Patterns and Practices for CIM Applications

Rahul Khare, Mostafa Khadem, Sainath Moorty, Kittipong Methaprayoon, Jun Zhu

***Abstract*—The role of Common Information Model (CIM) in the Electric Utility industry, especially in Smart Grid, is now becoming critical. In a recent letter to FERC, NIST has identified CIM as one of the "five foundational families of standards" that are "fundamental to Smart Grid interoperability". CIM applications range from CIM-based model exchange to CIM-based information management, in the areas of EMS, DMS, MMS and Substation Automation. This paper provides technical guidance and recommendations for designing and developing architecturally-sound CIM applications, based on authors' experience in large energy market design projects. It consists of a collection of patterns. Each pattern addresses a specific design problem in CIM application development, discusses considerations surrounding the problem, and presents a solution that balances various drivers or forces. The objective is to provide the user with a proven starting point from which robust and extensible CIM applications can be developed.**

***Index Terms*—CIM Applications, CIM Extensions, CIM Incremental, CIM Profiling, CIM Validation, Enterprise Application Integration (EAI), Enterprise Information Management (EIM), International Electrotechnical Commission (IEC), Model Exchange, Market Management System (MMS).**

I. Introduction

The Common Information Model (CIM) was first initiated by EPRI more than a decade ago, aimed to provide application programs a platform-independent view of the power system [1]. Since then, the International Electrotechnical Commission (IEC) Technical Committee 57 (TC57) has been actively engaged in standardizing CIM for the electric power industry. One of the significant achievement is IEC 61970 (Energy Management System Application Program Interface, EMS-API) designed to promote application interoperability for Energy Management Systems (EMS) [2]-[5]. A series of other standards were also developed to address critical integration requirements in other areas. IEC 61968 (Application Integration at Electric Utilities – System Interfaces for Distribution Management) addresses the enterprise integration requirements in Distribution Systems [6], while IEC 61850 (Communication Networks and Systems for Power Utility Automation) focuses on Substation Automation [7]. With CIM as core, these IEC standards provide a common language and protocols among heterogeneous power system applications, and enable development and use of plug-in applications.

As domain ontology [8], CIM describes all aspects of a power system. It defines domain entities and the relationships using a formal semantic specification. More specifically, CIM defines classes, attributes and relationships using Unified Modeling Language (UML). It is an object-oriented information model that provides a semantic framework to facilitate enterprise application integration in the power industry.

In current electric utility industry, both in United States and world-wide, CIM has become a critical component of Transmission, Distribution and Energy Market systems. In a recent letter to FERC [9], National Institute of Standards and Technology (NIST) has identified CIM as one of the "five foundational families of standards" that are "fundamental to Smart Grid interoperability". The work on harmonizing standards is on-going among NIST, IEC and a number of nations in the world [10]. All this is leading to a large number of CIM-based application developments.

Development and maintenance of CIM-based applications, however, is a challenging task. Some of the identified technical challenges include:

- CIM models all the aspects of a power system and its operation, thus presenting overwhelming information to application developers who normally concentrate on a specific application area.
- As a vendor-neutral standard, CIM is loosely-defined to provide flexibility needed for heterogeneous application integration. Unfortunately, this has resulted in semantic ambiguities and raises interoperability issues.
- CIM is an object-oriented information model. Mapping an object-oriented model to a proprietary relational database scheme is a challenging task.
- CIM is constantly changing, and so are the business requirements. The application design may be specific to the problems at hand, but must be general enough to adapt to the CIM evolution and the changing business requirements.

These and other challenges typically increase the cost of development, create project delivery risks, and make the software maintenance expensive.

This paper provides technical guidance and recommendations on how to design and develop architecturally sound CIM applications. It consists of a combination of CIM application specific design problems and their solutions – referred to as *patterns*. Each pattern addresses a specific design problem, discusses the considerations

R. Khare and M. Khadem are with Ventyx, an ABB company, Santa Clara, CA 95050 USA (e-mails: rahul.khare@us.abb.com, mostafa.khadem@us.abb.com).

S. Moorty and K. Methaprayoon are with ERCOT, Taylor, TX 76574 USA (e-mails: smoorty@ercot.com, kmethaprayoon@ercot.com).

J. Zhu is with Power Info LLC, Bothell WA 98012 USA (e-mail: junzhu@powerinfo.us)

surrounding the problem, and presents a solution that balances various drivers or forces. It incorporates the experience that architects and developers have gained by repeatedly building CIM solutions. The patterns described in this paper are discovered and observed based on the implementation of CIM solutions on large-scale energy market design projects. These real-world-based principles and guidelines will help CIM application designers and developers to create flexible, elegant, and reusable solutions.

II. CIM-BASED APPLICATIONS

This section explores how CIM is applied to the enterprise integration requirements in the energy industry.

