

# Knowledge Representation for Power System Modelling

A. deVos, Member IEEE  
Langdale Consultants  
Elanora Heights, NSW 2101, Australia

C.T. Rowbotham  
Langdale Consultants  
Elanora Heights, NSW 2101, Australia

**Abstract:** Modelling power systems is an area of ongoing interest in the transmission management and control systems community. Continuing development is driven by two forces. The traditional tasks of model maintenance and management must be achieved with fewer resources. At the same time, model exchange and coordination has become a priority. The latter force arises from the disaggregation of utility functions and the introduction of power markets.

This paper begins by identifying some of the power system modelling tasks that have become important, but are ill served by current tools and techniques.

Among these are model versioning and version control, migration of models between different schema, the transformation of models for different purposes or applications, and the merging of models from different sources. These tasks are typically handled by semi-manual methods or heavily customized software.

The paper then describes the application of knowledge representation to power system modelling. In particular, the power of this approach to provide generic solutions to the foregoing problems is explored.

Knowledge representation is contrasted with more common data representations and put into context with current industry initiatives, EPRI CIM, UMS DAF and XML/CIM.

Finally, the feasibility of using knowledge representation for power system models is illustrated with a case study from a major Australian distribution utility.

**Keywords:** power system modelling, knowledge representation, electronic data interchange, data models, power system control, transmission control, data management, data communication, software standards

## I. INTRODUCTION

Models are essential to the operation of modern power systems. Simulation of the power system is necessary for both planning and operations and depends on appropriate models. In the operations arena models are typically more comprehensive than those used for planning. Operational

models support analysis of incoming SCADA data as well as simulation of expected and unexpected operational scenarios.

An operational power system model is a relatively complex set of information, involving between 100 and 1000 different classes of information. Moreover, these models are closely tied to telemetry models and loosely tied to a range of other information throughout the utility. Creation, maintenance and verification of models are significant activities for most transmission operators.

The disaggregation of utility functions and the introduction of power markets in many national power systems has increased the model maintenance burden. For example, the split between Regional Transmission Operators (RTO's) and Independent System Operators (ISO's) in the United States means that many overlapping models must be created and coordinated with each other where previously there was a single model. This introduces new problems of model information exchange, and exacerbates old problems of version control and verification.

Distribution authorities also face new modelling tasks with the increasing deployment of Distribution Management Systems (DMS). While the DMS capabilities vary from system to system, all require models of the sub-transmission and feeder network that an earlier generation of control systems did not. In the distribution environment, model maintenance solutions developed for Energy Management Systems (EMS) are rarely suitable. Unlike most main transmission systems, a typical distribution system is subject to constant expansion and rearrangement.

In this context, it is worth revisiting the traditional modelling approaches to find improvements. As an aim, new techniques should enable existing teams to meet the greater challenges they now face. The best techniques would relieve the tedious and error prone aspects of the task, freeing maintainers to apply their judgment and experience to network and control system problems.

## II. MODELLING TASKS

### A. Model Versions

Maintenance implies the existence of multiple versions of a power system model. As a first approximation we can

envisage these versions to be created and deployed one at a time. Figure 1 shows this arrangement. Each version is created by editing the previous version and deployed simply by committing the edits.

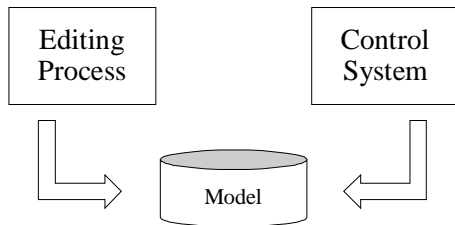


Figure 1

Practical modelling sub-systems are more elaborate than this. There is generally a verification step and the opportunity to make corrections before a model is placed in service. This leads to a data flow more like figure 2.

