Concept-Based Method for Extracting Valid Subsets from an EXPRESS Schema

Ghang Lee, Ph.D.1

Abstract: An EXPRESS schema is a data schema defined in EXPRESS, an international standard language for defining product data schemas. This technical paper proposes and formally defines a set of conditions for generating a minimum valid subset of an EXPRESS schema corresponding to a concept, where a concept is a general idea and a subset is a partial model of a data schema. We introduce a notion of “minimal set” to define the relationships between a subset and other subsets, and also between a subset and concepts. A minimal set is the smallest complete subset of a schema that corresponds to a concept. Using IFC, an international standard data model for the architecture, engineering, and construction industry, the proposed conditions have been implemented in a software application developed for extracting subsets from the IFC schema matching the concepts. A number of examples are demonstrated.


CE Database subject headings: Computation; Computer software; Data processing; Standardization.

Introduction

Several standard data models for the architecture, engineering, and construction (AEC) industry have become available for commercial use today as a result of decades of effort by standardization organizations, including International Alliance for Interoperability (IAI), American Institute of Steel Construction (AISC), and the ISO STEP Committee. Examples include CIS/2 (Crowley 2003), a standard data model for the steel construction industry, and IFC (IAI 2006), a standard data model for supporting data exchange and management through the entire lifecycle of buildings.

A number of software applications are certified for the support of these product models, but only a few can exchange data between each other in a standard data format. Translators that are being developed today are based on the intention of each software developer and, as a result, several problems arise in data exchange, even between software applications in the same domain (Fig. 1).

This is because, when data are exchanged, only a subset of data defined in a standard data model is exchanged, but there are no rigorous development guidelines on how and which subsets of a standard data model should be used or when. There have been efforts to resolve this subset issue by predefining subsets of a standard data model for specific use cases and then providing specifications for using them. These are represented by the conformance classes specification efforts by ISO STEP (ISO 2001) and CIS/2 (Crowley et al. 1999), and the Information Delivery Manual (IDM) effort (Wix 2005b) and the model view definition (MVD) effort by IAI (Hietanen 2006).

However, subset specification is a slow and error-prone procedure, which begins by defining specific information use cases in terms of process models. Then, the subset modeler selects entities and properties required by the specified use cases from a full data model. In this process, if any one of the supertypes, entities, properties, rules, functions, or other elements that are associated with the selected entities are not properly included in a resultant subset definition, the subset will be invalid. Eventually the errors can be fixed, but not easily. The same error may reoccur even after several cycles of careful review and correction processes, assisted by an automated error-checking tool.

This paper proposes a set of conditions for automatically extracting a valid subset from an EXPRESS schema corresponding to a specific concept. EXPRESS is an international standard language for defining product, or product data, models (ISO 1994b). An EXPRESS schema is a product data model defined in EXPRESS, for example, IFC and CIS/2. In EXPRESS, a schema is composed of unique ENTITIES and/or TYPES (Wilson 1996, p. 11).

The term “concept” in this paper is compatible with “concept” in MVD (Hietanen 2006), “aspect” in ProIT (VTT 2002), “functional part” in IDM (Wix 2005a), and “unit of functionality” in ISG (2000). A concept, as defined here, is a general idea independent of any schema, for example, a “wall” or “building.” Specific IFC entities that correspond to the concepts “wall” and “building” would be “IfcWall” and “IfcBuilding.” Subsets that correspond to the concepts “wall” and “building” would include not only “IfcWall” and “IfcBuilding,” but also other various entities, attributes, functions, and rules required to define “wall” and “building.”

A goal of this study is to provide an efficient method, as well as a tool, to support activities such as specification of information-exchange requirements. The theoretical foundation of this work extends from the definitions of subsets by ISO (2001), IAI (Hietanen 2006), Wilson (Wilson 1996), and AISC (Crowley et al. 1999), and the subset generation rules proposed by Yang and Eastman (2007).

The rest of the paper is organized as follows. “Subset Generation Efforts for Express Schemas” briefly reviews the history of
subset problems, various terms that apply to subsets, and previous efforts to develop a systematic approach for specifying subsets. “Conditions for Extracting Valid Subsets from Express Schema” proposes and describes a set of conditions that allow automatic extraction of a valid subset corresponding to a concept from an EXPRESS schema. A formally defined notion of “minimal set” and its relationship to “concepts” is also introduced in this section. “Implementation” illustrates an example of implementation of the proposed conditions and discusses some of the limitations.

Subset Generation Efforts for Express Schemas

Brief History of Subset Problems

EXPRESS is an international standard language for defining product data representations (ISO 1994b). It is referred to as an object-flavored language, rather than an object-oriented language, as it does not support several of the major features common to object-oriented languages, such as encapsulation and polymorphism. However, because of its ability to support inheritance and other object-oriented features, EXPRESS is effective in defining product data models that often have complex inheritance and aggregation structures.

