An Improvement on Data Interoperability with Large-Scale Conceptual Model and Its Application in Industry

Lan WANG[1], Shinpei HAYASHI[2], Motoshi SAEKI[2]

[1]Toshiba Corporate R&D Center, 1 Komukai-toshiba-cho, Saiwai-ku,Kawasaki, 212-8582, Japan. Lan.wang@toshiba.co.jp
[2]Tokyo Institute of Technology, Ookayama 2–12–1, Meguro-ku, Tokyo 152–8552, Japan.

Abstract. In the world of the Internet of Things, heterogeneous systems and devices need to be connected. A key issue for systems and devices is data interoperability such as automatic data exchange and interpretation. A well-known approach to solve the interoperability problem is building a conceptual model (CM). Regarding CM in industrial domains, there are often a large number of entities defined in one CM. How data interoperability with such a large-scale CM can be supported is a critical issue when applying CM into industrial domains. In this paper, evolved from our previous work, a meta-model equipped with new concepts of "PropertyRelationship" and "Category" is proposed, and a tool called FSCM supporting the automatic generation of property relationships and categories is developed. A case study in an industrial domain shows that the proposed approach effectively improves the data interoperability of large-scale CMs.

Keywords: conceptual modeling, data interoperability, property relationship

1 Introduction

In the world of the Internet of Things, various systems and devices need to be connected. One of the key issues is the data interoperability among systems and devices, i.e., systems and devices can exchange and interpret data automatically. A well-known approach is to build a conceptual model (CM) that unambiguously defines entities and their relationships so that different systems and devices can utilize the CM to exchange and interpret data. For example, a CM is defined in IEC 62264 [1], [2], [3] series for enterprise-control system integration, and the Common Information Model (CIM) is defined in IEC 61970/61968/62325 for smart grid systems to exchange data among different applications and systems. For electro-electronic systems, CM defined in IEC 61360-4 (also called IEC CDD) is utilized among semiconductor management systems to exchange data.

One of the characteristics of a CM in industrial domains is that a CM normally has a large number of entities. Such as in the CIM, there are more than 1000 classes and thousands of properties. In order to improve data interoperability with such large-scale CMs, property relationships in a CM need to be created so that data among involved properties can be systematically exchanged and interpreted. Furthermore, as

In: C. Cabanillas, S. España, S. Farshidi (eds.):
Proceedings of the ER Forum 2017 and the ER 2017 Demo track,
Valencia, Spain, November 6th-9th, 2017,
published at http://ceur-ws.org

different systems normally only use part of the whole CM, it is necessary to provide CMs with the capability of collecting only necessary entities so that when a system exchanges data with others, only necessary entities are included without redundancy.

In this paper, we propose to introduce two new types as "Category" and "PropertyRelationships" to the meta-model of our previous work [4] . A category can collect only needed entities such as classes, properties, and property relationships from the user's perspective. Category sets can be defined from different viewpoints with aggregation relations among them. Another feature of Category is that entities such as classes, properties, and property relationships can belong to multiple categories.

"PropertyRelationship" is for describing relationships among properties. Similarity between properties, transformation rules between properties etc. can be specified using this concept. Mechanisms for automatic generation of property relationships are also proposed. This work adopts natural language processing approaches combined with the CM structure.

A developed tool called the Framework for Sustainable Conceptual Modeling (FSCM) implements the above-proposed meta-model and mechanisms. Through a case study of creating a large-scale CM and applying it via FSCM to industrial domains, the proposed meta-model and mechanisms are approved to be effective.

The remainder of this paper is structured as follows. Section 2 introduces motivations for this research. Section 3 explains the proposed meta-model and its advantages. Section 4 elaborates on mechanisms for supporting the automatic generation of property relationships. Section 5 describes the FSCM tool. Section 6 introduces and discusses the case study. Section 7 shows related work, and Section 8 gives our conclusions.

2 Motivation

According to our previous work [4], the CIM for smart grid can be described with a set of structured tables (spreadsheets). The CIM described with a set of structured tables is called Parcellized-CIM. Via that work, CIM expressed with UML models such as packages or class diagrams can be represented as Parcellized-CIM in a tabular format. Furthermore, based on the CM of Parcellized-CIM, a commercial database platform parcimoser™ was developed for applications such as Transmission and Distribution Supervisory Control And Data Acquisition (SCADA), Energy Management System (EMS).

