USING INFORMATION DELIVERY MANUAL (IDM) FOR EFFICIENT EXCHANGE OF BUILDING DESIGN INFORMATION

Changhwan Kim [1], Soonwook Kwon [2], Seokjoon You [3], JungA Lim [4]

[1] Graduate student, Dept. of Civil, Architectural and Environmental System Engineering Sungkyunkwan Univ. Suwon 400-746, S. Korea, deestynova12@naver.com
[2] Associate Professor, Dept. of Civil, Architectural and Environmental System Engineering Sungkyunkwan Univ. Suwon 400-746, S. Korea, swkwon@skku.edu
[3] Researcher, Construction and Environmental Research Center, Sungkyunkwan Univ. Suwon 400-746, S. Korea, seokjoon.you@gmail.com
[4] Graduate student, Dept. of u-City Design and Engineering Sungkyunkwan Univ. Suwon 400-746, S. Korea, k4rangj@nate.com
Correspond to Professor Soonwook Kwon (swkwon@skku.edu)

Abstract

In the new era of computerized building information, communication in a design and construction project demand transactions of large volume building model data containing various engineering domains, which causes increased costs in handling large data and communication problems due to poor understanding about which data should be shared. In this study, the authors converted Information Delivery Manual, which is a documentation format for capturing workflow information among AEC project participants, into an UML-based conceptual model and then created a database schema based on it. A client-server application for browsing and editing the database was also created.

KEYWORDS: IDM(information delivery manual), Industrial Foundation Classes, Object-oriented, Relational Database, Information Model

INTRODUCTION

Background

Modern AEC (architectural engineering and construction) industry heavily relies on computer software. The industry has become 'information-centric', where most of its tasks are involved with digitalized design and engineering information to create new one with added values. As information technology advances, newer methodologies have been introduced to the area, such as three-dimensional geometric models, and the information for the AEC applications are becoming more complex than it used to be.

Today, most modern 3D geometry-based AEC software applications claim that they support BIM (Building Information Modeling). BIM incorporates 3D Geometry with various engineering properties associated with it; the associated properties ranges from physical properties to unit cost and delivery schedule for a specific part represented by the given geometry. Each BIM software application utilizes its own definition of the geometry model and associated properties - in other words, BIM definitions.

To facilitate interoperability with such applications, Industry Foundation Classes (IFC) were introduced in late '90s. IFC was designed to be a 'neutral' model of AEC software applications, having a 'core' definition of common building elements shared by the applications.

Interoperability issues with IFC

While IFC intends to capture holistic information model of a building (product) to become a single reference for all AEC business areas, there are several issues that prevent such ambitions from realization:

IFC cannot define all of the information carried by arbitrary software unless they are voluntarily reported by the vendor (who are less enthusiastic to IFC-based software interoperability)

'one-model-has-it-all' approach is criticized as too complicated by software developers who need to implement only a subset of the complete IFC schema.

Table 1 : Number of IFC entities actually needed for a specific task (for this table, tasks of structural engineering)

Stages	Required information items	Number of IFC entities
Architectural design	Frames, walls, finishes, MEP, etc.	880
Structural design	Structural frames only (Columns, load bearing walls, beams, etc.)	149
Structural analysis	Nodes, elements, loadings, stresses	107

To simplify implementation, handful of techniques have been introduced – among them, SABLE (Simple Access to the Building Lifecycle Exchange) and IDM (Information Delivery Manual) are notable (both were sponsored by IAI-now Building Smart, the creator of IFC). SABLE uses a simplified (subschema) of IFC, especially 'flattened' entity hierarchy, to make implementation easier, whereas IDM provides a task-specific 'process manual' for clear identification of IFC entity subsets. For this research, we focus on the descriptive nature of IDM, as it may help filling gaps between the IFC implementations by providing more specific definition of their information needs.

IDM (Information Delivery Manual)

IDM (Wicks 2007) allows describing information requirements of an AEC application in controlled manner. It first states individual tasks of an AEC project, along with the information items associated with the tasks, which is further broken down into more specific IFC entities. Also, dependency between the individual tasks are defined so that someone can identify which information is required to pass; also, a software developer can understand which part of the IFC need to be actually implemented to serve a specific task.