EXPRESS is also part of STEP (officially known as ISO 10303), an international standard for product data representation and exchange (ISO 1994a). In STEP, application protocols (APs) define “the scope, context, and information requirements of an application and the representation of the application information” (ISO 1994a, p. 8). At the present time, 33 APs are defined in STEP, including AP 225 for building design. The term “application protocols” originated from a proposal by Liewald and Kennicott (1982) for AP development (Bloor and Owen 1995, p. 121): i.e., “subsets of a large neutral data model such as initial graphics exchange specification (IGES) and graphic kernal system (GKS) to support data exchange between computer aided design (CAD) systems” in 1982 (Liewald and Kennicott 1982). The proposal eventually led to efforts, begun in 1984, to develop STEP as a successor to various early standardization efforts for product data exchange, such as IGES, SET, GKS, MIL-D specifications, and VDA-IS (Bloor and Owen 1995). However, once several APs had been developed and were made available for commercial use, it was soon realized that even smaller subset definitions were required to support data exchange between applications. The STEP community called these smaller subset definitions “conformance classes (CCs)” (ISO 2001).

IAI and AISC, which develop and maintain IFC and CIS/2, respectively, also came to recognize a similar problem and the necessity for subset definitions, when translators for IFC and CIS/2 became available in the market. IAI and AISC called these subsets MVDs (Hietanen 2006) and CCs (Crowley et al. 1999), respectively.

Terms for Subset

The following definitions of the terms for subset vary slightly, depending on the organization:

1. CC in STEP: “a subset of an application protocol for which conformance may be claimed” (ISO 1994a, p. 3);
2. CC in CIS/2: “a testable subset of the implementation schema” (Crowley et al. 1999, p. 1); and
3. MVD in IFC: “a documentation on how the IFC model specification (i.e., IFC schema and its documentation) is applied to data exchange between different application types” (Hietanen 2006, p. 4).

Based on these definitions, a MVD is not a subset, but rather is a document about a subset and its usage. Technically speaking, CCs as defined by ISO STEP and CIS/2 also require descriptions on application contexts and, thus, are not only subsets. However, when these terms are used in the general context, the terms “view,” “view definition,” and “conformance classes” are used to simply denote a subset of a schema, both by IAI and the STEP community, as these terms are commonly used by the other information systems communities.

In this paper, we use the term “subset” as a general term for denoting a partial model of a product model. We use the terms “views,” “view definitions,” and “conformance classes” to denote “valid subsets” of a product model and use them interchangeably, depending on context, without distinguishing them from the broader definitions, including the application contexts and requirements. A subset may or may not be valid, but given any valid set, its subset is valid if the subset is syntactically correct (i.e., well formed) and structurally conforms to its superset.
Subsetting Efforts

Efforts for defining subsets of an EXPRESS schema including STEP schemas, CIS/2, and IFC are still ongoing and, as a matter of fact, have been accelerated, as there are increasing demands from the industry for definition of more subset models for specific information use cases. Several efforts have been made to make the subset specification and validation process systematic and rigorous; the STEP specifications for conformance classes and testing are one example (ISO 2001). IAI’s efforts for developing information delivery manuals (IDMs) (Wix 2005a) and MVDs are another. However, these efforts serve to make the subset specification and validation process more rigorous and standardized and have little to do with automated subset generation.

There have been efforts to generalize and formalize subset query processes, both at schema and instance levels. The generalized model subset definition (GMSD) proposed by Weise et al. (2003) which deals with a subset query method at an instance level, was developed as a generalized method for applying predefined “view definitions” in querying data instances in a fast and efficient way. Based on a selected query and predefined views, it first eliminates irrelevant data objects in a schema and then searches only through the remaining data objects. It provides implementation-level details on how to minimize the long transaction time problem in engineering database systems that is caused by importing and exporting large data sets (Scherer et al. 2006). However, automated generation of schema-level subsets is not within its scope.

A notable effort for development of a subset generation method for a schema specified in EXPRESS is the “rule-based subset generation method” proposed by Yang and Eastman (2007). This method first categorizes subsets into four groups, using a two-by-two matrix of schema/instance verses syntactic/semantic subsets, as illustrated in Table 1. The term subset (including conformance classes and view definitions) as discussed in this paper, falls into the syntactic subset quadrant in Yang and Eastman’s classification.