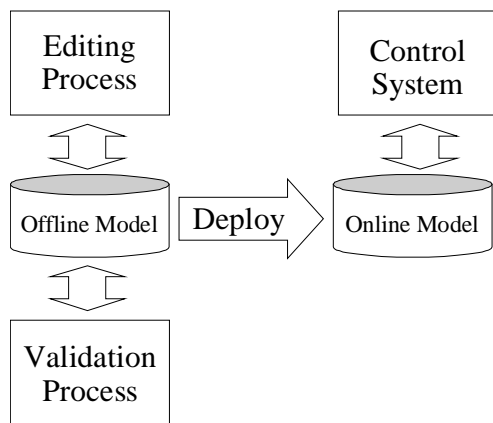


Figure 2

Many modelling sub-systems provide a limited form of parallel development. A typical architecture resembles figure 2 with the addition that the offline model database supports multiple users. In this approach, several maintainers work on the same model version until it is “frozen” for validation and deployment.

It is a methodology best suited to original system configuration or infrequent deployment of large packages of changes. None of the maintainers can validate or deploy their changes until everyone is ready. If, for example, an urgent model change is required, it may not be possible to fit it into the model release schedule.

In a distribution system, changes are deployed on a daily basis and fully parallel development is required. With disaggregation, this may become a requirement for transmission operators as well. In this scenario, several versions of the model are created and validated in parallel as shown in figure 3. (We call them projects.)

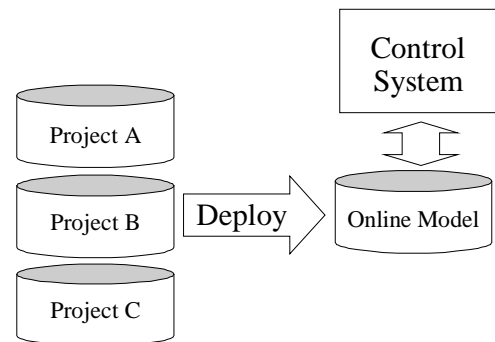


Figure 3

A system that supports parallel development must deal with the “lost update problem”. Projects A and B start from the same baseline and develop in parallel. If A is deployed before B, updates that were made in A are apt to be lost, unless they are somehow merged with B.

In practice, modelling systems deal with fully parallel development and the lost update problem with varying degrees of ease and reliability. Ad hoc and semi-manual systems are common.

#### B. Model Transformations

Power system models and the control systems that use them do not exist in isolation. Frequently, the models used in disparate control systems must be coordinated with each other and with other information systems. This generally requires the transformation of models from one representation and schema to another.

In a distribution system there may be different tiers of control, with a main control system at the top tier and substation or feeder automation at lower tiers. Asset management or geographic information systems may also be coordinated with the control systems. Each will support a different power system model representation, hence the need arises for model transformations. This is illustrated in figure 4.

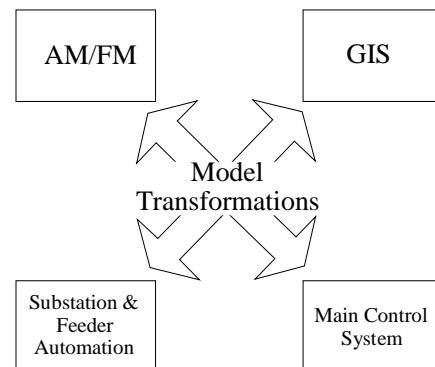


Figure 4

Similarly, the ISO/RTO case involves exchanging models between transmission system operators with disparate control systems and different model schema, see figure 5. In this case, native models are first transformed into a common representation and schema. The proposed standard in the US is a schema [5] derived from the EPRI CIM [4] and represented as XML. This is called the CIM XML language [7].

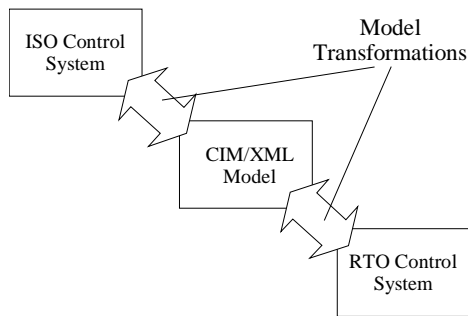


Figure 5

Model transformation solutions are frequently ad hoc, although vendors are beginning to develop converters (sometimes called filters) from their native format to the CIM XML model exchange language. In many cases model transformations remain labour-intensive and error-prone. Maintenance of hard-coded transformation software to track schema changes could also become a burden.