A. CIM for Model Exchange

Electric utility organizations have long needed to exchange system modeling information among various information systems or with one another, in order to construct simulation environments for power system economics, security and reliability analysis. The purpose of information sharing is to effectively and efficiently support system operation and planning. With an increasing number of sub-systems exchanging power systems models (Figure 1), there is a growing need for an open model exchange methodology [8].

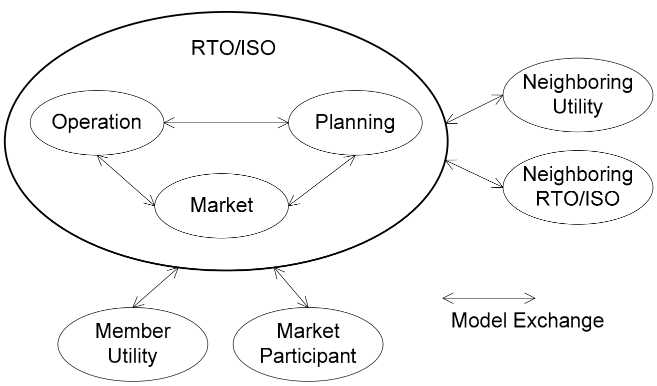

Fig.1. Model Exchange in a RTO/ISO Environment

CIM-based model exchange standards [4], [5] provide a comprehensive vendor-neutral approach to model exchanges. These standards leverage the latest software industry standards such as XML *(Extensible Markup Language)* and RDF *(Resource Description Framework)*, and specify a standard XML-based model exchange format called CIM/XML [8]. The common model exchange format is governed by a CIM resource definition language called CIM/RDF/OWL *(OWL: Web Ontology Language)* which encodes the CIM model definition with a standard semantic description language RDF/RDFS/OWL *(RDFS: RDF Schema)*. All of the tags used to mark up the CIM objects are supplied by the CIM/RDF/OWL schema. Specified by the common resource definition, the vendor's propriety model information can be converted into a self-contained standard-compliant XML document that can be readily consumed by applications in an external system.

EPRI has sponsored several CIM/XML Interoperability tests where participants perform full and incremental model exchanges based on the standard model exchange format [11], [12]. The issues and recommendations identified during these tests are forwarded to TC57 for further refinement of the standard. After several iterations between theory and practice, the CIM/XML-based model exchange approach has matured to a point where it has been widely accepted and practically applied in many EMS/MMS and other projects.

B. CIM for Application Integration

Electric utility operations now involve multiple business processes across several functional areas, requiring an integration of heterogeneous legacy applications, both internal and external to an electric utility organization [13]. Traditionally, the application integration was accomplished through a point-to-point application connection. As systems become more complex, the unstructured nature of the point-to-point connections provokes security risks, making it difficult to audit, control and configure. With the addition of new applications into the integration framework, software maintenance costs grow exponentially. This makes Enterprise Application Integration (EAI) a critical requirement for enterprises in today's power industry.

Enterprise Service Bus (ESB) concepts help to simplify and manage the inter-connectedness of a Service-Oriented Architecture (SOA). An ESB acts as a shared messaging layer for connecting applications and other services throughout a middleware infrastructure. At the heart of an ESB lies a messaging infrastructure that provides a service interface to the applications in the integration framework. CIM-based industry standards provide common vocabularies and semantics for defining messages that glue the underlying utility enterprise applications. A CIM-based message is a well-defined request or response data packet used to exchange information between CIM-compliant applications. Since CIM serves as a common language, applications can communicate with each other in a loosely coupled environment without knowing their underlying proprietary information models. This significantly reduces the risks and costs involved in integration projects and result in an open enterprise architecture that allows for long-term evolution and expansion (Figure 2).

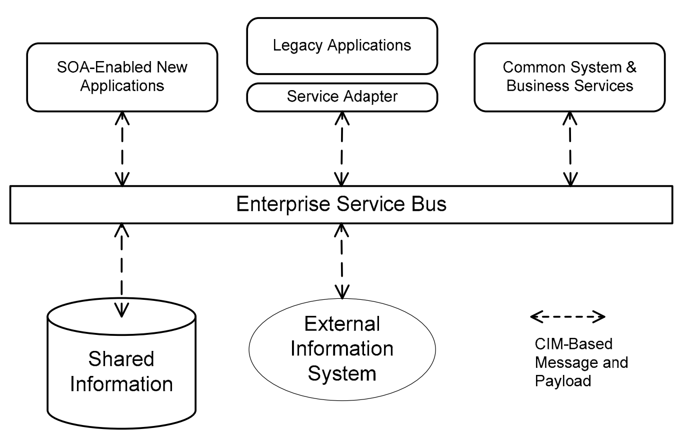

Fig.2. A Service-Oriented Architecture (SOA) Based on CIM Messaging

Furthermore, a CIM-based integration methodology prevents vendors lock-in any applications and allows customers to construct new services to meet the evolving business requirements.

