MDA-based facility management applications under BIM

Chia-Ying Lin , Chien-Cheng Chou *

Department of Civil Engineering, National Central University, Taiwan
** Corresponding author (ncuceit@gmail.com)*

Purpose The recent research trend of building information modelling (BIM) involves interoperability and issues pertaining to the design, construction, operations, and maintenance phases of a building. However, few researchers are concerned with problems encountered in the last two phases of the building life cycle. Since BIM has been widely used in the design and construction phases, further expanding the BIM data stream to the post-construction phase can not only establish a consistent, shared database for information exchange between the phases but assist in current facility management applications. This research reconceptualises BIM with object-oriented thoughts to a space-based representation, and tries to construct an interface between BIM and existing facility management software. **Method** Model-driven architecture (MDA) is one of the modern software design approaches. With its foundation – meta-object facility (MOF) – and model transformation, MDA provides an ideal solution for easily maintaining the interoperability. In this research, a MOF of a building is constructed by imitating the MDA approach, encapsulating BIM data and providing an interface for external facility management software. This MOF of a building is designed with spatial interference due to the maintenance and repair work is usually based on the space. **Results & Discussion** A conceptual demonstration of disaster mitigation is conducted to test the feasibility of using MDA to encapsulate BIM data for extended applications. Furthermore, the expected results are reconstructing maintenance records to a space-based database for a better software design and a solution for interoperability.

Keywords*: information technology; BIM; model-driven architecture; facility management*

INTRODUCTION

Building Information Model (BIM) technologies promise to provide a consistent communication and information exchange platform for all stakeholders involved in a building life cycle, as well as to create a 3D display environment to clarify a building's virtual representations. Many researchers study BIM-related issues, such as design and engineering, linking to analysis tools, energy innovations, facility management, and so on[1]. However, most studies concerned with the design and construction (D&C) phases of a building, few studies deal with issues pertaining to the operation and maintenance (O&M) phases[2, 3].

In O&M phases, many applications, such as disaster mitigation (which can be regarded as the most important field in the building industry), security, community care, require building information in a different way within the AEC industry, but these extended usages are seldom considered in BIM. Also, in these fields there is mature software for their applications, and BIM cannot cover all the fields which need building information as a possible input. As a result, the proposed approach suggests to follow the open/closed principle (OCP) – "open for extension, but closed for modification"[4]. For example, structural elements in a building will not be changed during their entire life cycle[5]; and thus, it may be a good way to operate such information just at the class level, not at the metaclass level to avoid future modification. For example, a house has four rooms. These four rooms should be represented as four classes, and their objects should represent the conditions at specific time instances. The traditional IFC-based approach treats these four rooms as four instances; and thus, applications in the O&M phases cannot operate at the correct level to seamlessly retrieve information from the D&C phases.

Government agencies usually retain a copy of building blueprints and related documents of D&C phases However, they are seldom updated unless special reasons. In O&M phases, disasters are hard to be predicted and may occur in different forms, and fires may be the most common form in a building environment. Investigation of fires may be conducted after a fire occurs, and fire scene reconstruction is one of important parts. Currently the Fire Dynamic Simulator (FDS) performs well and is widely used for it[9], but a good database may be needed to store correlated data. The proposed approach may provide a new model for properly retrieving information from BIM and for being able to store fire investigation data in the BIM-derived model, so a preliminary research result presented here can show how to deal with the interoperability issues of the information flow between the design/construction phases and the operations/maintenance phases.

RELATED WORKS

Nowadays, the software development process is facing the situation that requirements become more complicated and different platforms can be chosen as the working environment. To solve these problems, MDA was proposed by Object Management Group (OMG), mainly for the purpose of integration and interoperability[6]. This approach basically implements software development by using formal models.

In MDA, models occupy a significant position. Generally, model is used to describe and simplify the real world. Furthermore, metamodel is needed to define the description of a model. Take paintings for example, as shown in figure 1, a painting can be viewed as a model of a real world thing; actually, it is composed of lines and shapes, therefore, these shapes are the model description which is necessary to form a painting. In the higher layer there may also exists other abstraction to describe M2 layer[7].