### C. Partial Models

Another aspect of the ISO/RTO problem might be termed “the stitching problem”, meaning the problem of joining regional models together. Each regional model includes information that must be stripped away before it is stitched into the system model. That includes any overlap or reduced equivalent of the neighbours, the low-voltage network, and attributes that are only relevant to the regional operator.

Because of this overlap stripping, an ISO and an RTO will actually exchange a partial model. Figure 6 shows the whole and partial model for a region, B, with neighbours, A and C.

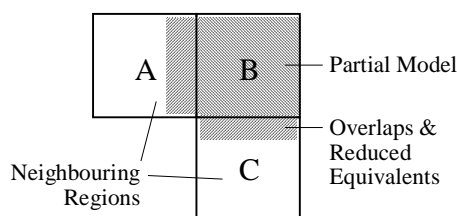


Figure 6

In general, a partial model is one that is incomplete for the intended application. The difference between two versions of a model is another form of partial model. Such difference models can form the basis of a version management strategy

within a utility. Difference models are also well suited to the ISO/RTO model exchange scenario.

Partial models, including difference models, represent a challenge for a modelling subsystem. They are nominally invalid because required relationships may be missing. Moreover, the modelling subsystem must preserve information about each partial model even after it is stitched into a complete model. This permits further editing or stitching operations to be performed correctly.

### D. Database Solutions

Some of the facilities outlined above prove difficult to implement with the usual modelling vehicle: the database management system. Mainstream database systems employ an interpretation of the relational model of data. In the next section, we will introduce a more flexible alternative for dealing with versions, transformations and partial models.

Here we list the aspects of the relational model, as implemented in mainstream database systems, that tend to limit modelling systems. These observations also apply to object databases although their diversity makes it harder to generalise.

#### 1 Records

The smallest, indivisible unit of information is the tuple or record, which embodies several data fields. While it is possible to make associations between tuples, there is no general mechanism to qualify the fields within a tuple with information about their source or version. Nor is there a general mechanism to provide alternative versions of individual field values.

#### 2 Keys, Table Names and Column Names

Keys, table names and column names are unique only within their respective scopes, local to a database. However, a concept of universal, identity is desirable when dealing with models that span different administrative domains. In other words, keys, table names and field names are likely to be ambiguous when models are exchanged between utilities or between different systems within a utility.

#### 3 Schema Migration

Schema migration is the process of transforming a model to comply with a different schema. In an isolated system, this is required only when the software is upgraded. In the current environment, however, such transformations may occur regularly.

Database systems make dealing with schema differences a difficult task because of their strict enforcement of a single, monolithic schema. For example, this prevents the original and migrated populations from residing in the same system.

Moreover, a complex, multiple pass migration strategy is required to avoid schema constraint violations.

Traditionally, the foregoing issues have been dealt with on a case by case basis by combination of schema design techniques and supporting software. A comprehensive solution requires a new framework layered above the database system. Knowledge Representation provides such a framework.

### III. KNOWLEDGE REPRESENTATION

Knowledge representation (KR) is a theory of data that can be applied to modelling tasks. KR concepts can be implemented in markup languages, programming interfaces, data transformation functions and data repositories. Two utility industry standards already use KR concepts to represent power system models:

- The CORBA Data Access Facility (DAF) [6] is an API based on a KR formulation of the EPRI CIM [5] schema.
- The CIM XML language, mentioned in the previous section, is a language for exchanging models. It uses the same KR schema as the DAF.

The attraction of the KR approach to standards is its generality. It can be applied equally well to a variety of control systems with different native model representations. Moreover, it is straightforward to create a concrete KR schema from the abstract EPRI CIM schema, currently formulated in UML. An equivalent relational database schema is not as obvious to derive and not as broadly applicable.

#### A. Resource Description Framework

The KR framework used in these standards was supplied by a W3C recommendation, the Resource Description Framework (RDF) [1]. RDF contains a KR data model, a syntax for encoding data in XML, and a vocabulary [2] for expressing schema information.

