Research Journal of Applied Sciences, Engineering and Technology 6(14): 2561-2567, 2013
DOI:10.19026/rjaset.6.3738
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: December 26, 2012 Accepted: February 01, 2013 Published: August 10, 2013

Research Article

Research on CIM Update and Extension Based on Ontology

Cao Zhe and Liu Dong
Key Laboratory of Control of Power Transmission and Conversion, Ministry of Education, School of Electronic, Information and Electrical Engineering, Shanghai Jiao Tong University, 200240, China

Abstract: The problem of model difference exists in the application integration based on CIM. Difference caused by diverse versions will lead to semantic conflict and inconsistency, which make data sharing impossible. To solve this problem, we need to perform research on CIM update and extension. Therefore we propose a CIM update and extension research method based on ontology. In this study the CIM update and extension are divided into four scenarios, add, delete, change and rename and corresponded algorithm are discussed. Finally, we conduct an experiment based on real model and compare the results with the ones did manually to verify this research's effectiveness and practicality.

Keywords: CIM, IEC 61968, OWL, update and extension

INTRODUCTION

Standardized and open power system information is the basis for application integration. IEC61968/70standards not only describes the semantics of the information to be exchanged between different applications, but also provide the maximum flexibility and scalability for data exchange and utilization (Nielsen and Nielsen, 2010). As the gradual standardization of the power system, a series of information integration has been practiced at home and abroad (King, 2009; Liu *et al*., 2008; Singh, 2009). However, the problem of model difference exists in application integration based on CIM. There are two main reasons. First, CIM is evolving with the application requirements, leading to frequent version updates (Cao *et al*., 2011; Liu *et al*., 2012). Second, different enterprises may privately expand CIM according to their demand, which may cause semantic conflict and inconsistency (Liu *et al*., 2012; Popovic *et al*., 2007). To summarize, the reasons that cause model difference are update and extension. Model semantic difference will lead to semantic conflict and inconsistency which will disable the ability of data sharing among components. Therefore fast and accurate model difference analysis has significance in engineering practice (Uslar *et al*., 2010).

By utilizing the results of difference analysis, electric power company technical staff can maintain CIM more easily and make demands on different manufactures' application integration system according to actual demand. On the other side, considering that manufactures always develop application integration system based on one version of CIM, the results of difference analysis can contribute to system maintenance as well as shorten the time for system updating and upgrading.

Reference (Yu *et al*., 2012) proposed a method for CIM consistency validation by parsing power grid metadata information from exchanged message among different applications and comparing metadata with unified CIM to analyze syntax's compatibility and semantics' consistency. However, the research lays particular stress on metadata semantics verification (Li *et al*., 2010) without deep study on CIM itself.

In order to solve the problem of model difference, we propose a CIM update and extension research method based on ontology. When build new systems based on a newer version of CIM, we also need to update the original system to map this new CIM. But, before that, the differences between these two different CIM versions should be detected. Considering CIM is a quite large model, if we find the differences manually, the time costs is huge and there will be many possible errors in the final results. Therefore, fast and accurate different analysis is a necessary pre-requisite. In this study, the CIM update and extension are divided into four scenarios. Through the proposed analysis method, we can easily know what classes are new in the new version CIM, what classes are deleted, what classes' attributed and relations have changed and what classes have renamed. According to these results, we can eliminate semantic conflict caused by various CIM

Correspondong Author: Cao Zhe, Key Laboratory of Control of Power Transmission and Conversion, Ministry of Education, School of Electronic, Information and Electrical Engineering, Shanghai Jiao Tong University, 200240, China

versions and proprietary extension and lay a foundation for seamless integration of service information (Lu *et al*., 2010).

CIM UPDATE AND EXTENSION SCENARIOS

Consider two different versions of CIM, the original model $Mo = \{c^o_1, \ldots, c^o_{o1}\}$ and the updated model $M_u = \{c^u_1, \ldots, c^u_{U\,1}\}$, where each c_i is either an original CIM class or updated CIM class. There are two types of CIM class, normal class and enumeration and c_i can be any kind. The normal class contains information of attributes and relations, while enumeration class contains only attributes information. Consider an original CIM M_{o1} and an updated CIM in Fig. 1 and 2.

