include power system modeling, enterprise integration, Smart Grid, and Semantic Web Technology

John Lizzi received a B.S. in Computer Science from Siena College in 2001, a M.S. in Computer and Systems Engineering from Rensselaer Polytechnic Institute in 2003, and a M.B.A. from the State University of New York at Albany in 2008. Since 2000, John has been working as a Research Scientist in the Computing and Decision Sciences Group at General Electric Global Research, in Niskayuna, NY. John has worked on developing technology and solutions in a variety of domains including maintainability engineering, air traffic management, television broadcast operations, healthcare, and energy services. His primary interests include software architecture, enterprise integration, modeling, and simulation.

Ron Larson began his career in 1983 as a software engineer for Harris Controls in Melbourne Florida. During his tenure at Harris, he held a number of increased senior software leadership and managerial positions focused on electric utility SCADA/EMS In 1995 Ron was the R&D director for Scientific-Atlanta's VSAT operations, and in 1997 he held program management and business development roles for Exigent Corporation. Ron joined the then GE Harris joint venture in 1999 as business development manager, leading technology due diligence activities and other initiatives in support of M&A and equity investments. Ron has held a number of key strategic software technology roles during his tenure at GE. Ron championed cyber security strategic initiatives for the T&D products and services in GE Energy. He is currently the Manager of Software Systems Engineering for the Energy Services business unit at GE Energy.

Ron has been directly or indirectly involved in the development of several electric utility industry standards. He was a key member of a cross-functional team under then EPRI UCA that defined the first version of the popular ICCP standard. During his tenure at Harris, under contract with EPRI, he led the development of the first reference implementation of ICCP. For the past 15+ years Ron has contributed to the evolution of the Common Information Model (CIM), data exchange protocols, and related information exchange standards and practices. Ron leads GE Energy's software systems team who are defining the software architecture for GE Energy's Smart Grid initiatives. Ron is actively involved in various electric utility industry standard efforts for Smart Grid software systems including activities to standardize CIM based service definitions in support of Smart Grid needs.

Ron attended Florida Institute of Technology, where he received his Bachelor of Science in Computer Science in 1983, and Masters of Science in Computer Science in 1986.