Yang and Eastman define a syntactic subset as an aggregation of what they call “base sets,” which is a “set of data types relating to only one base entity” (Yang and Eastman 2007, p. 138). The concept of base entity in Yang and Eastman came from the base entity in CIS/2: i.e., an entity of interest that acts as a “building block” to build larger conformance classes (Crowley et al. 1999, p. 4).

BR01–BR08 below (Yang and Eastman 2007) define rules for generating valid base sets, where BR stands for basic set rules. Reserved terms in EXPRESS are capitalized; EXPRESS supports the object-oriented approach and, thus, supports the concept of entities, attributes, SUPERTYPES, SUBTYPES, RULEs, and PROCEDUREs. EXPRESS has several data types, including simple (NUMBER, REAL, INTEGER, LOGICAL, BOOLEAN, STRING, and BINARY) data types, aggregation (ARRAY, LIST, BAG, SET) data types, named (ENTITY and defined) data types, and constructed (ENUMERATION and SELECT) data types. Other general descriptions regarding EXPRESS and the EXPRESS syntax are provided where necessary, but the reader is referred to ISO 10303 Part 11 (ISO 1994b) and Schenk and Wilson (1994) for more details on EXPRESS.

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>BR01</td>
<td>There is one entity data type that is a base set. (Note: A base set can have only one entity data type. The entity is called a base entity.)</td>
</tr>
<tr>
<td>BR02</td>
<td>A base set cannot be an ABSTRACT entity data type. (Note: ABSTRACT entity data types are not instantiable entity data types in EXPRESS.)</td>
</tr>
<tr>
<td>BR03</td>
<td>A base set includes attribute definitions of the base entity</td>
</tr>
<tr>
<td>BR04</td>
<td>A base set does not include entity data types referenced by the attributes</td>
</tr>
<tr>
<td>BR05</td>
<td>A base set includes constructed data types of referenced attributes</td>
</tr>
<tr>
<td>BR06</td>
<td>A base set includes defined data types of referenced attributes</td>
</tr>
<tr>
<td>BR07</td>
<td>A base set includes RULE or FUNCTION or PROCEDURE used in the definition of the base entity; and</td>
</tr>
<tr>
<td>BR08</td>
<td>A base set inherits SUPERTYPE(s) attributes, types, and functions. (Note that this rule does not require inclusion of SUPERTYPE entities in a base set. This guarantees to leave only one entity in a base set.)</td>
</tr>
</tbody>
</table>

A base set alone is syntactically incomplete (i.e., invalid) because, in EXPRESS, a data set must include all the referenced entities to be valid and it does not include any referenced entities (See BR04 above). However, a subset, which is an aggregation of base sets, is complete and valid. QR01–QR09 below (Yang and Eastman 2007) define rules for generating syntactically valid subsets in Quadrant 1, where QR stands for quadrant (1 subset) rules. QR03–QR05 defines rules for including referenced entities:

<table>
<thead>
<tr>
<th>Rule</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>QR01</td>
<td>A Quadrant 1 subset is a member of the powerset of a schema. (Note: This definition came from Wilson’s definition of “subtype combination” (Wilson 1996));</td>
</tr>
<tr>
<td>QR02</td>
<td>There is at least one base set within a Quadrant 1 subset</td>
</tr>
<tr>
<td>QR03</td>
<td>A Quadrant 1 subset includes base sets of all of the reference entities</td>
</tr>
<tr>
<td>QR04</td>
<td>Specialization of an entity in a referenced base set makes separate subsets</td>
</tr>
<tr>
<td>QR05</td>
<td>A Quadrant 1 subset does not include referenced ABSTRACT entity data types, but must include one of the base sets of the subtypes of the referenced ABSTRACT entity data types</td>
</tr>
<tr>
<td>QR06</td>
<td>Attribute values of SELECT data types make a different Quadrant 1 subset</td>
</tr>
<tr>
<td>QR07</td>
<td>An entity with INVERSE type attributes is included in a Quadrant 1 subset</td>
</tr>
<tr>
<td>QR08</td>
<td>Required entities in domain rules (WHERE clauses) are elements of a Quadrant 1 subset; and</td>
</tr>
<tr>
<td>QR09</td>
<td>A Quadrant 1 subset only includes one definition of each data type</td>
</tr>
</tbody>
</table>

Since both our rules and Yang and Eastman’s rules are based on the same EXPRESS syntax, there are similarities in the basic concept and the rules. However, there are also significant differences between the two approaches and, thus, the rules. Our proposed approach and its differences are described in the next section.

Conditions for Extracting Valid Subsets from Express Schema

The proposed method is based on the dichotomy of “concepts” (general ideas) versus “corresponding subsets in a specific (data)
schema,” to make concepts not only independent of schema versions but also reusable for other schemas. This follows a general approach taken by other IFC-related efforts, such as MVD (Hietanen 2006), ProtT (VTT 2002), IDM (Wix 2005a), and ISG (2000).