However, when applying such CMs to industrial systems, several problems arise. In this paper, two of the typical problems are discussed. To clarify the idea in this paper, Fig. 1 shows a simplified CM, and Fig. 2 shows its corresponding manufacturing process as an explanatory example. As Fig. 1 shows, the class of "Cycle" is composed of "Wheel" and "Frame". Each class has a set of properties, and each property is defined with a set of attributes for example {ID, Name.en, Unit, Version}.

In this example, the "Cycle" class has a set of properties with IDs P6, P7, P10 and property P6 is defined with a set of attributes as {P6, weight, kg, 1}. The "Wheel" class has a set of properties with IDs P1, P2, and P8. The class "Frame" has a set of

properties with IDs P3, P4, P5, P6, and P7. Fig. 2 illustrates the three processes to produce a cycle corresponding to Fig. 1.

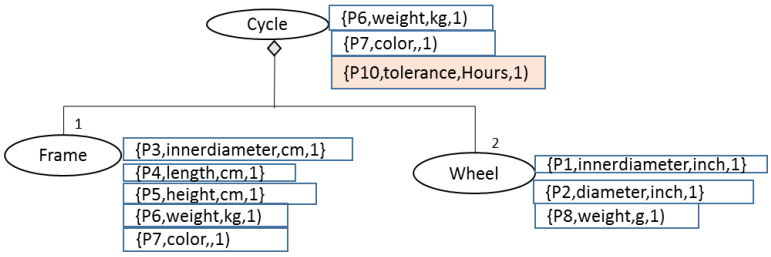

Fig. 1. A partial conceptual model for cycle manufacturing.

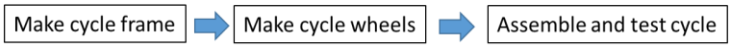

Fig. 2. A simplified product process for cycle manufacturing.

Two of the typical problems to be solved in this paper are as the following:

- (Q1) Property relationships need to be created to improve data interoperability such as consistency checking and exchangeability. In the example of Fig. 1, property P3 (inner diameter) of class "Frame" and property of P2 (diameter) of class "Wheel" should have the same value. Thus, if a property relationship between P3 and P2 existed, it could be used to support their data exchange and data consistency check systematically. In our previous work, property relationships cannot be defined. Furthermore, while there are thousands of properties in industrial domains, it is difficult for CM creators to manually create all property relationships.
- (Q2) Categories containing only necessary entities need to be created to filter out unnecessary information. In the example of Fig. 2, it is clear that for "Make cycle frame" systems, entities such as class "Wheel" and its properties defined in the category "Make cycle wheels" and those in "Assembly and test cycle" are not needed. For a large-scale CM with many entities, the necessity for a system to collect only required partial entities becomes more urgent.

3 Proposed Meta-Model

3.1 Meta-Model Overview

In this paper, an improved meta-model for conceptual modeling is proposed based on our previous work. Primitive elements and their relationships are shown in Fig. 3. In Fig. 3, each entity of "TypedElement" is described by a set of attributes. For "TypedElement" of "Class", "Property", and "Datatype", their attributes set is conforming to the specifications of IEC 61360-1 [5]/ISO 13584-42 [6], which is recoginized as the common conceptual modeling methodology for ISO and IEC standard domain models. Typical attributes for "TypedElement" such as "ID",

"Name", and "Definition" are specified in these standards. All available attributes for each child of "TypedElement" are described in the standard documents [5], [6]. Some known typical characteristics of this conceptual modeling methodology [7] are as follows:

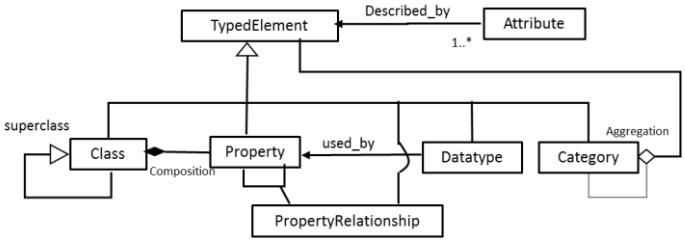

Fig. 3. Meta-model proposed (excerpted).

- Each entity such as a class, a property, or a datatype should have an identifier and a version number;
- Multi-lingual definitions for name, definition, etc. in a CM are available.
- Entity-based CM version management is available.