In the RDF data model, a *resource* is anything that can be identified. A Uniform Resource Identifier (URI) [3] is used to designate a resource. A *property* is any characteristic of a resource that can be described with a *value*. The triple: (resource, property, value) is the atomic unit of information in RDF and is called a *statement*. The parts of a statement are its *subject*, *predicate*, and *object* respectively. Figure 7 shows a statement diagrammatically.

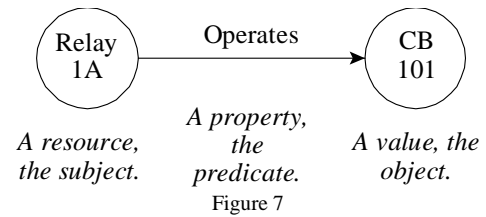


Figure 8 shows a collection of statements forming a small model. Because the value of some of the statements is a resource, which is in turn the subject of another statement, this model forms a directed, labelled graph (DLG).

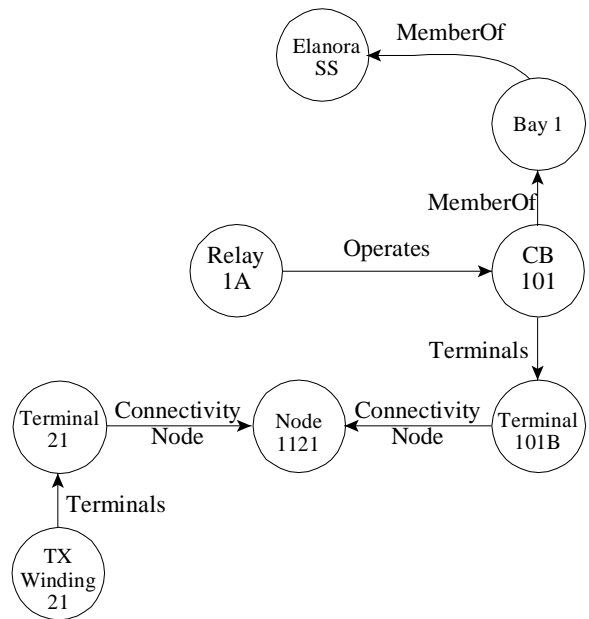


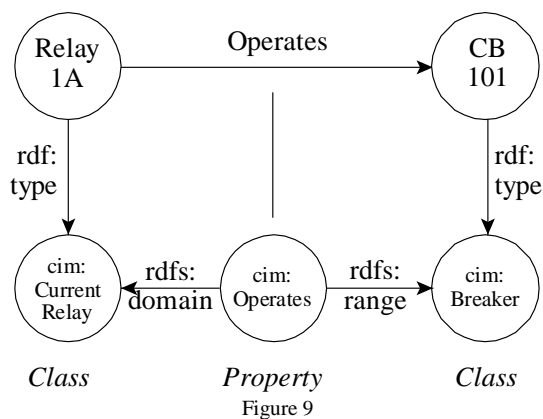
Figure 8

The labels in figure 8 are abbreviated for clarity. The full identification of a each resource and each property is a URI that embodies schema and version identification. A URI is unambiguous in any context, for example:

[http://iec.ch/TC57/2000/CIM-schema-cimu09a#Bay.MemberOf\\_Substation](http://iec.ch/TC57/2000/CIM-schema-cimu09a#Bay.MemberOf_Substation)

The *http:* in this URI indicates the use of the http naming scheme and not that any document is available at this address via the http protocol.

Schema information is introduced into this graph in two ways. First, any resource may have a *type* property that designates its *class*. Secondly, both properties and classes are themselves resources and can be described by further statements. Statements about classes and properties form the schema. This is illustrated in figure 9.



The picture is completed by another category of information: second order statements. A second order statement describes or qualifies another statement. This is a vehicle for conveying version, authorship, accuracy or similar qualifications, as shown in figure 10. Here the statement about Relay 1A has been reified, that is, made concrete as a resource. It then serves as the object of two other, second order, statements.