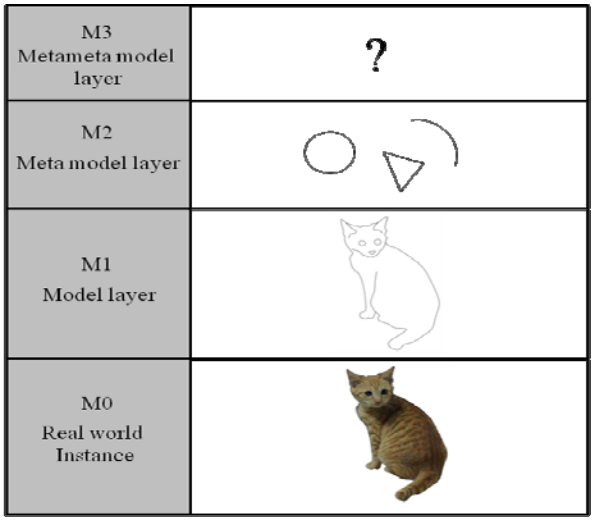

Fig.1. Common models

To take an overview of MDA, at first the separation of concerns should be introduced. Three MDA viewpoints of a system are described as follows: (1) Computation Independent Viewpoint (CIV), (2) Platform Independent Viewpoint (PIV), and (3) Platform Specific Viewpoint (PSV). CIV contains none of the computer-related processing details and only focuses on the business requirements; PIV is about the operations of a system but does not contain the details for a specific platform; PSV integrates PIV with the details of using a particular platform as the development environment[8].

As shown in Figure 2, four layers and their transformation mechanisms are identified based on these views. A computation independent model (CIM) is constructed from CIV. Based on the CIM the platform independent model (PIM) is created. With a sufficiently complete and precise PIM, platform specific model (PSM) can be generated by model-to-model transformation mechanisms, and the specific code model – which can be viewed as implementation – can be automatically transformed from PSM.

OMG defines a standard – Meta Object Facility – to provide metadata management and modelling language definitions. MOF is used with a metamodel hierarchy shown in Figure 3. A run-time system can be interpreted by a UML model, since the UML model is design by people, there can be many UML models from different perspectives. To describe a UML model, we need to define UML descriptions and notations as a communicating method. The UML definitions are also based on the definitions of MOF. With the descriptions of higher layers, the lower layer can be clearly explained.

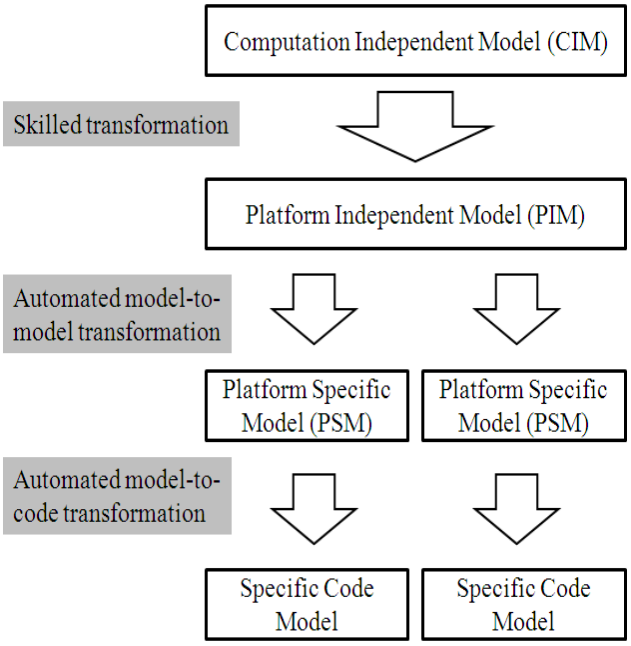

Fig.2. MDA Process

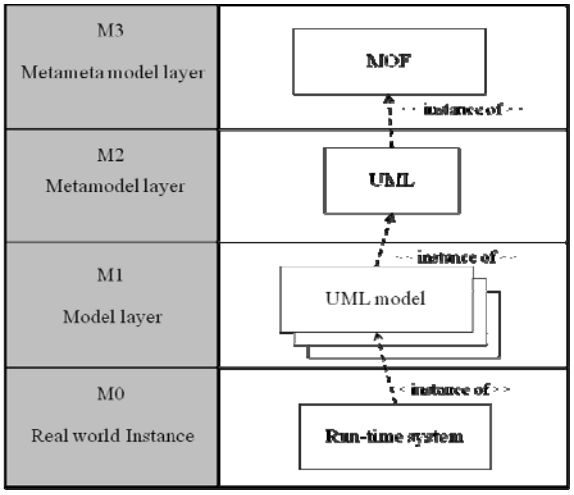

Fig.3. Metamodel hierarchy

RESEARCH APPROACH

Because of the wide use of BIM, building information is available for stakeholders in the O&M phases. However, the data structure is not easy to be used

