Common Information Model – A Developer’s Perspective

etrep02.pdf

Robin Podmore*, David Becker**, Rob Fairchild***, Marck Robinson***
(robin@incsys.com, dbecker@epri.com, rob@powerdata.com, marck@powerdata.com)

*Incremental Systems Corporation, 194th Ave. S.E., Issaquah, WA, 98027
**Electric Power Research Institute, 3412 Hillview Ave. Palo Alto, CA, 94304
***PowerData Corporation, 500 108th Avenue N.E. #960, Bellevue, WA, 98004

Abstract

The Common Information Model (CIM) defines a utility industry standard object-model for the development and integration of applications used for electric power systems engineering, planning, management, operation and commerce. This paper addresses the following questions:

• Why was it developed?
• What does it model? How is it defined?
• What is the impact on software developers?
• Who is using it?

Why was it developed?

Even though most Energy Management Systems (EMS) and Distribution Management Systems (DMS) are now supplied with standard operating systems on standard computer platform hardware, these systems are still built with proprietary databases. A proprietary database restricts access by third parties to design software and add applications. Future upgrades and add-ons that involve significant changes to the installed system must be made by the original supplier or by an in house team. Often the customer bears the full cost for such changes. By contrast, an EMS or DMS with an industry-standard database and interfaces that are openly published will enable incremental growth. The customer can select the best software without being restricted by proprietary terms or by non-standard designs.

The Electric Power Research Institute (EPRI) Working Group on Control Center Application Interfaces (CCAPI) has as its objectives to publish a set of guidelines for application interfaces, to develop associated support tools, and to promote the use of open software engineering approaches in EMS [1]. The Working Group consists of management and technical representatives from EPRI, utility companies, EMS vendors and EMS software developers. Recently, the group has been broadened to include DMS vendors and DMS software developers.

The CIM is the foundation of the overall CCAPI framework. The CIM provides a standard for representing power system objects along with their attributes and relationships. The CIM facilitates the integration of: EMS applications developed by different vendors; entire EMS systems developed by different vendors; or EMS systems and other systems concerned with different aspects of power system operations, such as generation or distribution management.

What Does it Model?

The CIM is partitioned into a number of submodels, or packages, for convenience: a Wires Model, SCADA Model, Load Model, Energy Scheduling Model and a Generation Model. The Wires Model represents physical equipment and the definition of how they are connected to each other. It includes information for Transmission, Subtransmission, Substation, and Distribution Feeder equipment. This
information is used by Network Status, State Estimation, Power Flow, Contingency Analysis, and Optimal Power Flow applications. The SCADA Model describes measurements, PTs, CTs, RTUs, scan blocks, and communication circuits. It supports operator control of equipment, telemetered data acquisition and alarming. The Load Model provides models for all load levels from customers to feeders to load areas to the system level. The load is modeled by time-varying curves that represent effects of different seasons and daytypes. The voltage and frequency dependence of loads can also be modeled. The Energy Scheduling Model includes objects for schedules, companies, control areas, and tie lines. It handles scheduling transactions for Energy, Generation Capacity, Transmission, and Ancillary Services. The Generation Model includes objects for generators, prime movers, fuels, and heat rate curves. This information is used by Unit Commitment, Economic Dispatch, Automatic Generation Control and Operator Training Simulator applications.

Focus groups are currently working on extending the CIM to handle computer and communication networks, water networks, distribution networks, asset management, and a common schematic display model.

The current scope of the CIM goes far beyond its original application in an EMS. The scope of the CIM can be best defined by paraphrasing the scope of the Object Management Group Utilities Task Force: “To provide standard objects for the inter-operation of systems and applications used for production, transmission, distribution, marketing and retailing functions of electric, water and gas utilities.”

How is it Defined?

The CIM is defined and maintained using a set of Unified Modeling Language Class Diagrams [2]. The Unified Modeling Language is an object-oriented modeling language used for system specification, visualization and documentation. UML is a way of describing software with diagrams and is a language that both users and programmers can understand. Extensions to the CIM are often introduced to the CCAPI Working Group by developing UML Use Cases. Use Cases are the most precise way for users, analysts, designers, programmers and testers to reach a common understanding of system requirements. Use Cases describe how various actors interact with the system.

As we describe a database that can be used by many different applications, the reader may conjure visions of large, complex models with class diagrams that look like a jumble of lines. Obviously, such models will not be practical to implement. The key to maintaining a model that is simple, flexible and extensible is to combine the most powerful concepts of good relational database design and good object-oriented analysis. In the past, references regarding relational databases have not considered object-oriented concepts of inheritance, aggregation or design patterns. This is changing with the advent of object/relational databases. On the other hand, references on object oriented analysis have not given weight to the relational concepts of normalization, views and independence of data from applications. Object-oriented references assume that the data is encapsulated and hidden by the object interfaces. In the real world there will continue to be procedural applications which require access to global data in relational databases even though newer applications may be object oriented.

In this section we will describe how the CIM has been designed in a manner that exploits the most powerful concepts of relational databases and object-oriented analysis. For the purpose of this discussion, we will assume that each class in the CIM object model is represented by a table in a relational database, and that each object or instance of a class in the CIM object model is represented by a row in a relational table.