On the other hand, IDM is basically a documentation for serving both software developers and expert AEC engineers altogether, using less 'IT terms' for better readability for those not having background knowledge; it is not initially intended for direct translation into computer programs as IFC is. For that reason, IDM's potential for facilitating interoperability is limited. Still, IDM offers structured approach for summarization of the required information, making itself a good start point for converting it into a computer-readable format.

RESEARCH METHOD

For figuring out which information needs to be exchanged between two interoperating applications, it is efficient to extract a subset of an entire IFC schema by analyzing the application-provided (thus computerized) IDM at runtime. To implement such functionality, we defined an information model for computerizing IDM, which is primarily a documentation format. With the definition of the IDM info model, we also implemented a database of IDM that are accessible from AEC applications to indentify which information items are required to pass on when they need to interoperate.

To define the IDM information model in object-oriented manner, we chose Unified Modeling Language (Booth et al, 1998) as a modeling formalism because UML is independent to any specific system type. For implementation of the defined IDM model, we used Microsoft SQL Server for generating the database schema that corresponds to the IDM model.

RELATED RESEARCH

Beside IDM and SABLE, several studies have been known regarding simplification of the IFC-based building information exchange. In Lee et al (2005), information of the tasks are collected, and then converted into a 'reference model', which states a required information item for exchanging data between the tasks. A collection of the 'model segments' of the tasks in a given domain (e.g. design of a building) will constitute a usable domain model (which is similar to IFC for AEC industry). It is being explored that the same structured survey approach might be used for deriving a reference 'subset' of a unit task out of IFC, which can be realized by mapping the derived reference model into IFC compliant subset, such approach might require advanced logics because the model structure might not be simply converted.

International Framework for Dictionaries (IFD) also provides a solution for 'impedance mismatch' problem between the application models. In its simplest form IFD is a mechanism that allows for creation of multilingual dictionaries or ontologies. IFD Library is one of the core components of the buildingSMART technology. IFD Library is a reference library intended to support improved interoperability in the building and construction industry. IFD Library provides a flexible and robust method of linking existing databases with construction information to an IFC based Building Information Model (BIM). Due to the multilingual capabilities of IFD Library this linking is language independent as well.

Construction Operations in Building Information Exchange (COBIE) provides simple exchange of IFC-derived information over spreadsheet software such as Microsoft Excel. COBIE intends the exchange of facility life-cycle data for use by facility management professionals over IFC. The COBIE team selected spreadsheets so that they can be widely used by smaller firms, not just on large ones.

MODELING IDM DOCUMENT FORMAT

IDM consists of three parts: model of a reference process, Exchange Requirements (ERs), and Functional Parts (FPs). Each part is a composition of several fields whose formats are also defined in the IDM specification. Also, an IDM part constitutes a complete document having its own attributes such as document name, identification code, change history, etc. Actual contents of the individual fields depend on their definition, which can be seen as similar to concepts in a conceptual model, or objects in an object-oriented information model.

IDM model of basic document structure

In IDM, the header section (collection of fields that are common to all parts) has the following fields: change log, identifier, and name. We defined change log as a separate entity because it provides vital information on the document (also an information model) history; on the other hand, the rest of the fields were not defined as separate objects, as suggested in object-oriented modeling methodologies such as Rumbaugh (1991), because there was little to gain by taking that suggestion here: as we use MS SQL Server, a relational database system, object inheritance is harder to implement.

Many fields that constitute an IDM document consist of natural-language paragraphs describing themselves. Rather than providing extra data structure for parsing those descriptive phrases, we define them in a variable-length text string data type. There exist some cases in such paragraphs that refer to other fields; however, most of the referential relationships are explicitly defined in elsewhere, mostly in other fields – if not, they can be represented using embedded tags (e.g. XML tags). merging point clouds from hetegenerous sources.