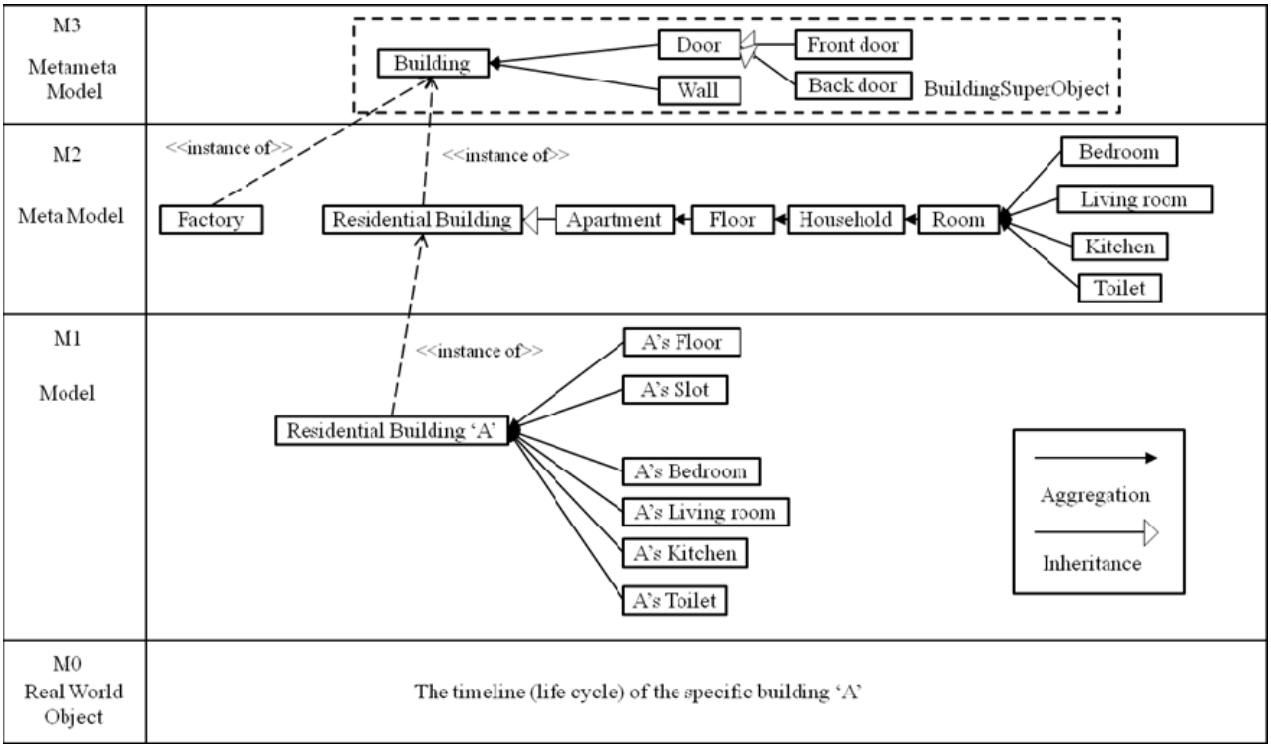

Fig.4. Metamodel hierarchy of a building

for users, and needs extra efforts to write programs for further use of this information. In order to offer a better way for the extended use, this research defines a metamodel hierarchy of building to interpret elements of a building.

Figure 4 shows the sample definitions of the metamodel hierarchy of a residential building. This hierarchy is designed as the basis of the proposed approach.

M3 layer

For different perspectives, the contents in M3 layer should be viewed as different roles. Inside this hierarchy, the M3 layer describes basic components of a building, such as doors, walls, windows, and so on. These components exist in any type of building, in other words, they are essential for a building; as a result they are designed to be in the M3 layer. Outside this hierarchy – which means extended applications without sufficient understanding of the building internal structure and BIM programming – for them, the layers below and those basic components are encapsulated due to OCP. Building information is available for them through the "BuildingSuperObject" but its data structure cannot be changed from the outside, that is, "BuildingSuperObject" is an interface as a communicating bridge.

M2 layer

This layer inherits basic components from the M3 layer, as well as different types of buildings and their basic units are described here. Different types of buildings may have rooms/zones/spaces for different purposes, and rooms are classified by purposes as a basic unit. For example, an apartment is classified as one type of residential buildings. Bedrooms and living rooms may only exist in a residential building, not in a factory.