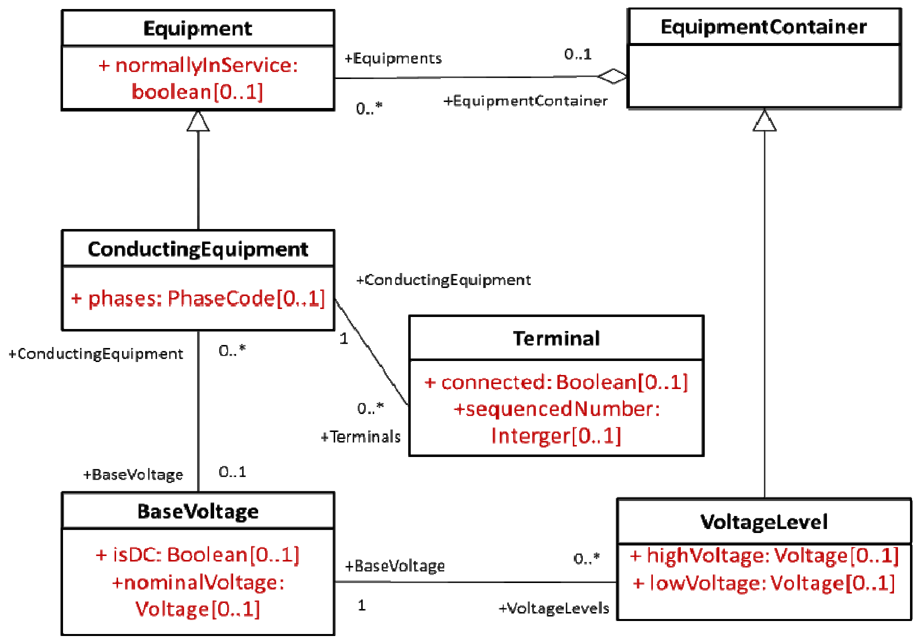

Fig. 1: Original class diagram

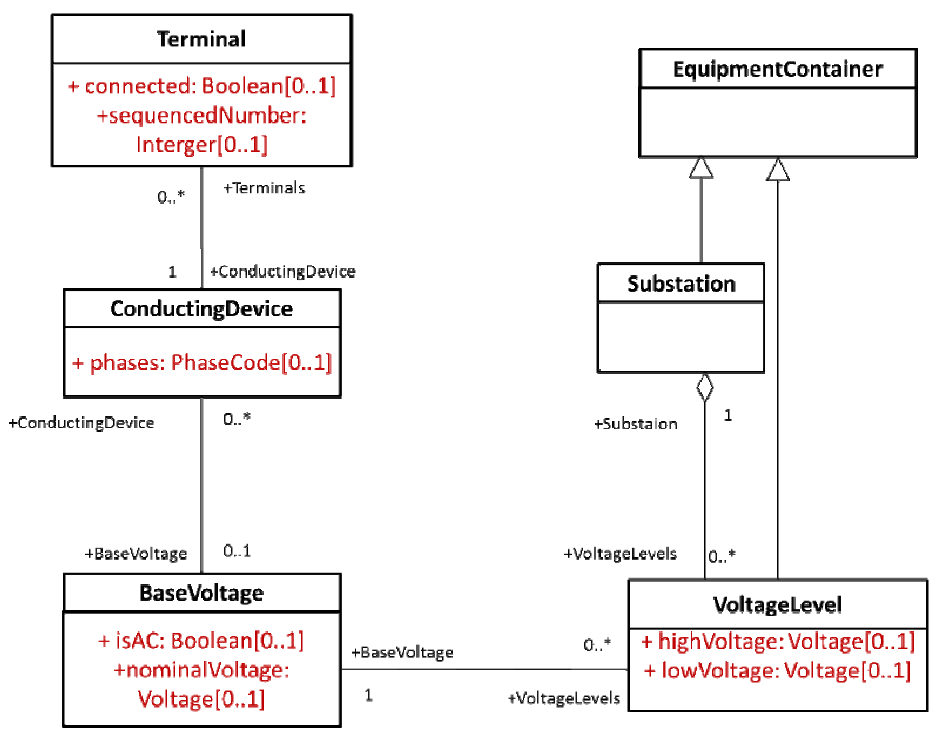

Fig. 2: Updated class diagram M_{u1}

M_{o1} = {Equipment, Base Voltage, Equipment Container, Voltage Level, Conducting Equipment}