The PJM Advanced Control Center Project (AC²) [14] is a recent implementation using CIM-based messaging for SOA implementation.

C. CIM for Information Management

Enterprise Information Management (EIM) is a business-driven process for organizing, structuring, and managing an organization's information assets. Its objective is to integrate business goals with technology to help the organization continually manage data, simplify business processes, increase productivity, and enhance adaptability. A key application of EIM in utilities is to share the information among the heterogeneous utility information systems and foster collaborations among them.

CIM is an object model for describing and sharing enterprise-wide utility information. It describes the domain based on reality, emphasizing what it is, rather than how it should be used. Because of these features, CIM provides a unified solution for managing heterogeneous utility information models without requiring an overhaul of the existing information infrastructure. The centerpiece of this solution is a CIM-based model management system as shown in Figure 3.

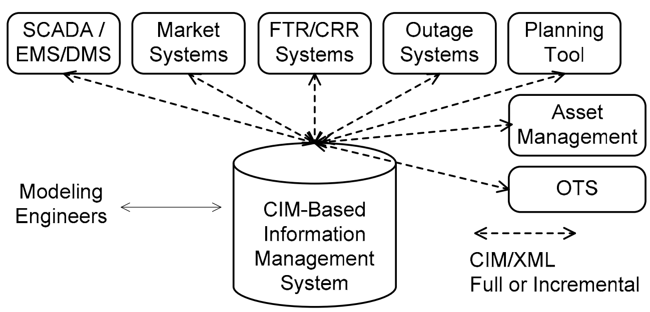

Fig.3. CIM-Based Single-Entry Modeling

In a CIM-based single-entry modeling environment, the centralized model repository of the system consolidates various application models based on CIM and eliminates duplication of data. As a single point of modeling entry for model building and maintenance, the model manager coordinates a variety of utility information system and supplies the shared information to the various information systems via CIM-based model exchange format. Under this scenario, the model information is entered once through the common modeling system, and then becomes available to all other information systems. The CIM-based single-entry modeling methodology provides an automated approach for managing the diverse utility Information systems and keeping them consistent and timely updated [15].

For the ERCOT Nodal Program, a CIM-Based Information Management System has been implemented to be the centralized model repository that provides data in a consistent manner for EMS, MMS, Outage Scheduler, Congestion Revenue Rights (CRR), Planning and Operator Training Simulator (OTS) sub-systems.

III. PATTERNS AND PRACTICES

Building robust and extensible CIM-based applications is a challenging task. CIM is a loosely-defined object-oriented model, different from the propriety information models used in the electric industry. It requires strong object-oriented modeling skills to bridge the gap between CIM and the propriety information model. The CIM application design may be specific to the problems at hand, but may not be general enough to allow for future model evolutions. CIM provides common vocabularies and semantics for message definition, but it is the responsibility of application developers to design reliable and extensible CIM-based message formats. These and other requirements make the development of CIM applications difficult.

The proven design patterns and best practices presented in this section is aimed to guide application developers to build sound and extensible CIM applications. These patterns and practices first describe the problem, followed by a combination of modeling guidance, architectural design, and implementation strategies.

A. CIM Extensions

Issue – When users compare their propriety information model with CIM model, they find that either (a) modeling information is missing in CIM, or (b) some application-specific modeling needs to be added.

Resolution – A common strategy is to extend CIM to meet the requirements. Almost every CIM-related project requires CIM to be extended to be compatible with the underlying application information models.

CIM standards support extension, allowing developers greater control over the information model. This involves adding new classes and attributes to the existing CIM under different namespaces and within the constraints of the classes defined in CIM. However, there is a risk involved in CIM extension if the established modeling principles and practices are not properly followed. Some of the recommended CIM extension practices include:

- Avoid changing the existing CIM. This translates to no deletion of classes and attributes, no alternation of property types and cardinalities, and no modification of existing CIM class hierarchies. Failure to do so can cause failure in applications that operated against the schema prior to the modification. CIM extensions should be considered as an add-on to the standard CIM, and an extension should never break the existing CIM.
- Follow established CIM modeling principles. For instance, the existing CIM modeling principle prohibits the use of multiple inheritances. Introducing such modeling practices in extended CIM could complicate the infrastructure support and make applications less interoperable with external CIM-compliant applications. On the other hand, CIM extension designers should

follow the proper modeling techniques under the CIM modeling principles. For example, since CIM does not enforce strict model hierarchy (*i.e.* objects of the same class can be grouped under different model hierarchies), extension designers can create domain-specific model hierarchies to fit their special needs.