In this research, "PropertyRelationship" and "Category" as children of "TypedElement" are proposed and extensively defined. In general, "PropertyRelationship" is to describe relationships among properties such as the "equal" relationship among properties. For each property, numbers of relationships can be defined if necessary. "Category" is defined as an aggregation of necessary entities available in a CM. A category can contain several sub categories. The proposed meta-model has the following advantages in brief:

- Property relationships can be assigned with different types or levels according to CM requirements. With created property relationships, data (value) consistency of involved properties can be checked, and their data exchanges become available, so that data interoperability among different systems utilizing involved properties can be improved.
- Not only classes, but also other entities such as a property, a data type, a created property relationship can be grouped into categories from different users' perspectives. As a result, redundant definitions which are not necessary for a "Category" can be filtered out.

To express the proposed meta-model, Table 1, Table 2, and Table 3 representing an excerpted CM complying with the proposed meta-model are utilized. Entities are the same as those in Fig. 1. except for the property relationships in Table 2 and categories in Table 3.

3.2 PropertyRelationship and Its Advantages

Overview. The "PropertyRelationship" is specified to express the relationships among properties. For conceptual models in industrial domains, it is normal for properties to gradually evolve or become connected to other properties with different envi-

ronmental conditions or operational procedures. Principles such as types, levels for "PropertyRelationship" can be defined depending on the requirements of a CM. For the example represented in Table 2, types of "Constraint" and "Reference" relationships are defined. The former is for mandatory relationships among all involved properties. The latter is an optional relationship, meaning that a property involved in this "PropertyRelationship" can make a reference to the other while necessary. In another CM case, property relationships such as "Temporary" and "Permanent" can be defined [3]. Using the CM in Fig. 1, three entities "PropertyRelationship" listed in Table 2 can be defined;

Table 1. Property definitions (excerpted)

ID	Name.en	DefinitionClass	Datatype	Definition	Unit	Version
P1	Inner diameter	Wheel (or class ID)	Real	Inner diameter for a wheel	Inch	1
P2	Diameter	Wheel (or class ID)	Real	Diameter for a wheel	Inch	1
P3	Inner diameter	Frame (or class ID)	Real	Inner diameter for a frame	cm	1
P4	Length	Frame (or Class ID)	Real	Length of a frame	cm	1

Table 2. Property relationships (excerpted)

ID	Name.en	Definition	Resource	Target	Relation	RelationType	Version
R1	diameterRelation	sample	Frame.P3	Wheel.P2	Wheel.P2=Frame.P3*2.54	Constraint	1
R2	weightRelation	sample	Wheel.P8, Frame.P6	Cycle.P6	Cycle.P6=Wheel.P8*2 /1000 + Frame.P6	Constraint	1
R3	colorRelation	sample	Frame.P7	Cycle.P7	Cycle.P7=Frame.P7	Reference	2

- As relationship R1, the property P2 (diameter) of class "Wheel" and P3 (inner diameter) of class "Frame" should have the same value ("="). However, as P2 and P3 have different units as cm (centimeter) and inch, R1 should be defined as "Wheel.P2=Frame.P3*2.54" to support the unit conversion between these two properties in this constraint.
- As relationship R2, the weight of a cycle (Cycle.P6) should be calculated as "Cycle.P6=Wheel.P8*2/1000+Frame.P6", i.e., the sum of wheels' weight and that of the frame.
- As relationship R3, the color of a cycle (Cycle.P7) should refer to the color of the frame (Frame.P7). This relationship is defined as "Reference" type, meaning that Cycle.P7 does not need to be exactly the same color as "Frame", but can make a reference to Frame.P7 while necessary.

With the above described approach, property relationships can be defined with different principles for different CMs.

Advantages. With the "PropertyRelationship" provided by the proposed meta-model, it is possible to describe the relationships among properties. These created property relationships can be utilized to enhance data interoperability such as data consistency checking and data exchangeability, thereby addressing the capability problem for describing property relationships raised as Q1 in Section 2. The remainder of the problem in Q1: automatic generation of property relationships, will be addressed in Section 4.

3.3 Category

Overview. A "category" is specified as a collection of necessary "TypedElements", and can be defined from various viewpoints. For example, from a usage viewpoint, "TypedElements" can be grouped into a category such as "SCADA" or "Demand and Response" in power grid systems. From a viewpoint of a product or system lifecycle, categories such as "General Design", "Detailed Design", and "Validation" can be defined. Depending on the purpose of a defined conceptual model, necessary sets of categories can be defined. With the meta-model defined in Fig. 3, an entity of "Category" can be an aggregation of several subcategories, so that all entities included in sub categories are also included in their parent category and ancestor categories.