The CIM class diagram describes the types of objects in the system and the various kinds of static relationships that exist among them. There are three principle kinds of static relationships:

- Associations (A Terminal is connected to a connectivity node)
- Generalization and subtypes (A switch is a type of conducting equipment)
- Aggregation (A winding is part of a transformer)

Generalization or inheritance is a powerful technique for simplifying class diagrams. The primary use of generalization in the CIM is shown in Figure 1.

0-7695-0001-3/99 $10.00 (c) 1999 IEEE
Figure 1: Generalizations for power system resource and conducting equipment

By defining a PowerSystemResource class, the attributes and relationships for this class can be inherited by all the other subclasses. The PowerSystemResource class is used to describe any physical power system object or grouping of power system physical objects that needs to be modeled, monitored or measured. All the subclasses of PowerSystemResource inherit the following relationships:

- PowerSystemResource “measured by” Measurement
- PowerSystemResource “owned by” Company

The ConductingEquipment class is used to define those objects that conduct electricity. The connectivity of these objects is specified by inheriting the following relationships as shown in Figure 2:

- ConductingEquipment “has” Terminals
- Terminal “is connected to” ConnectivityNode
- Connectivity Node “is member of” TopologicalNode

The use of Aggregation is illustrated by the transformer model in Figure 3. The aggregation relationship specifies that:

- A Transformer “has” one or more windings
- A Transformer Winding “has” 0, 1 or 2 Tap Changers

The CIM is very flexible about how different aggregation relationships can be defined. Figure 4 shows the aggregation relationships that are used in the EPRI Operator Training Simulator database and in the databases of several EMS vendors.
grouped into zero, one, or several containers. For example, a switch could have the following relationships:

- Switch “is member of” substation
- Switch “is member of” transmission line
- Switch “is member of” Feeder

The CIM has a number of patterns that can be used to model different real-life situations. The CIM measurement pattern is shown in Figure 5.

![Figure 5: Measurement Pattern](image)

This pattern can be used to model a complete range of relationships between physical objects and their measurements, calculated values, and manually-entered values.

In the CIM, all the state information for objects is stored in the MeasurementValue table. The MeasurementValue table thus contains a heterogeneous collection of different types of measurements (e.g., MW, Mvar, temperature, voltage, frequency, and pressure) for a wide variety of equipment (e.g., transformers, pumps, capacitors, generators, etc.). The MeasurementValue table contains real-time data and solutions from different applications (e.g. state estimator, powerflow user#1, and powerflow user #2). However, the user and the application developer will usually want to create a view with that state information appearing as specific attributes along with the corresponding objects. Views are described later in this section.

As the CIM is extended to include other models, such as computer networks, water networks, gas networks, and asset management, the use of additional patterns will be important to maintain its simplicity. A controller pattern with measurements as inputs, and controls for PowerSystemResource as outputs, would be useful, as would an authorization pattern in which organizational units and people have the authority to operate a PowerSystemResource.

The CIM is not dependent on any particular implementation. It can be implemented as a relational database, as an object database, or as an object/relational database. However, the CIM classes have been defined to ensure that a relational implementation will be normalized [3]. Normalization is the process of deconstructing a class with many attributes into many classes, each with fewer attributes in order to eliminate redundant data and to avoid problems with inserting, updating, or deleting data. In object-oriented analysis, two important steps are the analysis of the use cases and the identification of all nouns used. These nouns become candidates for classes. Relational database normalization corresponds to defining classes for all the important nouns in the use cases. The main advantage of normalization, or of finding nouns that can be used as classes, is defining a model that is not dependent on any single application. The classes, or tables, contain only the attributes which are inherent properties of the corresponding objects. Attributes which are properties of related objects are stored with their corresponding objects.

In relational database terms, a view is a virtual table that is composed of fields of one or more data tables. In relational databases a clear distinction is made between the base tables and views [3]:

- A base table is a “real” table, i.e., a table that physically exists in the sense that there physically exists stored records that directly represent that table.
- By contrast, a view is a “virtual” table, i.e., a table, composed of records from other, “real” tables, that does not actually exist in physical storage but is rendered to look like a “real” table. Views can be thought of as different ways of looking at the “real” tables.

The CIM is defined so that the base tables, or base classes, are normalized. In other words, they include only the attributes that are inherent properties of the object. A user or application will typically require a view that joins attributes from multiple tables. The view for a transformer that would be of interest to a power systems application engineer is shown in Table 1.
### Table 1: View class for transformer for use by power systems engineer