- Adhere to object-oriented modeling philosophy [16], [17]. When describing a domain based on the objects that exist in the real world, the object-oriented modeling stresses on specifying what an object is, rather than how it is used. CIM is an object-oriented model. When extending CIM, a common pitfall is to create an extension that strictly maps to the underlying proprietary information model. This modeling practice may facilitate the short-term information mapping for the integration project. However, such application-oriented CIM extensions are not sustainable in the long run. This is because the proprietary information model, designed to support the applications, is a logical view of the real world, where many conceptual modeling entities are introduced. These logical views no longer fit when the application domain changes. In contrast, an extended model based on real-world concepts reflects the reality and supports all the underlying applications that normally derive the logical view from the physical world. This is in-line with the philosophy that governs the standard CIM design.

B. *CIM Profiling*

Issue – CIM covers a wide range of modeling areas, ranging from operation to asset management. The resulting model is comprehensive, containing thousands of classes cross-referenced to each other. In addition, CIM being a loosely-defined object oriented model, many relationships are defined only at an abstract level with few modeling restrictions enforced. This results in some levels of semantic ambiguities during interpretation. The semantic ambiguities, however, are essential for designing a shared information model, since there is no "one-size-fits-all" solution to domain common information modeling. The generality and ambiguity of CIM, however, results in difficulties for CIM application developers to build applications. Quite often, they don't know where to start in face of overwhelming information defined in CIM. A lack of clear semantic interpretation and modeling constraints in CIM has resulted in various interoperability issues as well.

Resolution – IEC TC57 has recommended layered reference architecture for CIM applications as shown in Figure 4. It consists of three layers: *semantic*, *context*, and *syntax* layers.

At the top, the *semantic* layer contains domain information model including standard CIM and its extensions. The information model, maintained in UML, focuses on describing the domain and provides common vocabularies and semantics.

The *context* layer consists of a collection of profiles derived from the information model at the semantic layer. Each profile specifies a subset of the CIM classes and attributes specific to a business context or an application domain. For example, IEC 61970-452 [3] describes CIM Static Transmission Network Model Profiles (also referred to as *CPSM profile*). The ENTSO-E profile [18] describes a power system planning model exchange specifications. In general, a CIM profile serves as a contract or an agreement among parties that exchange the information using CIM.

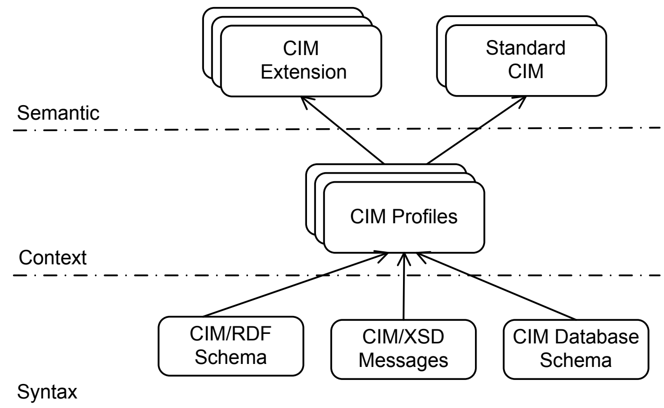

Fig.4. TC57 Layered Reference Architecture

These profiles are further converted into software industry standard formats at the *syntax* layer to drive applications, guide model consumption, and facilitate code and database schema generation. Depending on the application domains, various profile formats are generated, including RDF/OWL format for model exchange, XSD (XML Schema Definition) format for messaging, and database schema used to define CIM database.

Following TC57 CIM application design [19]-[20], the application developer can leverage the powerful CIM profiling mechanism to first create an application-specific CIM profile, and then generate the appropriate profile format to facilitate the CIM applications. A variety of modeling constraints can be added to the profile to meet the modeling requirements, such as specifying mandatory classes, attributes, and associations required to run applications, enforcing model hierarchies that map to application information model, etc. The profile formats are used to

- Guide the CIM/XML import process during which the required dataset, as defined in the profile, is extracted from the imported CIM/XML.
- Validate the imported CIM/XML against the modeling constraints specified in the profile to ensure the imported model is valid and all of the modeling requirements are satisfied.
- Generate a CIM database schema used to store the extracted CIM information.
- Generate a collection of CIM classes used in the CIM applications.

Experiences have demonstrated that, when applied properly, this model-driven approach could result in a robust and extensible CIM solution.

C. CIM Database Design

Issue – CIM databases are normally implemented as relational databases due to the industry-wide support of relational databases. However, CIM is an object oriented model, and there are some object-oriented modeling concepts that relational databases do not natively support, such as inheritance and many-to-many associations. Mapping an object-oriented CIM meta-model to a relational database schema has proved to be a challenge to many CIM database designers.

Resolution – Two different design approaches are explored and evaluated.

The left part of Figure 5 shows an inheritance hierarchy in CIM with base class at the top and subsequent sub-classes (or concrete classes, *e.g.* ACLineSegment) that normally represent a real-world entity. One of the common strategies applied in modeling inheritance in a relational database involves creating one table per class. In this example, each class, concrete or abstract, is mapped to a table as shown in Figure 5. The inheritance relationship is modeled as a typical relationship between sub-class and the base class using foreign keys. This mapping strategy is straightforward and the resulting relational model is normalized.