This concept of "Category" is evolved from the concept of "package" in UML with the following additional features:

- An entity can belong to multiple categories.
- Not only classes, but also all available "TypedElement" such as properties and property relationships can specify their own categories respectively.
- Each CM can have several sets of categories defined from different perspectives.

Table 3 gives sample category definitions for the CM in Fig. 1 and its manufacturing process shown in Fig. 2. Example categories are defined corresponding to the manufacturing process of Fig. 2. Specifically, category Ca1 is an aggregation of entities for the process of "Make cycle frame", Ca2 is for "Make wheel", and Ca3 is for "Assemble and test cycle". Since Ca3 contains Ca1 and Ca2, all entities included in Ca1 and Ca2 are also contained in Ca3.

Table 3. Category definitions (excerpted)

ID	Name.en	Super Category	ElementList	Definition	Version
Ca1	MakeCycle Frame	Ca3	{(Frame),Class}, {(P3,P4,P5,P6,P7),Property}	All entities available for functions, operations, systems etc. for making cycle frame	1
Ca2	MakeWheel	Ca3	{(Wheel),Class}, {(P1,P2,P3,P8), Property} {(R1), PropertyRelation}	All entities available for functions, operations, systems etc. for making cycle wheel	2
Ca3	Assemble AndTest Cycle	Root	{(Cycle),Class} {{P10},Property} {(R2,R3), PropertyRelation}	All entities available for functions, operations, systems etc. for assembling and testing cycle	1

- In the "Make cycle frame" procedure and its relevant systems, only entities defined in category Ca1 are necessary, meaning that for systems related to the procedure, only entities listed in the "ElementList" column of Ca1 in Table 3 are necessary.
- In the "Make cycle wheels" procedure, only entities defined in the "ElementList" column of Ca2 in Table 3 are required. Because the property relationship R1 is included in Ca2, systems related to the procedure need to check whether the wheel diameter equals the inner diameter of a cycle.
- For the "Assemble and test cycle" procedure, Ca3 which includes all entities defined in Ca1 and Ca2 can be utilized. It can also have its specified entities listed in

the "ElementList" column of Ca2 in Table 3. In consequence, systems relevant to the procedure must check whether each of R1, R2, and R3 are satisfied. For example, when executing R2, systems need to check whether the weight value of the cycle satisfies the relationship R2.

Advantages. The above descriptions show that category definitions containing only necessary entities can solve problem Q2 raised in Section 2. Because categories can be defined by CM users from different aspects and for various purposes, we do not discuss category generation in this paper. Further, due to space limitations, this paper does not introduce the approaches that offer supports to collect necessary entities for a given category. Some discussion of category creation and their advantages are discussed in the case study.

4 Generation of Property Relationship

As described in Section 3.2, some of the property relationships can be defined when designing the CM. Such property relationships are usually derived from the knowledge and experience of CM creators. In industrial domains, because a large amount of entities are available in a CM, it's necessary to provide approaches supporting automatic generation of property relationships. One approach in this work is to calculate similar properties in a CM and then recommend similar properties to build property relationships automatically. Model creators thus can build and add exact property relationships in existing CMs. For this purpose, natural language processing approaches combined with the CM class distances are proposed.

4.1 Similarity Calculation among Properties

One approach is to use semantic similarities among properties [8] combined with structure information. As illustrated in Section 3.1, each property is described with a set of attributes such as "Name", "Definition", "Datatype", or "Unit", and similarity between a selected property entity and other properties is calculated as

$$S(P1, P2) = \sum_{a \in A} W_a \cdot Sa(P1.a, P2.a) \,/\, \sum_{a \in A} W_a \quad (1)$$

$$Sa(P1.a, P2.a) = (\overrightarrow{v1} \cdot \overrightarrow{v2}) \,/\, (|\overrightarrow{v1}| \cdot |\overrightarrow{v2}|) \quad (2)$$

$$\overrightarrow{vi} = (Vi1, Vi2, \ldots Vin) \qquad Vij = (\text{TF-IDF-Score}(Termj)) \quad (i=1,2) \quad (3)$$

where Termj∈(Term1, Term2,…,Termn), an ordered set of terms and values of P1.a and P2.a.

In equation (1)(2)(3), S(P1, P2) is a weighted average similarity score between property P1 and P2, a is an attribute utilized by a property, P1.a and P2.a are respectively the value of a in P1 and P2, and W_a is the weight coefficient for attribute a, set from the aspect of similarity calculation. The larger Wa is, the more critical the attribute is to a property.