M1 layer

In the M1 layer, a "model" means a specific building and is constructed here. Most of the static components are inherited from higher layers. Also, the constant spatial components of a building are described here, since most of the time the floor plan of a building will not be changed after the D&C phases.

M0 layer

Finally, the instances of M1's building are stretched by time as a timeline and record the whole life cycle, which means the building information covering the D&C phases to O&M phases exists in this layer.

Suppose BIM data of a specific building is available. First its spatial relationship and the usage of spaces are re-drawn as the model shown in the M1 layer of Figure 4. The class library of this building can then be automatically generated by the model. Since attributes are also predesigned to be contained in the model, extended applications can use data they need through the model.

As shown in Figure 5, the metamodel hierarchy of building is simplified as the bold rectangle, and an interface is connected with the top layer – BuildingSuperObject. Those applications need not

to know the operation of the hierarchy; also, the applied domain is changeable. This design provides flexibility of the proposed model. It can be used not only for one specific application, but for providing better flexibility and reusability.

EXAMPLE

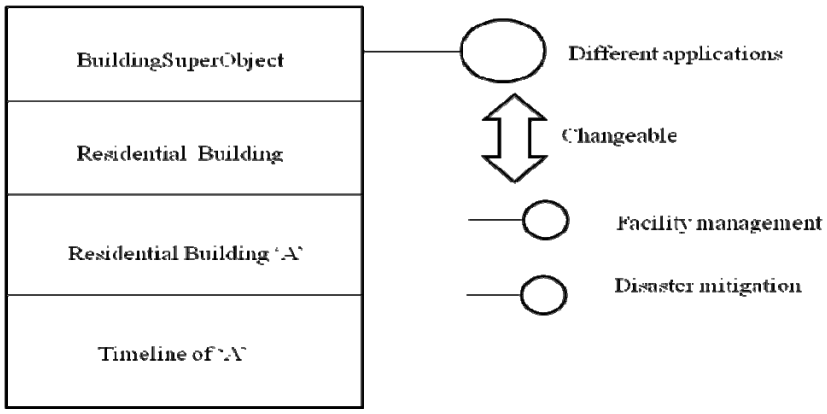

Fig.5. The concept of interaction between different applications and the metamodel hierarchy of building

an elevator. There are two households in each floor, and the rule of the address assignment is the floor number, plus the serial number. Each household is assumed to have three rooms, i.e., living room, bedroom and toilet. In Figure 6, rooms in address '02' are omitted. These rooms inherit attributes from their designed usage respectively. In addition, from the spatial perspective, 'B' can be viewed in two dimensions: horizontal and vertical. The horizontal dimension is equivalent to the floor concept; the vertical dimension, named "slot" here, is composed of rooms located on the same vertical line. Usually the floor plan of a household is the same as its neighbour upstairs or downstairs in an apartment; therefore, rooms of the same purpose belong to the same slot. However, the elevator is independent of a floor or a slot since it is movable.