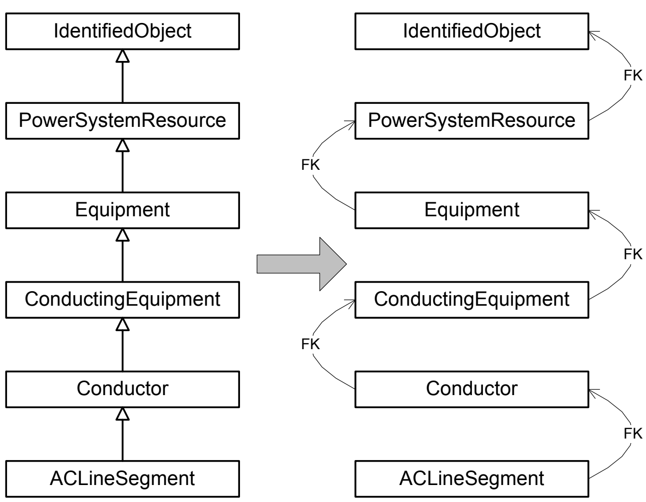

Fig.5. CIM Database Schema Based on One-Table-Per-Class Mapping

However there are following problems associated with this approach:

- Higher degrees of normalization typically involve more tables and create the need for a larger number of join operations. This can adversely impact the performance.
- Results in an ill-formed relational model. In CIM, since majority of the classes are derived from a few top-level abstract classes (*IdentifiedObject, PowerSystemResource*), the tables of these top-level abstract classes will be over-populated. Retrieving and update of a CIM object mostly involves searching a record in these over-populated tables. This results in not only poor performance, but also frequent locking of these tables in a shared environment.

An alternative mapping strategy is to define tables only for concrete classes such as ACLineSegment in the above example, *i.e.* to use a one-table-per-concrete-class approach. The defined consolidated table contains all of the attributes and associations inherited from the abstract classes. This mapping strategy overcomes the problems inherent in the one-table-per-class approach. It is also simple in structure, efficient in data access, and easy to use.

One of the problems with the one-table-per-concrete-class approach is that the resulting database schema is not well normalized. Some of the CIM associations may not be able to be translated into foreign keys, because these relationships are defined at abstract levels. To overcome this problem, auxiliary database tables or views can be constructed to facilitate the applications. For example, a database view can be defined to map a single association defined at the abstract level to multiple optional foreign keys referencing to various concrete class tables.

The auxiliary table/view approach can also be leveraged to map the many-to-many CIM associations. Many-to-many association is not often used in CIM, but it is an accepted modeling practice. The recommended practice is to add an associative table between two classes and decompose the many-to-many association into two many-to-one associations, as illustrated in Figure 6.

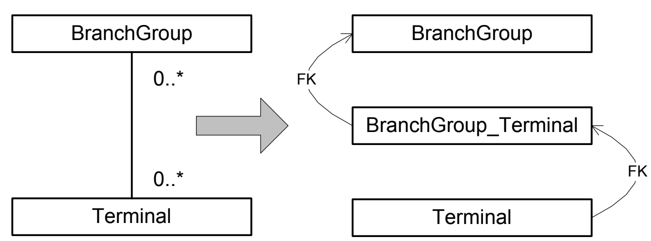

Fig.6. Mapping CIM Many-to-Many Association via Associative Table

D. CIM Validation

Issue – The IEC Standards [2]-[6] specify how CIM-compliant information should be exchanged. However, the compliance check doesn't guarantee the accuracy, integrity, and consistency of this CIM data that will be used by applications. Further data analysis is required to detect any violations of domain-specific modeling constraints, such as network topology errors.

Resolution – Schema-driven and Rule-based CIM validation have established as two promising practices in the CIM communities.

CIM-based schema languages provide meta-model frameworks for specifying the syntax, data structure, and simple modeling constraints such as cardinality restrictions of a CIM dataset. These schemas are based on the software industry standards such as RDF/OWL and XSD (both W3C standards). This makes it possible for an application developer to leverage existing off-the-shelf tools and components, and build a schema-driven CIM validation tool.

However, CIM validation, especially data quality check, is more involved, and cannot be performed in schema-driven approaches. Many domain-specific validation logics cannot be expressed in formal schema languages. A rule-based approach provides more power and flexibility for CIM validation. It allows the developer to specify validation logic in the form of *if-then* rules, and then relies on an inference engine to apply rules against data. Although the initial development effort is higher in building a rule-based validation tool, its benefits outweighs the cost in the long run.