Table 2 : Definition of header elements, defined as an abstract base class of all IDM objects

Object class	Role	Attribute	
IDM	abstract base class of all IDM document	none	
change log	Tracking history of the IDM document	date	Date of creation
		email	Email address of the creator
		description	Specific description of the changes

Modeling process maps

A process map is a collection of individual processes (or tasks). Along with the tasks, their dependency (for instance, which tasks has to be done first in order to others) are defined as directed arcs between two tasks. In IDM document standard, unlike other parts, the process map is required to be defined using Business Process Modeling Notation (BPMN), not textual paragraphs as other parts are. For simplicity, all BPMN diagrams are currently treated as images- this means that, in our model, each process map is the minimal unit of representing design tasks in a project. Table 3 lists object classes defined for representing the process map.

Table 3 : Definition of IDM Process Map objects

Object class	roles	Attributes
process model	Represent collection of process maps to represent a project; related to exchange requirements	name
		identifier
		IFC release
		overview
process map	Collection of unit tasks which constitute a specific process in a project (Defined in BPMN)	BPMN-based FILE
process explanation	Individual tasks in a process map	process ID
		type
		name
		documentation

Modeling Exchange Requirements

Exchange requirements are defined using two key classes: an 'Exchange Requirement' class , which is a main body of the part, and an 'Information Requirement' class, which is used for listing the required information in a specific task or a process map. Two additional classes, 'information type' and 'required level', are used for giving additional information for those requirements. For representing actors (i.e., consumer of the required information), we added an extra field to Information Requirement class.

Table 4 : Object classes defined for representing exchange requirements

Object class	Roles	Attributes
exchange requirement	Listing information items needed for exchange	name
		identifier
		overview
information requirement	Collection of functional parts to be exchanged between unit tasks	information needed
		actor supplying
information type	Type of the information (e.g., file type)	name
required level	Specifies whether given information is essential for exchange	name required, recommended, optional
project stage	To which given ER is applied to	stage code
		stage name

Modeling Function Parts

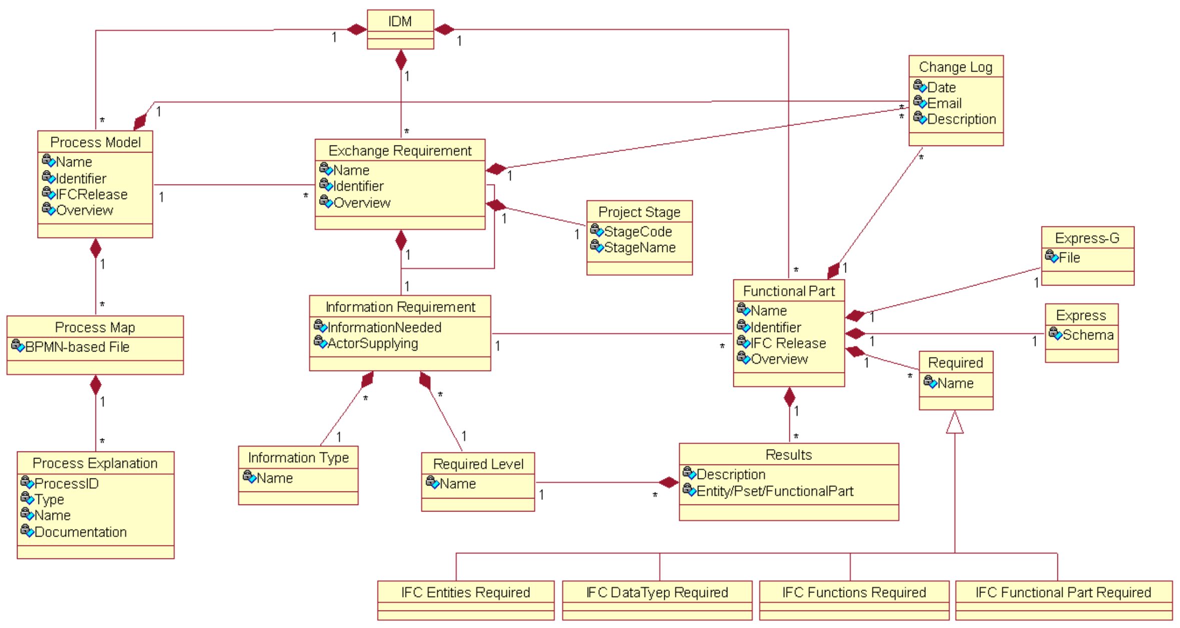
Functional Parts (FP) directly represent a set of IFC entities that represent specific part of a building. Functional Parts are the key to finding the necessary IFC subset for exchanging information two applications. Unlike other IDM parts, Functional Parts consist of structured information such as IFC entities and property sets.