In order to formally define the relationship between concepts and corresponding subsets, a new notion of “minimal set” is introduced here. A minimal set is the smallest nonempty complete subset of an EXPRESS schema corresponding to a concept. A minimal set thus defined is different from the “base set” in Yang and Eastman (2007) in that it can include more than one ENTITY and that each minimal set is complete as a subset by itself. The base set of Yang and Eastman can include only one ENTITY and each base set alone is not complete as a subset of a schema.

Definition. [minimal set] A minimal set $M$ of a schema $S$ with respect to a concept $c$ is the smallest nonempty complete subset of a schema $S$ that corresponds to a concept $c$. A minimal set $M$ of a schema $S$ with respect to a concept $c$ satisfies the following:

$$\exists c \in S, M \subseteq S \land \text{COMPLETE}(M) \land \forall f_m | f_m(c) = M \implies M = \emptyset, 0 < |M| \leq |\text{complete}(c)|$$

where, given a schema $S$, $M$ is a minimal set; $c$ is a concept in a nonempty set $C$ of concepts; COMPLETE($X$) is a function for checking whether a set $X$ is complete for a given schema; and $f_m = \text{complete}(c)$ from $S$ to Pow($S$) such that, for any $c$ in $C$, $f_m(c)$ contains an ENTITY declaration relevant to $c$.

Note: Pow($S$) denotes a powerset of a schema $S$, namely a set of all the subsets of $S$.

As structured data of a domain, a schema $S$ is set theoretically treated as a set consisting of a nonempty list of ENTITY declarations and a (possibly null) list of other TYPE declarations. A subset $S_b$ of a schema $S$ is said to be “complete” if every declaration in $S_b$ is well formed, satisfying all of the conditions to be stated presently, as well as all of the general conditions for a schema. This type of subset is minimal if the cardinality of $S_b$ is greater than 0, but not greater than any complete set $S_b$ in $S$.

The relationship between concepts and minimal sets is defined here as a function in set-theoretic terms. With this definition, each concept corresponds to a unique, namely one and only one, minimal set. It is, however, possible that the mapping from concepts to minimal sets is many to one. In this case, these concepts or, more specifically, concept correspond to a unique, namely one and only one, minimal set. It is, however, possible that the mapping from concepts to minimal sets is many to one. In this case, these concepts or, more specifically, concept $c_1$ and $c_2$ correspond to one minimal set $M_1$ and $M_2$.

Nine special conditions are proposed here to guarantee the validity (i.e., “well formedness”) of a minimal set. Here, the term “condition” is deliberately used throughout this paper to distinguish it from “RULE,” which is a reserved term in EXPRESS.

Condition 1. (Mapping between concepts and minimal sets). By the definition of “minimal set,” there must be one and only one minimal set $M$ corresponding to each element $c$ in a set $C$ of concepts

$$\forall c \in C \to \exists x | f_m(c) = x \land \forall y | f_m(c) = x \land x = y$$

where $c$ = concept; $C$ = set of concepts; and $x$ and $y$ = minimal set.

Note:

1. If more than one concept maps to a minimal set, these concepts are understood as referring to terms and considered as synonyms, belonging to a synonym set; and
2. Homonyms are not admissible in the set $C$ of concepts.

On the basis of a set of semantically related concepts, a new concept can be defined. There are at least two ways of forming new concepts: one is based on the generalization or specialization (kind of) relation and another on the aggregation or decomposition (part of) relation. For example, the concepts “living room,” “bedroom,” and “bathroom” are related to a concept “room;” this relationship is viewed as a generalization of the former to the latter and its inverse is a specialization. On the other hand, a concept “building” is related to other concepts like “room” and “space,” while this relation is the aggregation or decomposition relation.

By treating a concept as a set of properties, these relationships among concepts can be formalized. A general concept “room,” for instance, is understood as the intersection of all of its subconcepts, formally treated as sets, such as “living room,” “bedroom,” and “bathroom.” An aggregated concept “building” is the union of all of its subparts that include “room” and “space,” which are again treated as set-theoretic objects. [See Wilson’s “EXPRESS and Set Theory (1996)” for more detail on the relation between EXPRESS and the set theory.]

**Condition 2.** (Forming new concepts). Let $c_i$ and $c$ denote concepts or sets of properties:

1. $\cap c_i = c$ iff $c$ is a generalized concept such that each $c_i$ is a subtype of $c$;
2. $\cup c_i = c$ iff $c$ is an aggregated concept such that each $c_i$ is a part of $c$; and
3. If $i = 0$, then $c_0 = c$ such that they are synonymous.