M_{u1} = {Voltage Level, Terminal, Equipment Container, Base Voltage, Substation, Conducting Device}

Each class may have attributes and the lines between classes represent relationships. Basically, there are three types of relationships, *inheritance*, *association* and *aggregation* in CIM. In Fig. 1, class Conducting Equipment has an inheritance relationship with class Equipment, which defines Conducting Equipment as being a subclass of Equipment. As a subclass, it inherits all the attributes of its parent, but can also contain its own attributes. Conducting Equipment also has an association relationship with class Base Voltage. At each end of the association link is the multiplicity. For the Conducting Equipment-Base Voltage association, these indicate that a Base Voltage must have from 0 to many (0..*) Conducting Equipments, while a Conducting Equipment can have from 0 to 1 (0..1)Base Voltage. Furthermore, class Equipment has an aggregation relationship with class Equipment Container, indicating that Equipment Container is a container class for class Equipment. The multiplicity on the diagram operates in the same manner as to that of the association.

In this study, we identified several CIM update and extension scenarios referring to the updated model M_{u1} and original model M_{o1}.

Add and delete: Class Substation is an added class defined in mode M_{u1}. Class Substation has an aggregation relationship with class Voltage and is a subclass of class EquipmentContainer. Also, class Equipment has been deleted from model M_{o1}. We can conclude that if a class is deleted, all the relationships about this class will also be removed.

In particular, according to the definition of add and delete scenario, class Conducting Equipment in M_{o1} and class Conducting Device in M_{u1} should also be classified as add and delete scenario. However, considering the attributes of these two classes are identical, we define these two classes as a new scenario, which will be discussed later in this section.

Change: During the evolving process, from the original model to the updated one, if a class's attributes or relationships change, we define it as change scenario.

Considering class BaseVoltage, the first attribute of it changes, from *isDC:Boolean* to *isAC:Boolean*. Therefore, class BaseVoltage belongs to change scenario. Moreover, class EquipmentContainer and class VoltageLevel establish new relationships with added class Substation*,* so these two classes are also changed classes.

Rename: As previously stated, though class Conducting Equipment and class Conducting Device have different names, these two classes' attributes and relationships are identical, we define this scenario as rename. To judge whether two classes belong to rename scenario, we need to calculate two classes' similarity value. Define similarity value as follows:

For two classes $C_o = \{P_{o1},\ldots, P_{oi}, A_{o1},\ldots,A_{oj}\}$ and $C_u = \{P_{u1}, \ldots, P_{um}, A_{u1}, \ldots, A_{un}\}$, C_o contains i attributes and j relationships, C_u contains m attributes and n relationships, define the similarity value between C_o and C_u:

$$\mathrm{SIM}\left(C_o, C_u\right) = \frac{2*|C_o \cap C_u|}{i+j+m+n} \tag{1}$$

where, $|C_o \cap C_u|$ represents the total number of same attributes and relations between classes.

CIMUPDATE AND EXTENSION ANALYSIS METHOD

The flow chart of CIM update and extension analysis is shown in Fig. 3. It consists of two major steps. First, convert XMI to OWL (Motik *et al*., 2009). Second, difference analysis based on ontology.

Convert XMI to OWL: CIM is maintained by Enterprise Architect (EA), the class diagrams can export to XMI format. Therefore this step involves transition among three basic languages, UML (Unified Modeling Language), XMI (Metadata Exchange Language) and OWL (Web Ontology Language).