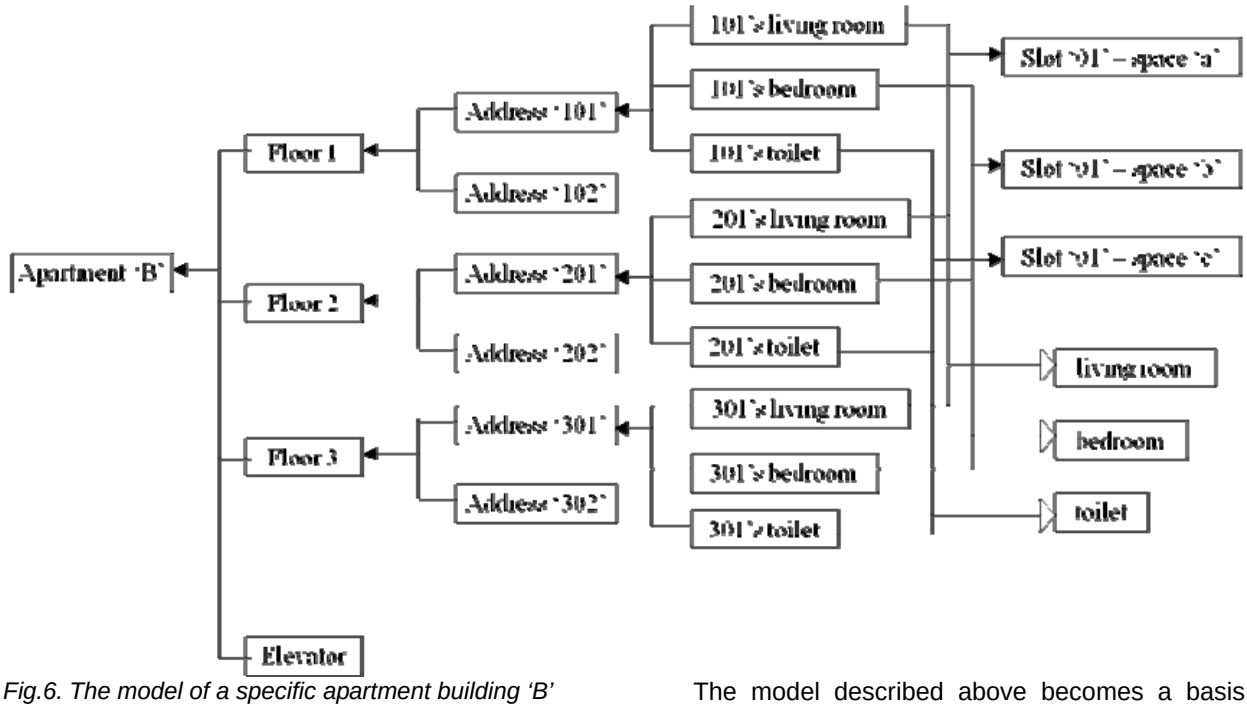

Fig.6. The model of a specific apartment building 'B'

Since the more floors and rooms a building contains, the more complicated the problem is. A three-floor apartment is presented in this section as a simple example to show the contents in metamodel hierarchy of an apartment building 'B'. Some details are also omitted for clearly interpreting the proposed approach. The relationship between contents and the expected application of the metamodel hierarchy are also interpreted in this section.

Figure 6 presents a sample model constructed following the metamodel hierarchy in previous section. In apartment building 'B', there are three floors and

The model described above becomes a basis for automatic code generation; a class library of 'B' is constructed for extension. The extended applications can use BIM data through this model – more specifically, users of these applications access BIM data through this model instead of directly operating with BIM. Since it is not easy to cross the threshold of the programming issues for BIM applications, it is believed that using this model will assist the development of extended allocation for BIM.

As suggested before, now building information is reconstructed to a spatial-based form. The fire investigation result can be store in BIM room by room and with the spatial relationship, such as slot and floor, it

can also perform several simple spatial inference function, for example, the charred facility may indicate how the fire go through.

CONCLUSIONS

This research has proposed a software architecture to integrate the static (BIM) information with the dynamic, O&M-related data. The MDA technique was utilized to store and synthesize the data. The BIM information is used to show the geometry aspect of the building. MDA has been also applied to other industries' applications, including the AEC industry. Further enhancement of the system is needed in order to integrate more dynamic information from different domains correlated with O&M phases and/or other BIM programs.

References

1. Becerik-Gerber, B., Kensek, K., "Building Information Modeling in Architecture, Engineering, and Construction: Emerging Research Directions and Trends", *Journal of Professional Issues in Engineering Education and Practice*, Vol. 136(3), pp.139-147, 2010.
2. Akcamete, A., Akinci, B., Garrett, J.H., "Potential utilization of building information models for planning maintenance activities", *Proceedings of the International Conference on Computing in Civil and Building Engineering*, Nottingham, UK, 2010.
3. Vanlande, R., Nicolle, C., Cruz, C., "IFC and Building Lifecycle Management", *Automation in Construction*, Vol. 18, pp. 70-78, 2008.
4. Meyer,B, "Object-Oriented Software Construction", Prentice Hall, 1988.
5. Hassanain, M.A., Froese, T.M., Vanier, D.J., "Development of a maintenance management model based on IAI standards", *Artificial Intelligence in Engineering*, Vol. 15, pp. 177-193, 2001.
6. Soley, R., "Model Driven Architecture", *White Paper, Draft 3.2*, Object Management Group, 2000.
7. Gaševic, D., Djuric, D., Devedžic, V., "Model Driven Engineering and Ontology Development", Springer, 2009.
8. Miller, J., Mukerji, J., "MDA Guide Version 1.0.1", *Object Management Group,* 2003, http://www.enterprise-architecture.info/Images/MDA/MDA%20Guide%20v1-0-1.pdf, accessed August 2011.
9. Shen, T.S., Huang, Y.H., Chien S.W., "Using Fire Dynamic Simulation (FDS) to reconstruct an arson fire scene", *Building and Environment*, Vol. 43, pp.1036-1045, 2008.