By the same token, a new minimal set can be defined as the intersection or union of other minimal sets.

**Condition 3.** (Forming new minimal sets). Let $m_i$ denote a minimal set:

1. $\cap m_i = m$ iff $m$ is a minimal set corresponding to a new generalized concept; and
2. $\cup m_i = m$ iff $m$ is a minimal set corresponding to a new aggregated concept.

**Proof for Condition 3**

Let $c_1$, $c_2$, and $c$ be concepts and $M_1$, $M_2$ and $M$ be a minimal set:

1. If $c_1 \rightarrow M_1 \land c_2 \rightarrow M_2$, then $c_1 \cap c_2 \rightarrow M_1 \cap M_2$ by the definition of the intersection operation. \( \therefore \) If $c_3 = c_1 \cap c_2 \land M_3 = M_1 \cap M_2$, then $c_3 \rightarrow M_3$;
2. If $c_1 \rightarrow M_1 \land c_2 \rightarrow M_2$, then $c_1 \cup c_2 \rightarrow M_1 \cup M_2$ by the definition of the union operation. \( \therefore \) If $c_3 = c_1 \cup c_2 \land M_3 = M_1 \cup M_2$, then $c_3 \rightarrow M_3$.

Since we aim to define a minimal (sub)set of an EXPRESS schema, it must conform to the EXPRESS syntax as well as to its definition. The conditions for guaranteeing the conformity of a minimal set to an EXPRESS schema are as follows. First, since a minimal set is also a type of EXPRESS schema, it consists of a set of ENTITIES, TYPES, FUNCTIONS, or RULEs. There must be at least one ENTITY in an EXPRESS schema. Others are optional.
Condition 4. (Requiring an ENTITY). A minimal set $M$ consists of a set of ENTITIES $E$, TYPES $T$, FUNCTIONS $F$, or RULES $R_w$. There must be at least one ENTITY in the minimal set $M$. Others are optional

$$M = \{E, T, F, R_w\}$$

where $M=$minimal set; and $E=$non-null list of ENTITIES, whereas others are possibly null lists: $T=$list of TYPES; $F=$list of FUNCTIONS, and $R_w=$list of RULEs.

EXPRESS supports inheritance and ENTITIES in a minimal set must inherit attributes and RULE clauses from their SUPERTYPES in order to be self-contained. The inheritance from SUBTYPES is optional. There are two strategies for including attributes and RULE clauses of SUPERTYPES in a minimal set: (1) the inheritance structure can be flattened: i.e., all the attributes and RULE clauses can be inherited into the lowest level ENTITY and the SUPERTYPES can be removed; or (2) the SUPERTYPES can be included in a minimal set as they are, without flattening the inheritance structure.

Yang and Eastman (2007) took the first strategy (BR08). Theoretically speaking, both approaches produce a syntactically well-formed minimal set. The second approach, however, is taken here because the first approach (the “flattening” method) requires a computationally complex and costly restructuring process. Suppose, for instance, there is more than one ENTITY in a minimal set that shares common attributes. The inheritance structure then needs to be rebuilt to avoid redundancy in the shared attributes. Moreover, once an inheritance structure is destroyed, it is practically impossible to rebuild the lost inheritance structure in a systematic way and to rename a newly defined SUPERTYPE. Yang and Eastman’s method cannot accommodate the second approach simply because it violates BR01: a base set must have only one ENTITY.

Condition 5. (Inheriting SUPERTYPES). If a minimal set $M$ contains an ENTITY $e$, then $M$ must contain all the SUPERTYPES $E_{ip}$ of $e$ if there are any. SUBTYPES of $e$ need not be contained in $M$

$$\forall M, e \in M \rightarrow \forall E_{ip}[e < :E_{ip} \rightarrow E_{ip} \subseteq M]$$

where $M=$minimal set; $e=$ENTITY; $E_{ip}=$set of SUPERTYPES, and $x <: y$ denotes that $x$ is a subtype of $y$.

In EXPRESS, ABSTRACT SUPERTYPES are SUPERTYPES that cannot populate instances. It is illegal to have an ABSTRACT SUPERTYPE without SUBTYPES in an EXPRESS schema. Thus, if an ENTITY, which is an ABSTRACT SUPERTYPE, exists in a minimal set and its SUBTYPES do not exist in the minimal set, then the minimal set is invalid.

Condition 6. (Normalizing ABSTRACT ENTITY). If an ENTITY $e$ in a minimal set $M$ is an ABSTRACT SUPERTYPE, then there should be SUBTYPES $E_{sh}$ of $e$ in the minimal set $M$

$$\forall e \exists M[[e \in M \wedge ABSTRACT(e)] \rightarrow \forall E_{sh}[e < :E_{sh} \subseteq M]]$$

where $M=$minimal set; $e=$ENTITY; $E_{sh}=$set of SUBTYPES; $ABSTRACT(x)=$function to test if $x$ is an ABSTRACT SUPERTYPE.