Parsing XMI: While the XMI standard purports to facilitate the interchange of UML models, it has been largely ineffective in practice. There are two technical reasons for this. First and foremost, XMI attempts to solve a technical problem far more difficult than exchanging UML models; it attempts to provide a mechanism for facilitating the exchange of any language defined by the OMG's Meta model Object

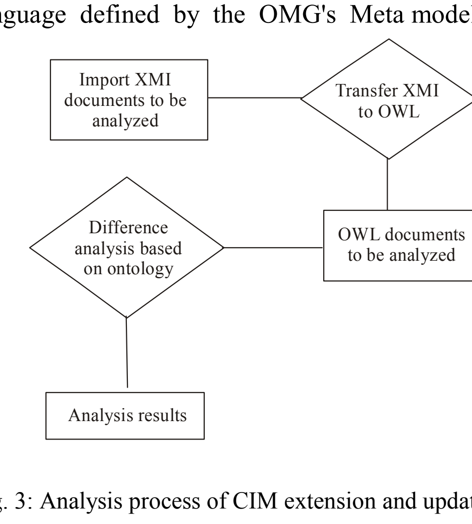

Fig. 3: Analysis process of CIM extension and update

Table 1: Mapping between UML and OWL concepts

UML	OWL
Class	Class
Specializes	Sub class of
Generalizes	Inverse sub class of
Association	Object property
Unidirectional association	Functional property
Unidirectional association (reversed)	Inverse functional property
Aggregation	Object property with annotation
Composition	Object property with annotation

Table 2: Node representation of relative OWL concepts

Node name	OWL Description	Object type*
Data type node	Data type property	NP
Enum property node	Enum property	NP
Aggregate node	Aggregate property	NA
Composite node	Composite property	NA
Functional node	Functional property	NA
Inverse node	Inverse property	NA
Sub class node	Subclass property	NA
Super class node	Super class property	NA
Property node	Property	NA
Individual node	Individual	EP

(*NP stands for normal class's attribute, NA stands for normal class's relationship, EP stands for enumeration class's attribute)

Table 3: Node representation of class ConductingEquipment

Class name	ConductingEquipment
Enum property node	phases: phase code
Functional node	Base voltage
Inverse node	Terminals
Super class node	Equipment

Table 4: Node representation of enumeration class coolant type

Class name	Coolant type
Individual node	Air
Individual node	HydrogenGas
Individual node	Water

Facility. Secondly, the UML 2.x Diagram Interchange specification lacks sufficient detail to facilitate reliable interchange of UML 2.x notations between modeling tools.

In order to parse XMI, we construct an XML parser to recognize constructs of interest while ignoring surrounding syntax. When a construct is recognized, corresponding statements are inserted into an OWL/RDF model. The XMI id's (rather than the human readable, modeling names) are used to link up scraps of UML gleaned from different parts of the XMI document.

A translation phase then renames each XMI id for the modeling name and assigns a namespace. In other words, the URI reference is constructed for each class and property.

Convert UML to OWL: After parsing XMI, considering the conceptual similarity between UML class diagram and OWL, we can convert UML class diagram into OWL. The mapping between UML and OWL concepts is shown in Table 1.

Difference Analysis based on ontology:

Pre-processing: It can be concluded from previous analysis that CIM is made up of classes and relationships among them. A class can be divided into a normal class or an enumeration class. A normal class contains attributes and relationships, while an enumeration class contains only attributes. During the process of difference analysis based on ontology, we define *node* to represent OWL-related concepts, these concepts are organized in Table 2.

With reference to the above definition, the node description for class ConductingEquipment in M_{o1} can be shown as Table 3 represents. By the way of node description, a normal class'sattributes and the relationships associated with it can be accurately and completely described.

Similarly, an enumeration class contains only attributes and each attribute corresponds to an *IndividualNode*. Take enumeration class CoolantType (cooling mode) in CIM 61970 Wires package for example, CoolantType contains 3 attributes and Table 4 depicts its node description.

Algorithm description: The algorithm discussed in this section is based on class. Considering that a normal class contains both attributes and relationships while an enumeration class contains only attributes, we assume enumeration class has 0 relationships. Figure 4 depicts the entire algorithm whose concrete steps are described below.