E. Metadata-Driven Implementation

Issue – There are several ways to implement CIM-based application. A straightforward approach is to program the domain model information and business logics in applications. Such an implementation is quick and easy to accomplish, but is vulnerable to evolve and difficult to maintain. As CIM evolves and business requirements change, a tremendous effort is needed to re-engineer the solution. A CIM importer/exporter application based on a database view mapping is such an example, where model mapping between CIM and underlying proprietary information model is achieved through SQL script programming. When CIM or underlying proprietary information model changes, the existing mappings no longer work and must be reprogrammed.

Resolution – In comparison, a metadata-driven approach is adaptable and robust. In this approach, domain knowledge, including CIM, proprietary information model, and mapping between them, is separated from the software infrastructure and process control. Metadata is *data about data*, or the knowledge of what data set is stored in database, how it is encoded and structured in the exchange format, how it is mapped to other data sets, etc. A CIM/RDF schema or a CIM/XSD message definition is metadata. So is the set of modeling validation rules in a formal rule language. By separating such domain knowledge which is subject to change from the application process, a metadata-driven solution can be built to adapt to the future model evolution and meet the changing business requirements. Two examples of metadata-driven solutions are described below.

Figure 7 illustrates a metadata-driven solution of a CIM validation tool where the model validation rules are separated from validation process and stored in a rule base. During run time, the validation engine checks the input CIM data against the validation rules to detect any modeling violations. Users are able to customize the validation tool by changing the rules through a rule-base editor.

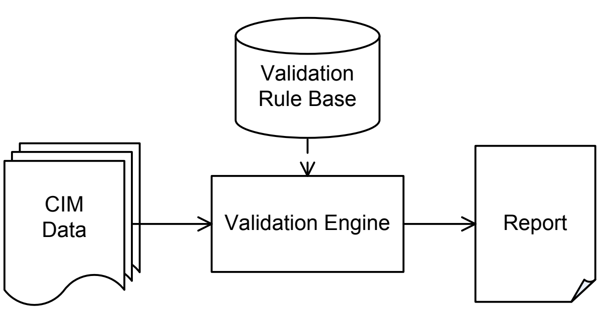

Fig.7. A Knowledge-Based CIM Validation Application

Figure 8 shows another example – a metadata-driven CIM importer/exporter. The mapping logic used to map CIM and the propriety information model is specified in a mapping profile. Guided by the mapping logic specified in the mapping profile, the model transformation engine converts CIM to the propriety information model and vice versa. In some cases, where mapping logic is too complicated or too exceptional to generalize, model transformation helpers may be developed to assist the model transformation. These model transformation helpers can be plugged into the model transformation engine through an interface. This metadata-driven implementation of CIM importer/exporter makes it extremely easy to maintain. For instance, when a new CIM version needs to be supported, majority of the work involves creating a new specification of mapping profiles. Minimum programming changes may be involved to modify the model transformation helpers.

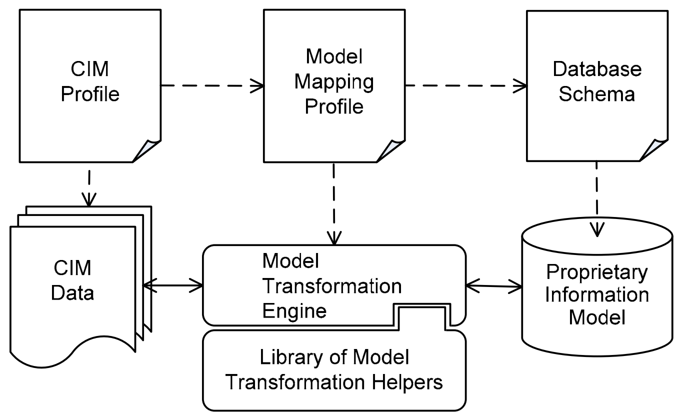

Fig.8. A Metadata-Driven CIM Importer/Exporter

F. Dealing with Large RTO/ISO CIM Models

Issue – Typically, the CIM/XML file generated by the centralized information modeler in an RTO/ISO contains information of all modeling aspects including network, generation, DTS, SCADA/ICCP, market, planning, ratings, and contingencies. This can lead to extremely large CIM models, with file size running in Gigabytes. Consuming such a large CIM/XML model file is a challenge for downstream applications.

Resolution – Our practice in dealing with large RTO/ISO model has proven that *filtering* process is an important step for consuming and analyzing bulk CIM models. A CIM model filter, as illustrated in Figure 9, scans the bulk CIM model and extracts the CIM information necessary to run the applications based on a profile specification.

Model indexing has also proved to be an effective approach to achieve high scalability for the CIM applications. By indexing the CIM objects through either database or in-memory indexing mechanisms, model information retrieval or model navigation is instant, thus significantly enhancing the performance of CIM applications.