In EXPRESS, among four data types (i.e., simple, aggregation, named, and constructed data types), simple and aggregation types do not require any additional definition because they are predefined data types and also because implementers will know what they are without any additional declaration. However, named and constructed data types require additional definitions because they are user-defined TYPES.

Condition 7. (Declaring user-defined TYPES, FUNCTIONS, and RULES). If a minimal set $M$ contains an ENTITY $e$ and $e$ contains the label (or name) $u$ of $U$, which is a user-defined TYPE or FUNCTION or RULE (declaration), then $M$ must contain $U$. User-defined TYPES include constructed TYPES and named TYPES

$$\forall M, e, U, u[[e \in M \wedge USERDEFINED(U) \wedge LABEL(u, U) \wedge e \in M]$$

where $M$ is a minimal set, $e$ an ENTITY, $U$ a user-defined TYPE, FUNCTION or RULE, $u$ the label of $U$, USERDEFINED($x$) a function to test if $x$ is a user-defined TYPE, FUNCTION or RULE, and LABEL($y, z$) a function to test if $y=$label of $z$.

Condition 8 defines a condition for mandatory, OPTIONAL, and INVERSE relations. If the minimal cardinality of a mandatory or INVERSE relation between ENTITIES is set to 0, then the relation is also regarded as optional, even when there is no OPTIONAL keyword. In the current version of IFC (i.e., IFC 2.3), the minimal cardinality of all the INVERSE relations is defined as 0.

The same rule applies to the aggregation data types. An aggregation data type is a collection of either attributes or ENTITIES with the upper and lower bounds. The aggregation data types include SET, BAG, LIST, and ARRAY, and they can be interpreted as in self-reflective relations. If the lower bound (or minimum cardinality) of a SET, BAG, or LIST is set to 0 or omitted, then the aggregation data type is also regarded as in the OPTIONAL relation. However, ARRAY data types require the OPTIONAL keyword to be optional.

Condition 8 guarantees the inclusion of any referenced ENTITIES in a mandatory relation in a minimal set, whereas Yang and Eastman’s rule BR04 eliminates ENTITIES referenced by the attributes in a base set. However, Condition 8 is not strict about OPTIONAL elements (i.e., attributes or ENTITIES in the OPTIONAL relations). In order to make the size of a minimal set minimal, OPTIONAL elements should be removed. However, if OPTIONAL elements are excluded from a minimal set, then the minimal set may lose critical semantics. Thus, this rule leaves the issue of whether to include or exclude OPTIONAL elements open to the choice of implementers.

Condition 8. (Mandatory, OPTIONAL, and INVERSE relations). If a minimal set $M$ contains an attribute or ENTITY $e$, and $e$ is in a mandatory relation with another attribute or ENTITY $e_0$, then $M$ must also contain $e_0$

$$\forall M, e, e_0, R[[e \in M \land \Box R(e, e_0)] \rightarrow e_0 \in M]$$

where $M=$minimal set; $e$ and $e_0=$either ENTITIES or attributes; $R=$relation; and $\leq=$necessity operator.

Notes:
1. The INVERSE relation can also be categorized as mandatory and optional relations. The same condition applies to ENTITIES in an INVERSE relation; and
2. If mandatory or OPTIONAL ENTITIES and attributes are included in a minimal set, then RULEs, FUNCTIONs, and PROCEDUREs that are related to them should also be included in the minimal set (see Condition 7).

In some cases, RULEs defined in UNIQUE and WHERE clauses change the relationships between ENTITIES and attributes. An example is the WHERE clause of IfcActorRole in IFC2 × 3. IfcActorRole, exemplified below, is an ENTITY with one mandatory attribute Role and two OPTIONAL attributes UserDefinedRole and Description. Based on Condition 7, the two OPTIONAL attributes can be removed. However, the WHERE clause forces a type relation to be mandatory. It imposes a condition that says whether Role is not USERDEFINED, or if Role is USERDEFINED, then the attribute UserDefinedRole must exist in an ENTITY and is no longer OPTIONAL. In this case, the attribute UserDefinedRole should be interpreted as a mandatory attribute being included in the minimal set. In any case, a minimal set should satisfy all the RULEs in a minimal set.