- Pretreat two different CIM versions to obtain original model's class set and updated model's class set, the data structure for each class is shown in Fig. 3.
- Traverse two class sets, compare class names in original and updated model and classify the class set as new name class set, removed class set and same name class set.
- For the new class set and removed class set, calculate the similarity value of each two classes, if the similarity value exceeds threshold, defined as the rename scenario. Through this step, we obtain three sets, Add, Delete and Rename.
- For the same name class, compare corresponded attributes and relationships to obtain same and Change sets.
- Obtain difference results, Add, Delete, Rename and Change sets.

CASE STUDY

Data source: We conduct the test using IEC standard iec 61968 cim 10v28.eap and iec 61968 cim 10v 31.eap. 10 v28 represents IEC standard the cim10's 28^{th}revision. We export standard XMI file using Enterprise Architect (EA) tool. The export options are as follows:

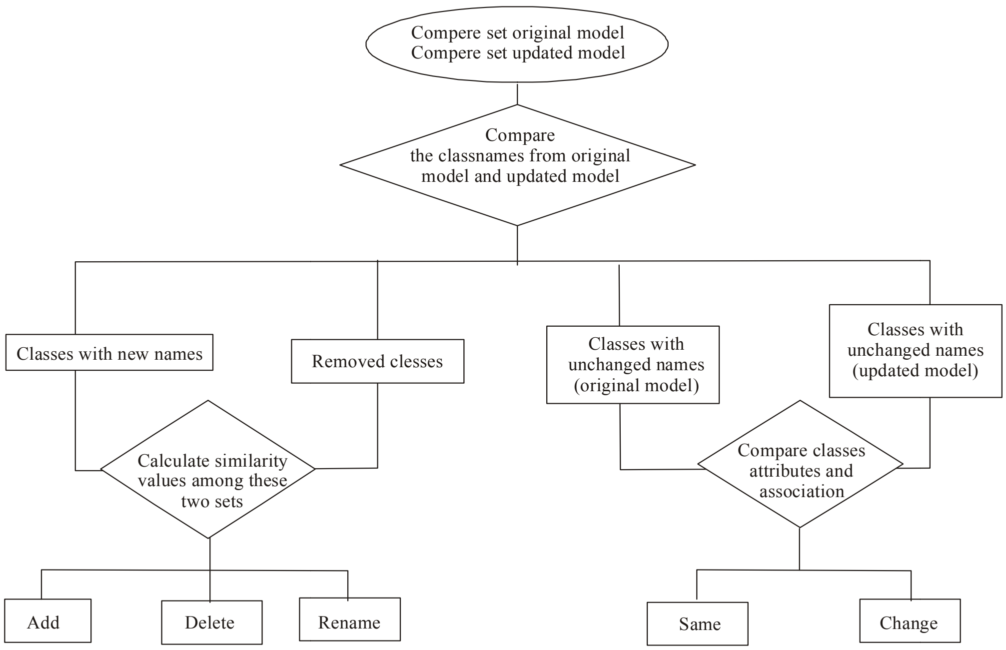

Fig. 4: Algorithm description

```
</IEC61968/Informative/InfCommon#DocDocRole>
    a      owl:Class ;
    rdfs:comment "在文档和其它文档之间充当的角色";
    rdfs:isDefinedBy </IEC61968/Informative/InfCommon#Package_InfCommon> ;
    rdfs:label "DocDocRole" ;
    rdfs:subClassOf </IEC61968/Informative/InfCommon#Role> ;
    uml:id  "EAID_D149FEA0_ACBE_4941_B93C_492180A7139B" .
```

```
</IEC61968/Informative/InfGMLSupport#GmlLabelPlacement.displacementX>
    a      owl:FunctionalProperty , rdf:Property , owl:DatatypeProperty ;
    rdfs:comment "给出距主几何点的X方向的位移来渲染一个文本标签";
    rdfs:domain </IEC61968/Informative/InfGMLSupport#GmlLabelPlacement> ;
    rdfs:isDefinedBy <Package_InfGMLSupport> ;
    rdfs:label "displacementX" ;
    rdfs:range xsd:string ;
    uml:hasStereotype uml:attribute ;
    uml:id  "EAID_5688B962_EABA_46cf_BE69_70F0D7E3903B" .
```

```
</IEC61968/Customers#CustomerAgreement.ServiceLocations>
    a      owl:ObjectProperty ;
    rdfs:comment "All service locations regulated by this customer agreement." ;
    rdfs:domain </IEC61968/Customers#CustomerAgreement> ;
    rdfs:isDefinedBy </IEC61968/Customers#Package_Customers> ;
    rdfs:label "ServiceLocations" ;
    rdfs:range </IEC61968/Customers#ServiceLocation> ;
    uml:id  "EAID_C20149AA_BEFF_43c5_B26A_F75F414F345B-B" ;
    owl:inverseOf </IEC61968/Customers#ServiceLocation.CustomerAgreements> .
```