<table>
<thead>
<tr>
<th>View Class</th>
<th>View Attribute</th>
<th>CIM Class</th>
<th>CIM Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transformer</td>
<td>Name</td>
<td>PowerSystem</td>
<td>Resource</td>
</tr>
<tr>
<td>Transformer</td>
<td>FromBusRef</td>
<td>TopologicalNode</td>
<td>Name</td>
</tr>
<tr>
<td>Transformer</td>
<td>ToBusRef</td>
<td>TopologicalNode</td>
<td>Name</td>
</tr>
<tr>
<td>Transformer</td>
<td>TapRatio</td>
<td>Measurement</td>
<td>Value</td>
</tr>
<tr>
<td>Transformer</td>
<td>LowStep</td>
<td>TapChanger</td>
<td>LowStep</td>
</tr>
<tr>
<td>Transformer</td>
<td>HighStep</td>
<td>TapChanger</td>
<td>HighStep</td>
</tr>
<tr>
<td>Transformer</td>
<td>StepSize</td>
<td>TapChanger</td>
<td>StepSize</td>
</tr>
<tr>
<td>Transformer</td>
<td>Status</td>
<td>Measurement</td>
<td>Value</td>
</tr>
<tr>
<td>Transformer</td>
<td>FromMW</td>
<td>Measurement</td>
<td>Value</td>
</tr>
<tr>
<td>Transformer</td>
<td>FromMvar</td>
<td>Measurement</td>
<td>Value</td>
</tr>
<tr>
<td>Transformer</td>
<td>ToMW</td>
<td>Measurement</td>
<td>Value</td>
</tr>
<tr>
<td>Transformer</td>
<td>ToMvar</td>
<td>Measurement</td>
<td>Value</td>
</tr>
</tbody>
</table>

In order to obtain rapid access and updates, application programs will often physically store the denormalized views as data structures within their local program space. This will work as long as there is a mechanism for keeping the local data synchronized with the CIM base tables. The most modern object/relational database management systems allow views to be created rapidly. These dynamic views can be updated in real time, and they allow objects to be updated, inserted and deleted [4].

### Impact on Software Developers

The widespread adoption of the Common Information Model is a significant benefit to software developers. Software developers can build their applications to run with the CIM with the assurance that there will be a growing base of customers and of systems integrators who will benefit from such work. The CIM, and the commercial tools that support the CIM, give software developers a head start for building their applications. As an example, the PowerDeveloper CIM Software Developers Kit provides the following [4], [5]:

- Openly published data models
- Graphical tools to change models and to view application results.
- A real-time relational database.
- Integrated simulator for testing applications under simulated, real-time conditions.
- Test suites of hypothetical and actual power system models.
- Historical recording of real-time analog and status data with correlation to actual models.
- Power system schematic one-line diagrams.

The CIM is maintained using Rational Rose. Rose supports generation of JAVA and C++ header files. Rose also supports generation of Data Definition Language for relational database models. Software developers can build applications using the CIM in a number of languages. Those familiar with developing object-oriented applications can program in object-oriented languages using the CIM base classes. Developers of procedural applications can define views of the CIM that contain the attributes which are tailored to their applications.

### Who is using it?

The U.S. coverage by EMS vendor network application models is a patchwork quilt. In the U.S. and Canada twenty three Security Coordinators have been delegated by NERC with the responsibility of performing real-time network analysis functions and monitoring the overall security for the system in their regions. Of the twenty-three Security Coordinator systems, seven systems are provided by ESCA, six systems by ABB, four by Siemens, and four by GE Harris. The database formats and data dictionaries for the operational models provided by the different EMS vendors are proprietary and non-standardized. Each vendor has used inconsistent data formats, as well as unique databases and libraries, though each vendor’s system basically represents the same information.

Security Coordinators need to build their operational models by exchanging model data with their underlying control areas, and then exchange their operational models with each other. The CIM provides the ideal solutions to these needs. Six Security Coordinators have projects underway to model their systems using the CIM. Other Security Coordinators are expected to begin such efforts in 1999. It is
possibility that the three major interconnections in
the U.S. and Canada will be modeled using the
CIM by the end of 1999.

The CIM is endorsed by the following
organizations: EPRI, IEC TC57 Working Group
13 on Energy Management Systems, IEC TC57
Working Group 14 on Distribution Management
Interfaces, NERC Data Exchange Working
Group and the Utility Task Force of the Object
Management Group.

All of the major EMS vendors have agreed that
they will support an interface between their
proprietary databases and the CIM.

Summary

The Common Information Model (CIM) was
originally developed to support the integration
of multi-vendor applications into Energy
Management Systems (EMS). By combining the
most powerful concepts from relational
databases, such as normalization and views, with
the most powerful concepts from object-oriented
analysis, such as generalization, aggregation,
and design patterns, the CIM supports the
integration of a wide range of utility systems
and applications. Model data can be entered
once into the CIM and all the connected systems
that use the data will be updated automatically
in near real time.

NERC sponsorship of the CIM as the standard
for exchanging operational power system models
between control areas and security coordinators
has given impetus to wide-spread deployment of
the CIM by users and vendors throughout North
America.

Software developers can now develop CIM-
compliant applications without having to be
concerned about proprietary interfaces for the
myriad of vendor systems. System Integrators
can now integrate applications from multiple
suppliers without having to build customized
interfaces for each application. And end users
benefit by having more choices of applications.

References

[1] EPRI. “Guidelines for Control Center Application
Program Interfaces.” Final Report for Project 3654-01,
Corporation”.
Corporation”.

Common Information Model – A Developer’s Perspective (etrep02.pdf)