When calculating similarity score Sa (P1.a, P2.a) of attribute a between P1 and P2, we adopt natural language processing approaches such as WordNet [9] for word simi-

larity, TF-IDF Cosine similarity [10] shown in Equation (2)(3) for sentence similarities. The $\overrightarrow{v1}$ and $\overrightarrow{v2}$ are vectors of the calculated TF-IDF scores for each attribute a.

Taking the properties listed in Table 1 as an example. Suppose we generate relationships of property P1, for each other properties in the same CM, the following procedure is adopted to calculate the similarities between P1 and other properties. The idea is clarified with Table 4 which lists calculated similarity scores for property P2, P3, and P4 to P1.

Table 4. Similarities between properties

Attributes	ID	Name.en	Definition Class	Datatype	Definition	Unit	Version	Similarity (no weight)	Similarity (with weight)
Weight	1	10	1	10	10	10	1	-	-
P2 to P1	0	0.7	1	1	0.89	1	1	0.79	0.88
P3 to P1	0	1	0.5	1	0.8	0.82	1	0.74	0.98
P4 to P1	0	0	0.5	1	0.2	0.82	1	0.5	0.504

- In the first step, a similarity score between every attribute is calculated with its value (content); For example, regarding the similarity score of the attribute Name.en, P1 defines Name.en as “Inner diameter”, Name.en of P2 is “Diameter”, and the similarity score between the “Inner diameter” and “Diameter” is calculated as 0.7, then this score is listed in the Name.en columm of line “P2 to P1”.
- In the second step, a weight coefficient is assigned to each attribute. An important attribute should be assigned with a lager weight coefficient. For example, the attributes “Name,” “Definition,” “Datatype,” and “Unit”, which to a large degree determine property instances, should be assigned larger weight coefficients than others.
- Finally, the similarity score between a property P*x* and a selected property P1 is calculated with Equation (1), using the weighted average similarity scores for P1 and Px attributes. In the example of Table 4, in the column of “Calculated Similarity (no weight)”, similarity scores with no specific weight coefficient, i.e., 1, are recorded, and in the column of “Calculated Similarity (with weight)”, the similarity scores with the set weight coefficient in line 2 are listed.

The similarity score results show that different weight coefficients affect similarity score rankings. In the “Similarity (no weight)” column property P2 has the highest score, but in the “Similarity (with weight)” column property P3 has the highest score. This difference is due to weighting of the “DefinitionClass” attribute. Because we want to focus on property relationships among different classes, properties within the same class are set with lower weight coefficients. According to this principle, we adopt the result in the Table 4 column “Similarity (with Weight),” and P3 which has the highest similarity score to P1 is recommended to systematically build a property relationship with P1. The relationship R1 displayed in Table 4 is the corresponding definition.

4.2 Class Distance for Property Relationship Recommendations

Structure features are also considered when ranking property relationships. When deciding the recommendation rank of generated property relationships, besides the

calculated similarity score of P*x* and P*y*, the distance among definition classes of P*x* and P*y* is also utilized. Because in this proposed CM, classes have is-a (specialization) hierarchical relationships, a property can be inherited from ancestor classes by a child class. Therefore, if two properties with high similarity scores simultaneously have an ancestor-descendant relationship or neighborhood relationship, in principle, these two properties are highly recommended for creating a property relationship.

In order to calculate the distance among classes, a known Lowest Common Ancestor (LCA) [11] are utilized in this work. In LCA, the distance between classes is calculated as

$$\text{Distance}(cls1, cls2) = \text{Dist}(root, cls1) + \text{Dist}(root, cls2) - 2 \cdot \text{Dist}(root, lca) \quad (4)$$

where Dist(root, cls1) is the distance from cls1 to root class in the is-a hierarchical tree, and "lca" is the lowest common ancestor of cls1 and cls2.

Through these approaches with CM semantic definitions and structure features, similar properties can be collected so that the relationship among properties can be ranked and recommended systematically.

5 Support Tool

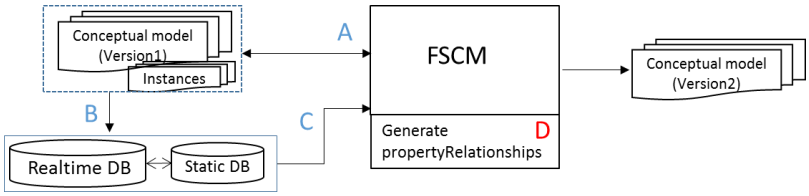

Fig. 4. FSCM supporting sustainability of CM.