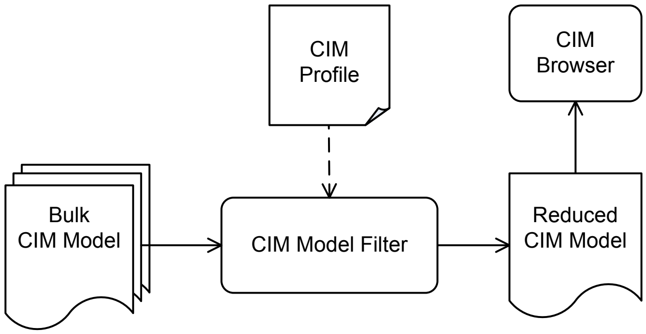

Fig.9. A Metadata-Driven CIM Model Filter

G. *Incremental CIM Model*

Issue – In a coordinated RTO/ISO modeling environment, the model changes must be properly managed and timely exchanged. For example, the transmission service providers must submit their model changes to RTO through formal processes, the RTO/ISO must provide ability to track changes from submission to implementation, and further notify other market participants of pending model changes. All this requires a need to support incremental model changes, especially in cases where the base model is very large.

With the correct model management process, two additional factors that are important in the RTO/ISO environment are (1) minimize the time taken to test the applications (usually a manual process) when model changes are introduced, and (2) minimize the down time required to load an updated model (either full or incremental model).

Resolution – CIM provides a standard approach to represent the incremental model changes. The CIM Difference Model describes the difference between a base model and its revision (or increment) on an object-by-object basis [5]. This approach is very generic. It enables users to describe arbitrary model changes. However, the generation or consumption of this generic incremental model exchange representation is challenging due to two reasons (a) the source or the target information model may not be CIM-compliant, and (b) the described model changes are arbitrary in nature. In addition, the CIM incremental may not contain adequate context information to support the model conversion if the mapping between CIM and the proprietary information model is not straightforward.

In order to support an unrestricted incremental model exchange, a full CIM model representation is essential. In the context of CIM incremental import, for example, the full CIM representation serves as an interface, and is initialized with the base model first. The CIM incremental can be then easily applied to the CIM-compliant base model, since both full model and incremental model are native to each other. Finally, a full model conversion is performed to propagate the updated full CIM to the target proprietary information model. The process is illustrated in Figure 10.

In order to minimize the time taken to test applications after a model update, one solution is to provide business users with CIM tools that present the incremental model changes in a user-friendly format.

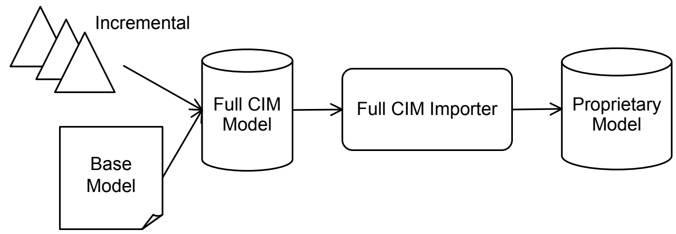

Fig.10. Importing CIM Incremental

IV. CONCLUSIONS

Developing and maintaining CIM-based applications is a challenge. This paper describes patterns, *i.e.* identifies design problems faced by CIM application developers, and proposes solutions to them as a combination of modeling guidance, architectural design, and implementation strategies. Patterns presented here are from authors' experience in implementing CIM solutions in energy markets. These real-world based principles and guidelines provide a proven starting point for CIM application designers and developers to build robust and extensible CIM applications.

V. ACKNOWLEDGMENT

The authors gratefully acknowledge the contributions of ABB and ERCOT Nodal MMS project team members.

VI. REFERENCES

[1] Guidelines for Control Center Application Program Interfaces, Technical Report, EPRI, February 1996. http://my.epri.com/portal/server.pt?Abstract_id=TR-106324

[2] Draft IEC 61970-301, Energy Management System Application Program Interface (EMS-API), Part 301: Common Information Model (CIM) Base, Revision 023, based on IEC61970CIM14v15. October 2010. http://cimug.ucaiug.org/CIM%20Model%20Releases/Forms/AllItems.aspx

[3] Draft IEC 61970-452, Energy Management System Application Program Interface (EMS-API), Part 452: CIM Static Transmission Network Model Profiles, Revision 7.04, based on CIM14v15. October 2010. (URL same as [2])

[4] Draft IEC 61970-501, Energy Management System Application Program Interface (EMS-API), Part 501: Common Information Model Resource Description Framework (CIM RDF) schema, December 2005. http://cimug.ucaiug.org/IEC%20TC57%20WG%20Draft%20Documents/Forms/AllItems.aspx

[5] Draft IEC 61970-552-4, Energy Management System Application Program Interface (EMS-API), Part 552-4: CIM XML Model Exchange Format, Revision 10, October 2009. (URL same as [4])

[6] IEC 61968 Drafts documents. (URL same as [4])

[7] IEC 61850 Drafts documents, http://iec61850.ucaiug.org/IEC61850_Drafts/IEC%2061850%20Drafts/Forms/AllItems.aspx

[8] A. deVos, S. Widergren, J. Zhu "XML for CIM Model Exchange", 22nd International Conference on Power Industry Computer Applications, Sydney, Australia, May 2001.