ENTITY IfcActorRole;
  Role: IfcRoleEnum;
  UserDefinedRole: OPTIONAL IfcLabel;
  Description: OPTIONAL IfcText;
WHERE
  WR1: (Role () IfcRoleEnum.USERDEFINED) OR
  ((Role=IfcRoleEnum.USERDEFINED) AND
  EXISTS(SELF.UserDefinedRole));
END_ENTITY;

Condition 9. (Conforming to RULEs). If a minimal set \( M \) contains RULEs \( r_u \), then \( M \) satisfies all of its RULEs \( r_u \): i.e., \( M \) should include all the elements that are required to satisfy all of its RULEs \( r_u \).

\[
\forall M, r_u [r_u \in M \rightarrow r_u] 
\]

where \( M \)=minimal set, and \( r_u \)=rule.

Implementation

Based on the definition and constructions defined in “Conditions for Extracting Valid Subsets from Express Schema,” an early version of a software application called “IFC Model View (MV) Extractor” (Lee 2007) was developed using Microsoft.NET Framework 2.0. It is still an alpha version undergoing debugging processes. The programming language was VB.NET, which is an object-oriented programming language. Using its object-oriented capability, an EXPRESS parser was developed, which can load an EXPRESS schema as object-oriented classes on the memory for efficient search and retrieval. As its name implies, it was originally developed to support the IAI’s MVD activities. However, it should work with other types of EXPRESS schemas.

Fig. 2 shows a screenshot of the IFC Model View Extractor. The first pane deals with the concept-to-ENTITY mapping. It searches and lists all the ENTITIES that may be related to a concept that has been input. From the list, a user decides and selects an ENTITY that is relevant to this concept. Theoretically, it should be possible to search ENTITIES using synonyms or terms not used in a schema. However, the current version of IFC Model View Extractor searches only through ENTITIES that include a typed-in phrase such as “building” or “structure.” As a makeshift solution, a simple thesaurus for the AEC industry can be built in. Eventually, however, a larger and more complete dictionary will be needed. An effort has been initiated for developing this termi-
nology and ontology library for the AEC industry. This effort is being undertaken by the International Framework for Dictionaries (IFD) specification (ISO 12006-3) group composed of members from the International Construction Information Society (ICIS) for the AEC industry (IAI 2007). When this IFD work is done, the dictionary can be added to the module.

If a user double clicks an ENTITY from the first pane of the IFC Model View Extractor in Fig. 2, the IFC Model View Extractor automatically returns a valid minimal set in the fourth pane. This set contains the selected ENTITIES based on the conditions proposed in the previous section. It is technically possible to select multiple ENTITIES accumulatively or simultaneously and to create a minimal set that includes the selected ENTITIES. This type of function, however, has not yet been implemented in the current version. The second pane shows the entire IFC schema (or any schema in EXPRESS) for the user’s reference. The third pane lists the names of ENTITIES included in a minimal set shown in the fourth pane, again for the user’s reference.

The current version of the IFC Model View Extractor includes a function to include or exclude optional elements in a minimal set (see the bottom of the first pane in Fig. 2). Through test cases, it has been shown that, since many attributes in IFC are defined as OPTIONAL, many ENTITIES becomes empty if all of the OPTIONAL attributes are removed from ENTITIES.

The proposed conditions were validated through multiple iterations of EXPRESS syntax validation tests. The assumption was that, if automatically generated subsets passed the EXPRESS syntax validation test, the rules (conditions) that were used to derive them were also valid. The syntax of test results was checked first by using the EDM EXPRESS schema validation tool and then the conformity to the given conditions was manually checked. The proposed conditions were elaborated and the tool was debugged through multiple cycles of this process.

Since even the smallest sample for a building element can be very large, due to its inheritance structure, it is impossible here in this paper to show a full example of a minimal set related to a building element. Instead, an example of a minimal set for “Postal Address” in IFC2 × 3 is provided. The following minimal set was automatically produced using IFC Model View Extractor. The resultant minimal set properly includes the SUPERTYPES and user-defined TYPEs of IfcPostalAddress. Due to the WHERE clauses in IfcPostalAddress and IfcAddress, all of the OPTIONAL attributes are included in the minimal set. In the current implementation, ENTITIES in an INVERSE relation are ignored.