A conceptual modeling tool called FSCM shown in Fig. 4 is developed based on a developed tool parcimoser™, which was introduced in our previous work [4]. In that work, an Excel-based FSCM supports functions for components A, B, and C. This is explained briefly as follows:

- Component A provides functions such as CM design and generation, instance template generation, and static data input.
- Component B provides functions such as database schema design, table creation, CM and static instance data storage to a database.
- Component C provides functions such as exporting and synchronizing CM and instance data in a database to FSCM.

Continuing this work, we newly developed component D for this FSCM to automatically generate property relationships utilizing the first version of CM and its instance data. Category creation and collection of necessary entities are newly implemented as an extension of component A. "Conceptual Model Version1" and "Conceptual Model Version2" are the different versions of the same CM. In this paper, we focus on only the differences in property relationships between CM Version1 and CM Version2, Namely,

CM(Version1) ∪{generated property relationships} = CM(Version2).
Other entity changes from CM Version1 to Version2 are not discussed in this paper.

Fig. 5 shows a sample CM created by the FSCM. In Area I, a class is-a hierarchical tree is displayed. In Area II, detailed information of a selected class, including its category, is listed. In Area III, properties defined for that class and those inherited from the ancestor classes can be listed. For each property, attributes such as "Name", "Datatype(valueType)", "Category", can be represented.

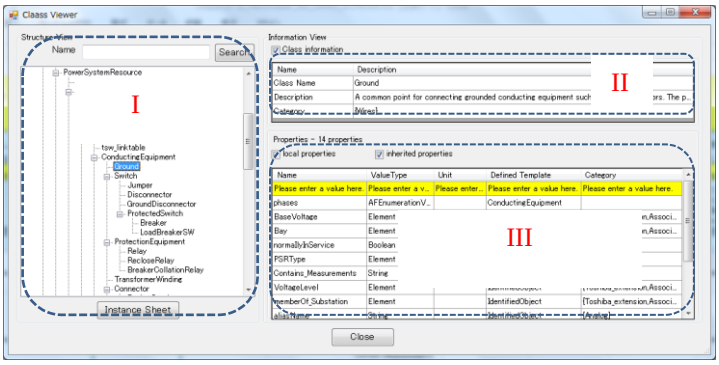

Fig. 5. Example of conceptual model in FSCM.

6 Case Study

An FSCM based on the proposed meta-model was applied to a commercial system, and the provided CM was shown to be efficient and successful in an actual industrial application. In this work, some of the standardized industrial conceptual models[1,2] were adopted and defined using the FSCM tool with internal extensions. In this use case, the CM shown in Table 5 was utilized to explain the contributions of the newly proposed "Category" and "PropertyRelationship". Table 5 lists only necessary entity types; those not pertinent to this paper are omitted.

Table 5. Conceptual model in our case study

Entity type	CM1 Entity number
Class	1312
Property	5629
Datatype	55
Category (created with this work)	24

Category definition and discussion. Fig. 6 shows the example of 24 categories defined in the CM from one viewpoint. As already explained in Section 3.3, categories can be defined from different perspectives according to users' requirements and CM utilization in individual user systems. The 24 categories in Fig. 6 are just one of the potential category sets. In Fig. 6, entities of classes and properties included in each

[1] http://std.iec.ch/cdd/iec61360/iec61360.nsf/TreeFrameset?OpenFrameSet
[2] http://collaboration.iec.ch/other_sc3dworkingmaterial/IEC62656-Part3/

category are presented. Some categories, such as Categories 1, 2, 4, and 5, clearly have only limited entities, while categories such as 17 and 19 have large number of entities. Obviously, no category contains all entities available in the CM. It's explicit that especially for small size categories, the category concept greatly contributes to reducing the unnecessary information.

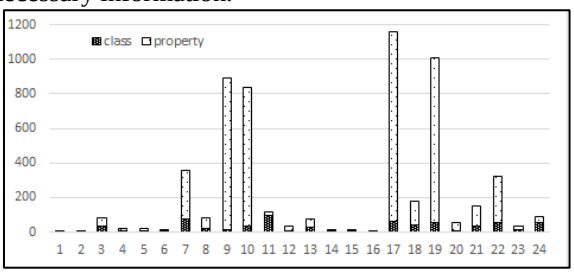

Fig. 6. Categories and their containing entities.

Property relationships generation. Component D of the FSCM shown in Fig. 4 was used to generate property relationships with the mechanisms explained in Section 4.1. In this case study, the weight coefficient W_a of the "Name", "Definition", "Datatype" and "Unit" attributes were set to 1, and those for other attributes were set to 0 as discussed in Section 4.1.