Fig. 5: XMI parsing result

- Enable full EA Roundtrip = unchecked
- XMI Type = UML 1.4 (XMI1.2)
- Unisys/Rose Format = checked
- Export diagrams = unchecked

After exporting, we obtain two XMI files, 61968cim10v28.xmi and iec61968cim10v31.xmi.

Parsing XMI: The section describes parsing XMI document. We utilize conceptual similarity between UML and OWL to extract useful information.

Figure 5 shows the XMI parsing results. Corresponding information about Class, DataProperty and ObjectProperty is listed in this figure. Class contains class name and the inheritance relation.

Table 5: CIM extension and update analysis's statistical result

	Total No.	Add	Delete	Change	Rename
Original model	757	15	93	65	11
Updated model	679				

Table 6: Analysis result of class EndDeviceAsset

EndDevice Asset	Node name	Description
Delete	Property node	ElectricalInfos
Add	Data type node	Phase count: interger
	Data type node	Rated Voltage: Voltage (V)
	Data type node	Rated current: Current Flow (A)

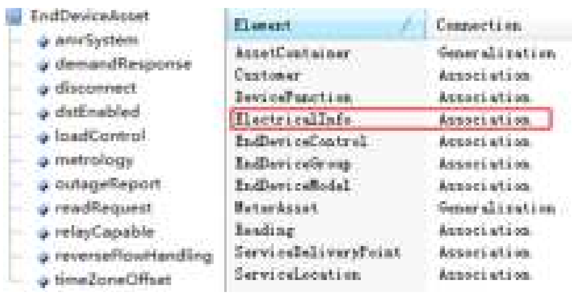

Fig. 6: Detail information of original model

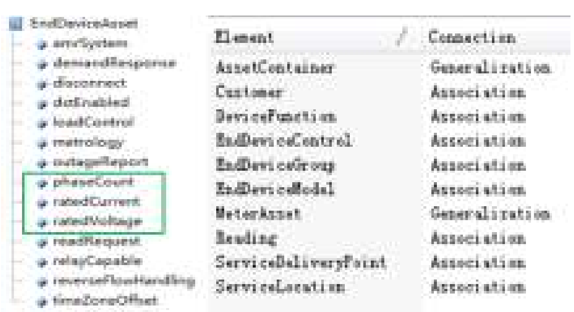

Fig. 7: Detail information of update model

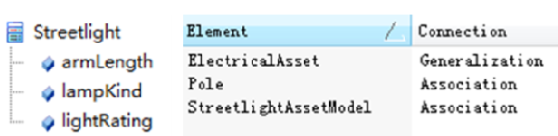

Fig. 8: Class Streetlight detail information of original model

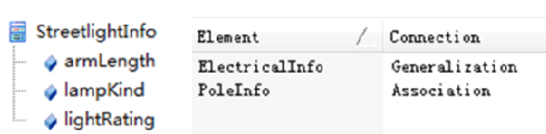

Fig. 9: Class StreetlightInfo detail information of original model

DataProperty contains domain and range information. ObjectProperty represents relationships between classes.

RESULT ANALYSIS

Statistical data is shown in Table 5. The original model contains a total of 757 normal classes and the updated model contains 679 normal classes. During the process of evolution, 15 classes belong to add scenario, 93 classes belong to delete scenario, 65 classes have changed and 11 classes are renamed.

Change scenario: Class EndDeviceAssetis listed in Fig. 6 which belongs to change scenario. According to the algorithm analysis, class EndDeviceAsset in original model removed the relationship with class Electrical Infos, meanwhile class EndDeviceAsset in updated model added three attributes which are depicted in Table 6.