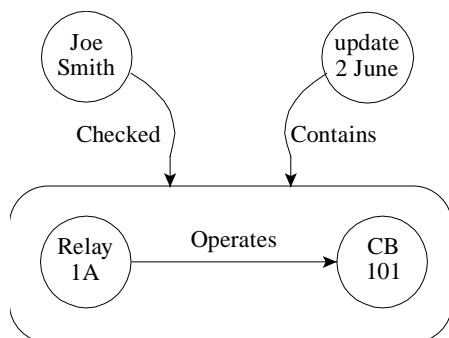


Figure 10

## B. Comparison

Comparing the RDF information model to that of conventional databases, several differences emerge:

- (i) The statement is a fundamental unit of information, sometimes called a *fact*. In contrast, tuples and objects are composite structures.
- (ii) Schema is descriptive not prescriptive. A schema can be enforced, by modifying the population to conform, but this is not mandatory as it is in a database system.
- (iii) Schema is flexible. A set of statements can stand alone, without a schema. It is also possible to have a partial schema or multiple schema. The schema can support functions other than validation, including inference and transformation.

- (iv) Schema is extensible. Generic RDF software might be expected to interpret public schema vocabularies such as [2] and [8]. However, applications may define and use additional application-specific schema information.
- (v) Second order statements can qualify individual facts (corresponding to individual fields in a tuple or object). Several conflicting facts can be represented at once, distinguished by second order statements.
- (vi) Universal identification enables one model to extend, qualify or describe another. The models could reside in different systems in different administrative domains.

## IV. AN APPLICATION

The foregoing KR framework is a promising environment in which to formulate power system modelling tasks. We will briefly describe a case study that demonstrates the advantages of this approach in solving the modelling problems outlined in chapter II: versioning, transformations, and partial model stitching.

The case study answers the question: can these techniques work in practice where highly proprietary systems are involved?

### A. The Problem

One of Australia's largest distribution utilities, Integral Energy, operates a distribution network with an installed capacity of 5,300 MVA.[9]. Its network is under continuous expansion and the task of maintaining the control systems is significant.

The power system model used by the control system includes an equipment model, electrical topology, display layout information, and telemetry and local control configuration data.

The control system has two tiers: a main SCADA system with replicated control centres and a network of substation management systems. The two tiers require different models although the same source information is used for their configuration. Model transformations are necessary both for efficiency and to eliminate manual errors in coordinating the two tiers.

The control system team must work in a fully parallel fashion, with model updates for different substations under concurrent development. This leads to the creation of partial models representing a set of changes or an area of interest. These must be stitched (merged) with each other and the complete model.

The original tools for editing and maintaining the model were ad-hoc. They did not anticipate a) the scale of the task, b) the complexity of the task or c) the present, reduced size of the SCADA team. As with many ad-hoc approaches these tools were arcane and somewhat error-prone.

Given the tools available and the extent of the power system augmentation work, there was considerable pressure on the SCADA team. Skilled staff were spending many hours editing, testing and correcting low-level data files rather than using their expertise to solve more important network issues.

## B. Solution

The authors developed a general-purpose KR-based tool, called I-Builder, that performs data capture, display, editing, merging, transformation and export of KR models. Figure 11 is a block diagram of I-Builder. The multiple, concurrent project approach illustrated in figure 3, is used, and figure 11 shows I-Builder in the context of one such project.

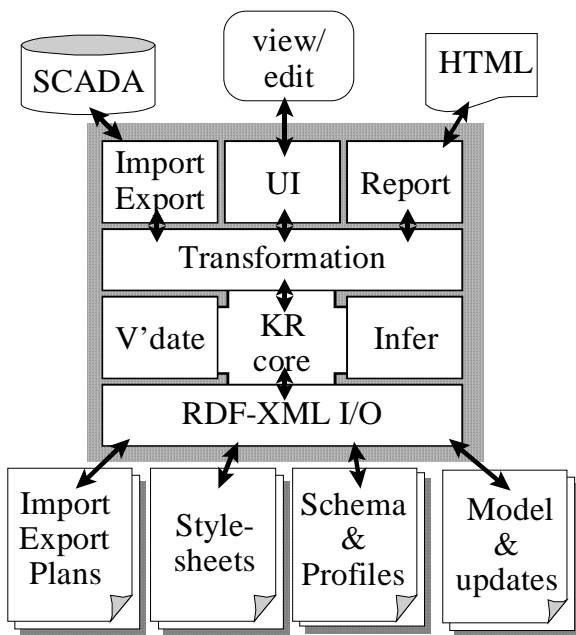


Figure 11

A project lifecycle consists of:

- Capturing the relevant baseline power system model segments from the SCADA systems (Import/Export).
- Updating or extending the model the via the user interface (UI). Profiles are used, which are pre-built templates ranging in size from a single point to a whole substation.