Fig. 7 shows the results. We totally obtained 15,840,006 property relationships throughout the total CM, 4,508,160 of which were between properties in different categories. Notably, 1,144 property relationships received similarity scores of 1. In Fig. 7, numbers of generated property relationships with similarity scores ranging from 0 to 1 are illustrated. These results were adopted for the evaluations.

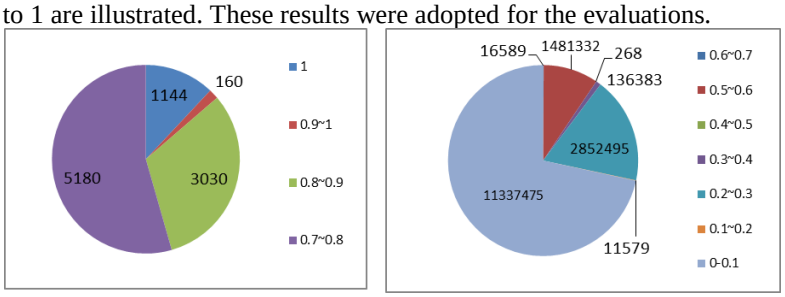

Fig. 7. Automatic generated property relationships at different similarity scores.

Evaluation of automatically generated property relationships. From automatically generated property relationships with similarity scores 1, 0.9~1, 0.8~0.9, 0.7~0.8, 0.6~0.5 and "0.5 or less", we randomly selected 20 relationships from each similarity score range as samples to evaluate the proposed approach. Totally 120 automatically generated relationships were evaluated and Fig. 8 shows the case study results.

In Fig. 8, the horizontal x-axis represents the similarity score (S) of generated property relationships from 1, and 0.9~1 down to "0.5 or less". Here, 1 means S=1, 0.9 means 0.9=<S< 1, 0.8 is 0.8=<S< 0.9, and the others have the similar meanings. The y-axis shows the results as evaluated by one of the authors. For the randomly

selected property relationships, one of the authors looked through the generated property relationships and judged whether generated property relationships were correct. Numbers on the y-axis represent precision of property relationship generation, using the number of accepted property relationships divided by the number of automatically generated property relationships. Fig. 8 shows four types of curved lines. The "3" line shows the results for property relationships with class distances no larger than 3, the "4" line is for those with class distances between 3 and 4. Other lines have the similar meanings. Class distances were calculated using Equation (4).

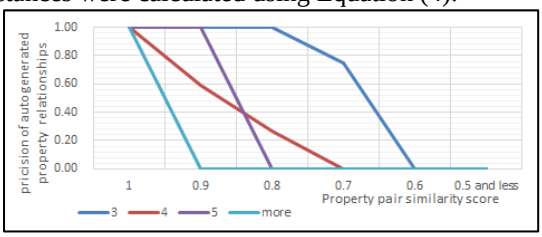

Fig. 8. Evaluation of automatic generated property relationships.

Fig. 8 shows the following:

- For generated property relationships with similarity score 1 (x=1), despite the class distances, all generated property relationships were judged to be acceptable, namely that the precision was 100%.
- For generated property relationships with similarity scores 0.9 to 1 (x=0.9), the precision of automatically generated properties was 71% when class distance were not considered, and the precision improved to 89% when class distances were restricted to less than 5.
- For generated property relationships with similarity scores 0.8 to 0.9 ($x = 0.8$), the overall precision was 45.3% when class distances were not considered. When class distances were restricted to less than 4, the precision improved to 75.5%.
- For generated property relationships with similarity scores 0.7 to 0.8 ($x = 0.7$), the overall precision was about 30%, and the precision improved to 50% when class distances were restricted to less than 3.
- For generated property relationships with similarity scores 0.6 to 0.7 ($x = 0.6$) and others less than 0.6, the precision was 0 regardless of class distances.

From these results, we arrive at the following conclusions:

- Class distance effectively improves the precision of automatic generated property relationships for similarity scores ranging from 0.7 to 1. For those with similarity scores of 1 or less than 0.6, class distance has no influence.
- Class distances less than 5 significantly improve the precision of automatically generated property relationships. When class distance is larger than 5, even the automatic generated property relationships have similarity score 0.9, they were judged to be not correct relationships. Thus, the effective class distance threshold for this CM1 should thus be set to 5.
- Automatically generated property relationships with similarity scores exceeding 0.8 are 75.5% acceptable with class distance no larger than 4. Thus, property rela-

tionship similarity scores combined with class distances need be considered simultaneously when FSCM deciding the precision threshold for the automatically generated property relationships.