[9] "NIST-identified Standards for Consideration by Regulators, Release 1.0," NIST's letter to FERC Chairman Jon Wellinghoff, Oct 6th 2010, http://www.nist.gov/public_affairs/releases/smartgrid_100710.cfm

[10] "Smart Grid Architecture and Standards: Assessing Coordination and Progress", NIST Testimony of Smart Grid Interoperability, July 2010. http://www.nist.gov/director/ocla/testimony/upload/DOC-NIST-testimony-on-Smart-Grid-FINAL-with-bio.pdf

[11] 2008 CIM-XML Interoperability Including CIM-Based Tools Test, Technical Report, December 2008. http://my.epri.com/portal/server.pt?Abstract_id=000000000001018402

[12] Major 2010 Common Information Model (CIM) Interoperability Test of Power System Model Revisions, Technical Report, November 2010. http://my.epri.com/portal/server.pt?Abstract_id=000000000001017804

[13] Jun Zhu "Web Services Provide the Power to Integrate: A Web-Services-Based Framework for Integration of Power System Applications", *IEEE Power & Energy Magazine*, Nov./Dec. 2003

[14] PJM Advanced Control Center Project (AC²): Adventures in SOA, CIM, and Virtualization, John Baranowski, 2008 EMS Users Conference, September 8, 2008 www.emmos.org/prevconf/2008/02%20PJM%20EMS%20Upgrade.PPT

[15] Kamal Parekh, Joe Zhou, Kelly McNair, Greg Robinson "Utility Enterprise Information Management Strategies", Grid-Interop Forum, Albuquerque, NM, November 2007. http://www.gridwiseac.org/pdfs/forum_papers/114_127_paper_final.pdf

[16] Jun Zhu, David Lubkeman, "Object-Oriented Development of Software Systems for Power System Simulations*", IEEE Transactions on Power Systems*, Vol. 12, No. 2, May 1997.

[17] Jun Zhu, Paul Jossman, "Application of Design Patterns for Object-Oriented Modeling of Power Systems", *IEEE Transactions on Power Systems*, Vol. 14, No. 2, May 1999.

[18] ENTSO-E CIM Model Exchange Profile, 2nd Edition: Draft Version 7 – IOP Version, July 2010. http://cimug.ucaiug.org/Groups/ENTSO-E_IOP/1_SDO_July2010/default.aspx

[19] Draft IEC 61970-Part 1, Energy Management System Application Program Interface (EMS-API), Part 1: Guidelines and General Requirements, Revision 10, August 2004. . (URL same as [4])

[20] Draft technical report, IEC TC57 CIM Model Maintenance, Revision 4, February 2009. (URL same as [2])

VII. Biographies

Rahul Khare obtained his Ph.D. degree in Electrical Engineering from University of Washington in 1997. He is currently working with Ventyx, an ABB company, as a Project Lead on large-scale energy market design projects. His areas of interests are energy market operations, CIM, SCADA/EMS, Settlements and Energy Accounting. He is member of IEEE and US member of IEC TC57 WG16 technical committee. Dr. Khare can be reached at rahul.khare@us.abb.com.

Mostafa Khadem obtained his Ph.D. degree in Nuclear Engineering from University of California Berkeley in 1984. He is currently working with Ventyx, an ABB company, as a Senior Engineering Manager on applications overseeing design and developments of application for Market Management Systems. He is a US member of IEC TC57 WG16 technical committee. Dr. Khadem can be reached at mostafa.khadem@us.abb.com.

Sainath Moorty obtained his M.A.Sc. degree in Electrical Engineering from the University of Western Ontario in 1990. He is currently working at the Electric Reliability Council Of Texas (ERCOT) as Principal, Enterprise Architecture. He is responsible for the design and delivery of the ERCOT Nodal Market Management System. Mr. Moorty can be reached at smoorty@ercot.com.

Kittipong Methaprayoon obtained his Ph.D. degree in Electrical Engineering from University of Texas at Arlington in 2006. He is currently a market analyst at the Electric Reliability Council Of Texas (ERCOT). His area of interests includes application of artificial neural networks in power system, and electricity market planning and operations. Dr. Methaprayoon can be reached at kmethaprayoon@ercot.com.

Jun Zhu obtained his Ph.D. degree in Electrical Engineering from Clemson University in 1994. He is the founder of Power Info LLC, and is currently working as a consultant with Ventyx, an ABB company. Previously he worked for Nanjing Automation Research Institute (NARI), Siemens PTI, and AREVA T&D. His special fields of interest include power system modeling, interactive power grid visualization, CIM-based model exchange, and CIM-based enterprise application integration and information management. As a hobby, he built CIM tools and shared them with peers in the CIM community. Dr. Zhu can be reached at junzhu@powerinfo.us.