- Exporting the updates and new sections of the model back to the SCADA system(s).

The following sections consider how this architecture deals with the problems of versioning, transformation, and partial models posed in section II.

### 1 Versioning and Partial Models

Version management relies on reducing the model to atomic units, the statements, in the KR core. Each model statement is qualified by a second order statement (in the manner of figure 10) that records its origin.

Editing does not change existing statements but adds new statements to the KR core. These are allowed to contradict existing statements. The new statements are also qualified with details of their origin.

The lost update problem described in chapter II is avoided because only the new statements are exported back to the SCADA system. These are identified by their second order statements.

During the lifetime of a project, unfinished work may be saved many times. Rather than commit the project back to an offline SCADA system database, the changed contents of the KR core are saved in XML files. A simplified RDF syntax similar to [10] is used. The origin of all information, embodied in second order statements, is therefore preserved. (For convenience, the contents of the KR core is split among XML files based on origin.)

Infrequently, the segment of the baseline model captured in the project may be updated in some other project or in the SCADA system. In this situation, the baseline model may be recaptured without losing updates already made in the project. The updated can then be validated against the new baseline and adjusted as required.

The KR core is responsible for stitching updates to baselines. It does this based on the universal identification scheme (URI's) used in RDF. This type of stitching also takes place between models and the schema, between related schema and between schema and import/export plans.

### 2 Transformation and Inference

The two tiers of Integral's control system use different model representations and different schema. This creates a requirement for model transformations. The transformation infrastructure in I-Builder serves as the I/O or query subsystem. It drives the import/export function, the user interface and generates reports.

I-Builder implements a general purpose mapping between KR graph models found in the KR core and the hierarchical

XML Information set. Transformation results are presented to the other modules through the Simple API for XML (SAX) [11].

Transformations are directed by stylesheets, which are themselves KR models. These are a KR equivalent to the XML transformation language, XSLT [12]. The stylesheet models are intimately linked with schema information because properties and classes are the most important criteria to select and process statements. They are, in effect, schema extensions.

Figure 12, illustrates rules coupled to schema (compare to figure 9). In the figure, a rule is associated with the Operates property. The rule specifies an action to be performed when that property is encountered in the model, and a context in which this rule applies.

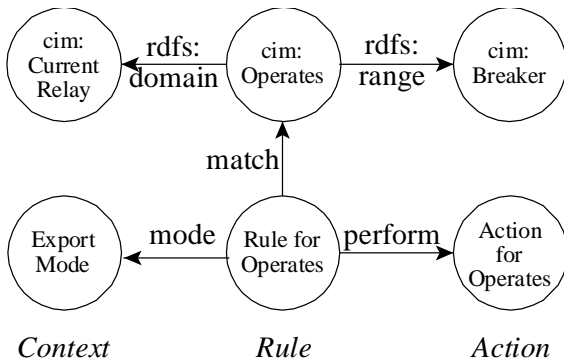


Figure 12

The import/export module shown in figure 11 also forms part of the transformation chain. It is responsible for converting XML to SQL and rowsets back to XML. The chief challenge is to order the SQL queries correctly. The correct order depends on relationships found in the schema. Thus the import/export plan is also a KR model that extends the schema.

Finally, inference is a process related to transformation. Inference adds statements to the model based on other statements found in the model and a set of rules (often the schema itself). For example, inference is used when profiles (ie templates) are copied into the power system model. Statements must be added to the profile based on inference rules that are formulated as a schema extension.

### 3 Experience

I-Builder is now the vehicle for all model maintenance at Integral and clear improvements in speed and accuracy have been achieved. Creating and validating a new substation, for example, is many times faster. We attribute this to the features that the KR techniques enabled.