SCHEMA PostalAddress;
ENTITY IfcPostalAddress
  SUBTYPE OF (IfcAddress);
  InternalLocation: OPTIONAL IfcLabel;
  AddressLines: OPTIONAL LIST [1:?] OF IfcLabel;
  PostalBox: OPTIONAL IfcLabel;
  Town: OPTIONAL IfcLabel;
  Region: OPTIONAL IfcLabel;
  PostalCode: OPTIONAL IfcLabel;
  Country: OPTIONAL IfcLabel;
  WHERE
  WR1: EXISTS (InternalLocation) OR
    EXISTS (AddressLines) OR
    EXISTS (PostalBox) OR
    EXISTS (PostalCode) OR
    EXISTS (Town) OR
  EXIST (Country) OR
  EXIST (Supertype IfcAddress)
END_ENTITY;
ENTITY IfcAddress
  Purpose: OPTIONAL IfcAddressTypeEnum;
  Description: OPTIONAL IfcText;
  UserDefinedPurpose: OPTIONAL IfcLabel;
  WHERE
  WR1: (NOT(EXISTS(Purpose))) OR
    ((Purpose=IfcAddressTypeEnum.USERDEFINED))
  OR
    ((Purpose=IfcAddressTypeEnum.USERDEFINED))
  AND
    EXISTS(SELF:UserDefinedPurpose));
END_ENTITY;
TYPE IfcAddressTypeEnum=ENUMERATION OF
  (OFFICE
    .SITE)
  .HOME
  .DISTRIBUTIONPOINT
    .USERDEFINED);
END_TYPE;
TYPE IfcText=STRING;
END_TYPE;
TYPE IfcLabel=STRING;
END_TYPE;
END_SCHEMA;

This schema contains two ENTITY and three TYPE declarations. Suppose it had only the first ENTITY IfcPostalAddress. In this case, the schema would be invalid, for the attribute SUPERTYPE IfcAddress is totally unspecified or undefined.

Conclusions

This paper proposes a set of validity conditions for automatically extracting a minimal subset of an EXPRESS schema corresponding to a general “concept” and demonstrates how these conditions are implemented through an example. The notion of “minimal set” was introduced and formally defined.

This work is developed as an effort to automate a process of linking concepts to IFC views in the IAI’s MVD activities. The proposed approach is based on previous subset definitions by the ISO STEP Working Group (ISO 2001), IAI (Wix 2005b), Crowley et al. (1999), and Yang and Eastman (2007). Among them, Yang and Eastman proposed a set of conditions for generating a subset of an EXPRESS schema. There are similarities between the definitions and conditions proposed in this paper and those of Yang and Eastman, because both are based on the EXPRESS syntax. However, there are also fundamental differences between the two methods, as follows:

A minimal set, proposed in this paper, is the smallest “complete” subset of an EXPRESS schema that corresponds to a “concept,” whereas a base set proposed by Yang and Eastman is an “incomplete” set corresponding to an “ENTITY” in a schema. While a base set can be regarded as a “minimum functional unit” of an EXPRESS schema, a minimal set can be regarded as a “minimum semantic unit” that corresponds to a concept. The relationship between concept terms, or simply between concepts, and minimal sets is many to one. This implies that there can be
multiple concept terms that mean the same thing. The relationship between base sets and ENTITIES is one to one.

A base set per se is not complete as a subset. A valid subset is defined by combining several base sets. A minimal set is complete by itself. It also can form a larger subset (minimal set) corresponding to a different concept by combining several minimal sets.

In minimal sets, inheritance structures and all the SUPERTYPES in them are preserved as they are, in order to simplify the restructuring processes of multiple entities and their SUPERTYPES. Base sets, on the other hand, are constructed by inheriting all of the attributes and also by flattening the inheritance structures, for a base set can have only one ENTITY by definition, thereby eliminating its SUPERTYPE ENTITIES.

Since the notion of “minimal set” is based on semantic concepts, semantics is an important factor in defining subset generation conditions. A minimal set can be defined by either including or excluding OPTIONAL data types. If OPTIONAL data types are excluded, the size of a minimal set can be minimized. This type of minimal set, however, may lose some important semantic information kept in the main schema by removing all of the OPTIONAL data types. Our test cases have shown that many IFC ENTITIES becomes empty if OPTIONAL attributes are removed. Some conditions related to OPTIONAL data types are therefore relaxed.

The proposed method has been implemented as a software application, called “IFC Model View Extractor,” and tested with several cases. The syntax of the resultant minimal sets was checked first by using the EDM EXPRESS schema validation tool, while the conformity to the proposed conditions was checked manually. Through the test cases, the generation conditions were elaborated.

In order to fully implement the proposed concept-subset mapping method, a terminology and ontology library for defining “concepts” is needed. Currently, such efforts for the AEC industry are being undertaken by the IFD Specification Working Group (ISO 12006-3) and IAI (2007). It is expected that the proposed method can be helpful to various subset specification efforts in the AEC industry such as IAI’s MVD and ProIT, and other similar enterprises in other domains based on EXPRESS.

Acknowledgments

This work was supported by the Korea Research Foundation Grant funded by the Korean Government (MOEHRD) (Grant No. KRF-2006-8-0886). The writer is grateful to Kiyong Lee for his constructive comments.

References