Table 5 : Object classes defined for representing functional parts

Name	Role	Attribute	
functional part	Overview of the FR, and information of constituent IFC counterparts	name	
		identifier	
		IFC release	
		overview	
EXPRESS	A collection of IFC entity definition for the specific functional parts	schema	Defined in EXPRESS (doesn't contain whole IFC schema)
EXPRESS-G	A graphical notation of EXPRESS	file	EXPRESS-G diagram which represent the desgnated sub schema
required	Corresponds to the 'required levels' in Exchange requirements	name	One of the three: required, recommended, optiona
results	A result of the exchange process. It could be another IFC entities or other functional parts	description	
		entity/pset/ function/part	

Summing up

Figure 1 illustrates relationships of the IDM object classes we defined.



F
igure 1 : UML class diagram of the IDM document format

CREATING AN IDM DATABASE USING OUR MODEL

Converting UML class definition into relational database schema is rather straightforward except for mapping class hierarchies and one-to-many relationships. For referential relationships between the classes, they can be represented using foreign keys. For one-to-many relationships, additional tables were added (for instance, table ProcessModelExchangeRequirement is used for holding multiple exchange requirement objects referenced by ProcessModel object). Most descriptive information can be stored using variable character string field. We translated our UML model of IDM into the database schema by hand. To write the DB schema, we used Transact SQL, a Microsoft's flavor of Structured Query Language (SQL-92). To populate the schema with real IDM data, Microsoft SQL Server 2000 was used as database management software (DBMS). To allow access to the database (for the applications and the users), the database schema was slightly modified to have additional fields. We are working on the software interface that allow applications to retrieve the IDM database information.