One important observation is that interactive performance is satisfactory. We expected the richness of the data model

implemented in the KR core to mitigate against performance. We attribute the performance achieved to the use of partial models, relying on stitching process to coordinate with the overall model. This is an inherently scalable approach.

The architecture has produced a few other bonuses: because the user interface is driven by the KR based transformation infrastructure, it is immune to schema changes. Only the stylesheets need be updated. We have tested this by implementing the EPRI CIM schema [5].

Another bonus is that the models, the related transformation rules and the schema are kept in XML with a straightforward vocabulary. These are corporate assets and locking them away in hard-coded software and more proprietary forms of data storage would be undesirable.

## V. CONCLUSION

This paper has outlined some of the issues with typical power system modelling solutions. The difficulty of making progress with these issues in a conventional database framework was discussed. Knowledge Representation techniques were introduced as a potentially more fruitful approach. Although KR techniques are unfamiliar to most practitioners, the case study we describe demonstrates that they are feasible, at least as formulated in the Resource Description Framework.

Several features of RDF including its unit-of-data, the second order statements and universal identification make it scalable to large data handling problems. It is particularly useful where multiple systems and administrative domains are involved. (The originators of RDF intend it to work at the scale of the web.) Those qualities have seen RDF incorporated into several power industry standards [5,6,7]. Going beyond that, we have found RDF and KR techniques can be incorporated into a software framework to solve significant real-world problems.

## VI. REFERENCES

- [1] *Resource Description Framework (RDF) Model and Syntax Specification*, W3C Recommendation, 22 February 1999, Ora Lassila, Ralph R. Swick.
- [2] *Resource Description Framework (RDF) Schema Specification*, W3C Candidate Recommendation 27 March 2000, Dan Brickley, R.V. Guha, Netscape
- [3] *Uniform Resource Identifiers (URI): Generic Syntax*; Berners-Lee, Fielding, Masinter, Internet Draft Standard August, 1998; [RFC2396](#).

- [4] *Guidelines for Control Center Application Program Interfaces*, EPRI Technical Report TR-106324, Project 3654-01, Final Report, June 1996.
- [5] *CIM RDF Schema Exported from Ccapi.mdl Version: CIMU09a*, Leila Schneberger. OMG TC Document dtc/2000-11-10.
- [6] *Utility Management System Data Access Facility*, Arnold deVos. OMG TC Document dtc/2000-11-09.
- [7] *Simplified RDF Syntax for Power System Model Exchange*, Arnold deVos, November 2000. <http://egroups.com/group/cimxml>
- [8] *DARPA Agent Markup Language* <http://www.daml.org/>
- [9] *Integral Energy Annual Report, 1999*
- [10] *A Strawman Unstriped Syntax for RDF in XML*, Tim Berners-Lee, November 1999
- [11] *Simple API for XML (SAX)*, David Megginson <http://www.megginson.com/SAX/>
- [12] *XSL Transformations (XSLT) Version 1.0*, W3C Recommendation 16 November 1999, James Clark et al

## VII. BIOGRAPHIES

**Arnold deVos** (M' 96) received his BE(Elec.) from the NSW Institute of Technology, Sydney, Australia in 1981. He worked in system operations at the NSW Electricity Commission and developed software for load forecasting and operational data handling. He also developed inter-utility data exchange standards for the S.E. Australian interconnection. Subsequently he joined Megadata P/L (now part of Logica, UK) where he was the architect of the MOSAIC SCADA system, and developed its distributed database and user interface. In 1995 he joined ALSTOM ESCA where he developed EMS power system modelling tools. He is now a partner in Langdale Consultants where he consults on system integration issues and develops software to address this area. He is active in several utility forums and is the author of the CORBA standard interface for EPRI CIM models.

**Christopher Rowbotham** received his BE(Elec) from University of NSW in 1975. In 1981 he joined Megadata P/L (now part of Logica, UK) where he became Technical Director. In this role he oversaw the companies product development. In 1995 he founded Langdale Consultants, who develop software and advise clients in the area of realtime system integration.