In order to verify whether the program analysis results are correct, we use the EA tool to compare EndDeviceAsset information in two CIM revisions by human. In Fig. 6 and 7, the red box shows removed relaionships from the original model and the green box shows new attributes, therefore program analysis results are in line with the real data.

Rename scenario: Via algorithm analysis, we found class StreetlightInfo in original model and class Streetlight in updated model belongs to rename scenario. Similarly, we will compare algorithm analysis with information in EA tool.

Even though these two classes have different names, their attributes are same and their relationships are identical. By calculating similarity value between these two classes, it can be concluded that these two classes are in line with rename definition (Fig. 8 and 9).

By comparing difference results with actual model, we find that the accuracy of our algorithm is over 99% and the average processing time is about 20 seconds. These data demonstrated our proposed difference analysis method can quickly and accurately deal with differentiated model. With respect to the huge time and physical resource cost by manual analysis, our proposed method has huge advantages.

CONCULSION

In this study, aiming at the problem of model difference in the application integration based on CIM, we propose a CIM update and extension research method based on ontology. This method can effectively detect the variation between the models accurately and efficiently avoiding possible errors and huge time costs by manual analysis.

For future work, we will extend our work to develop visual interface and realize CIM metadata management.

ACKNOWLEDGMENT

The authors would like to thank the support of the National High-tech R&D Program of China (863 Program with issue number: 2012AA050803) and IEC

61968 interoperability testing program sponsored by distribution working group of National Power System Management and Information Exchange Standard Committee of China.

REFERENCES

Cao, Y., J. Yao, S. Yang, H. Jiang and Z. Gao, 2011. Latest advancements of smart grid core standard IEC 61970. Autom. Elec. Power Syst., 35(17): 1-4.

King, L., 2009. Enterprise service bus implementation profile: Integration using IEC 61968. EPRI Technical Report 1018795 [R].`

Li, X., H. Zhicheng and X. Zhong, 2010. Mapping analysis of IEC 61850 and IEC 61970 semanteme model based on ontology. Low Volt. Appar., 21: 57-61.

Liu, H.T., Z. Jianghe and S Jian, 2008. ESB based information integration of distribution automation and management system. Autom. Elec. Power Syst., 32(16): 47-51.

Liu, H.X., W. Fubao, D. Daxing, Y. Hongchun, W. Chun and H. Weiguo, 2012. Extended common information model for microgrid energy management system. Autom. Elec. Power Syst., 36(6): 45-51.

Lu, Y.M., L. Dong, L. Jinsong, H. Yuhui, L. Wanshui and G. Jianwe, 2010. Information integration demand and model analysis for smart distribution grid. Autom. Elec. Power Syst., 34(8): 1-4.

Motik, B., P.F. Patel-Schneider and B. Parsia, 2009. OWL 2 Web Ontology Language Structure Specification and Functional-Style Syntax, Retrieved from: W3C. http://www. w3. org/TR/owl2-syntax/.

Nielsen, T.D. and T.D. Nielsen, 2010. CIM interoperability challenges. Proceeding of Power and Energy Society General Meeting, Utility Integration Solutions, Inc. (UISOL), Lafayette, CA, USA, pp: 1-5.

Popovic, D.S., E. Varga and Z. Perlic, 2007. Extension of the common information model with a catalog of topologies. IEEE T. Power Syst., 22(2): 770-777.

Singh, U., 2009. Standard based intelligent approach for smart grid integration. Power Syst. Technol., 33(15): 30-36.

Uslar, M., S. Rohjans, M. Specht and J.M.G. Vázquez, 2010. What is the CIM lacking?. Proceeding of Innovative Smart Grid Technologies Conference Europe (ISGT Europe), OFFIS, Inst. in Oldenburg, Oldenburg, Germany, pp: 1-8.

Yu, Y., L. Dong, L. Yiming and G. Jianwei, 2012. Consistency validation for IEC 61968 information model based on ontology. Autom. Elec. Power Syst., 36(14): 46-51.