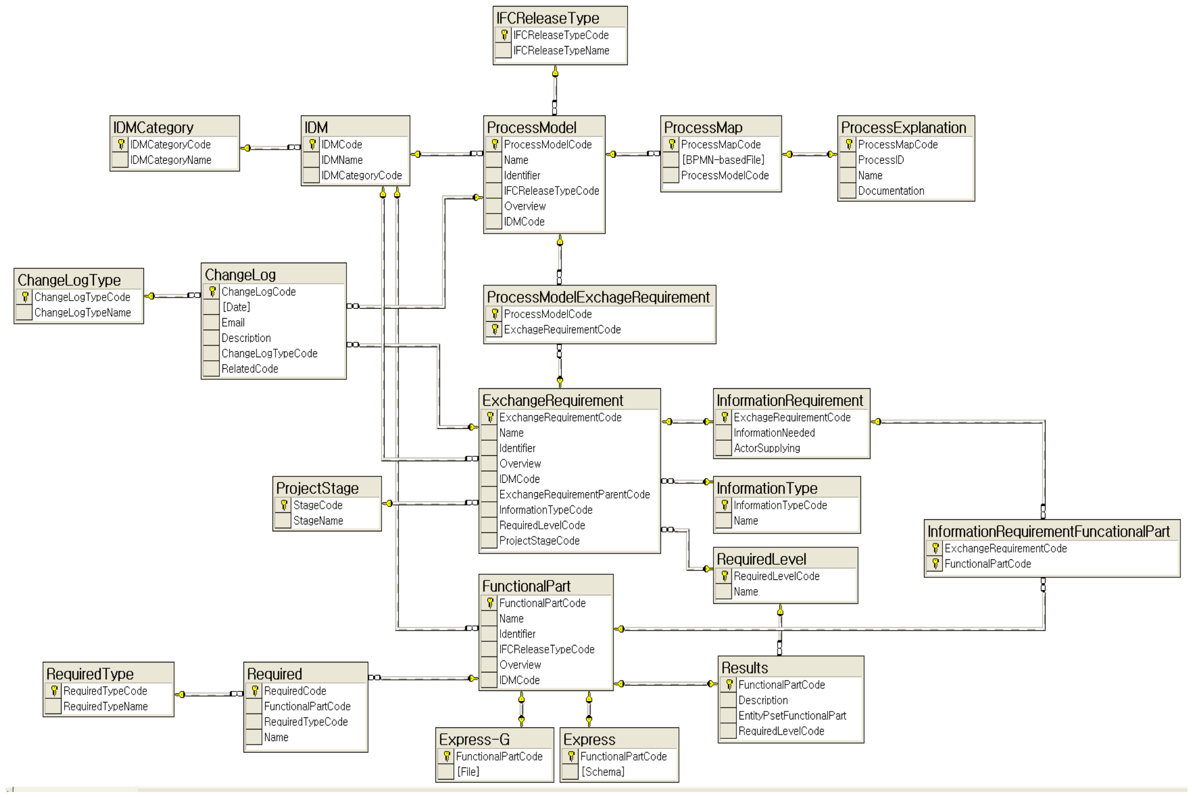


Figure 2 : Relationships between the IDM database tables. They are converted from the UML model in Fig. 1

CONCLUSION

In this paper, we defined an information model of information exchange process, which is based on Information Delivery Manual (IDM) document specification. With the IDM information model, we defined a database schema to store the IDMs. The database, when fully populated with standardized IDMs, can help understanding how information is managed throughout the building design project. The database can also be used by IFC-enabled AEC software applications for identifying an IFC entity subset required for a specific interoperability scenario.

ACKNOWLEDGE

Part of this work was funded from Korea Institute of Construction and Transportation Technology Evaluation and Planning for its Construction Technology Innovation Program (Project code: 06-‘hightech convergence’ E01).

This work is financially supported by Korea Minister of Ministry of Land, Transport and Martime Affairs(MLTM) as U-City Master and Doctor Course Grant Program

REFERENCES

[1] Ambler, S.W.,. Mapping Objects to Relational Databases. An AmbySoft Whitepaper. AmbySoft Inc., Http://jeffsutherland.com/objwld98/mappingobjects.pdf, 1997

[2] BuildingSmart Norwayk, IDM official website. http://www.iai.no, Norway, 2006

[3] BuildingSmart Norway, IDM in Confluence, http://idm.buildingsmart.no/confluence/homepage.action, Norway, 2006

[4] Booch G, Rumbaugh R, and Jacobson I, The Unified Modeling Language User Guide. Addison-Wesley, Reading MA, 1998. ISBN 0-201-57168-4.

[5] ISO 10303-11:2004, Industrial automation systems and integration? Product data representation and exchange? Part 11: Description methods: The EXPRESS language reference manual, 2004

[6] Lee, G., Eastman, C. M., and Sacks, R., "Generating IFC views and Conformance Classes using GTPPM." Joint International Conference on Computing and Decision Making in Civil and Building Engineering (ICCCBE), Montreal, Canada, pp. 1715-1724., 2006

[7] Rumbaugh, R et al, Object-Oriented Modeling and Design. Prentice-Hall. Englewood-Cliffs NJ, 1991. ISBN 0136298419 0-136-29841-9

[8] White, S.A., Introduction to BPMN. Object Management Group, http://www.bptrends.com/publicationfiles/07-04% 20WP% 20Intro% 20to% 20BPMN% 20 - % 20White.pdf, 2004

[9] Wix, J., Information Delivery Manual Guide to Components and Development Methods (version 1.1). buildingSMART Norway.http: / / www.iai.no/idm/idm_resources/idm_methods_guides/IDM2_Methodology_20071022.pdf 2007.