This case study shows that property relations can be automatically and precisely generated with the proposed meta-model of "Category" and "PropertyRelationship." The CM with categories was already applied to actual industrial systems via FSCM to support data interoperability such as data exchange and interpretation. Automatically generated property relationships can be sequentially adopted for further improvement.

7 Related Work

In [12], with the goal of determining semantic similarities among entity classes from different ontologies, natural language processing approaches were utilized to create a similarity function for calculations using synonym sets, semantic neighborhoods, and other aspects of ontologies such as WordNet and the Spatial Data Transfer Standard (SDTS). In that work, similarity among entity classes was calculated with attributes grouped by function, description, and so forth. Distances among entity classes were not considered. Reference [13] proposed levels of the conceptual interoperability model. Depending on the interchanged data, up to five levels (level 0 for system specific data, level 1 for documented data, etc.) of the conceptual interoperability model were proposed. In [14], a conceptual framework utilizing deep instantiation (DI) for conceptual models was introduced to improve the ecosystem interoperability in the oil and gas industry. Concepts such as potency (representing DI for each entity) were proposed, but this work did not describe systematic generation with respect to these concepts.

In this work, the proposed category and property relationships are adaptable, regardless of the data interchange level. Furthermore, the automatic generation and recommendation mechanisms of property relationships should improve data interoperability among heterogeneous systems.

8 Conclusion

We introduced an improved approach for large-scale conceptual modeling. We first proposed a meta-model that newly specifies the concept of "Category" and "PropertyRelationship." To support automatic generation of property relationships, we introduced mechanisms based on weighted similarity calculations among properties, and class distances of these properties, and described a tool called as FSCM that implemented the proposed ideas. Through a case study of a large-scale CM, the proposed meta-model for conceptual modeling and the FSCM were proved to be able to automatically and precisely generate property relationships. The CM and FSCM were applied to actual industrial systems and approved to be effective for solving their data interoperability problem. In future work, we will further evaluate this and other CMs to promote FSCM to various industrial domains to improve their data interoperability.

References

[1] IEC 62664-1, "Enterprise-control system integration – Part 1: Models and terminology," 2013.

[2] IEC 62664-2, "Enterprise-control system integration – Part 2: Objects and attributes for enterprise-control system integration," 2013.

[3] IEC 62664-4, "Enterprise-control system integration– Part 4: Objects and attributes for manufacturing operations," 2015.

[4] L. Wang and H. Murayama, "A Multi-version CIM-Based Database Platform for Smart Grid," IEEJ Transaction on Electrical and Electronic Engineering, vol.10 No.3,pp.330-339, 2015.

[5] IEC 61360-1, "Standard data elements types with associated classification scheme for electric items - Part 1: Definitions - Principles and methods," 2009.

[6] ISO 13584-42, "Industrial automation systems and integration -- Parts library -- Part 42: Description methodology: Methodology for structuring parts families," 2010.

[7] L. Wang, A. Hosokawa and H. Murayama, "An Evolutive and Multilingual CIM Ontology Management System," Energy Procedia, vol 12, pp.18-26, 2011.

[8] L. Wang, "Data Management Equipment, program and method.". Patent P2013107262.

[9] Ted Pedersen, "WordNet::Similarity - Measuring the Relatedness of Concepts," Demonstration Papers at HLT-NAACL,pp.38-41, 2004.

[10] A. Palakorn, H. Xiaohua and X. Shen, "The Evaluation of Sentence Similarity Measures," DaWaK, LNCS vol 5182, pp.305-316, 2008.

[11] M. A. Bender and M. Farach-Colton, "The LCA problem revisited," Proceedings of the 4th LATIN, LNCS vol. 1776, pp. 88–94, 2000.

[12] M. Rodriguez and M. Egenhofer , "Determining Semantic Similarity among Entity Classes from Different Ontologies," IEEE Transactions on Knowledge and Data Engineering, vol. 15, no. 2, pp.442-456, 2003.

[13] J. A. Andreas Tolk, "The Levels of Conceptual Interoperability Model," Fall Simulation Interoperability Workshop, 2003.

[14] M. Selway , M. Stumptner , W. Mayer, A. Jordan, G. Grossmann and M. Schrefl, "A Conceptual Framework for Large-scale Ecosystem Interoperability," Proc. ER. LNCS vol. 9381, pp.287-301, 2015.