Meta Object Facility to Access and to Integrate Building Information Models

Torsten Krämer, Wolfgang Huhnt
TU Berlin, Germany
Torsten.Kraemer@tu-berlin.de

Abstract. Computationally aided construction management tasks require the integration of information provided by numerous domain disciplines. The required information is authored in software products specifically tailored to the needs of each domain and may not be available in the desired structure. A uniform data handling concept is sought in order to integrate all available different data sources. The Meta Object Facility (MOF) defines a standardized modeling system which is successfully applied in software engineering projects. In this paper the usage of MOF-models as a uniform way of data handling is proposed for application development in civil engineering. The necessary steps to acquire information from highly specialized software products up to the integration of many model fragments into an interconnected model system will be discussed.

1 Introduction and related work

Since a couple of years the construction industry is steadily turning towards a computational model based design approach. Early steps were taken in the mid 1970s through to the 1980s as reported in Eastman (1992). Already in the early stages it became clear that for real building design support there must be more than collecting lines for drawings. The idea of modeling building-component wise on different levels of detail began to arise. The names of this approach varied at this time from “building model” to “building product model”. One of the first commercial design systems was Graphisoft’s ArchiCAD1 with the “Virtual Building” approach in the late 1980s (Laiserin 2003). The term BIM appeared in a marketing paper, was pushed up by Laiserin in 2003 and finally adopted by the AEC software industry (Howell & Batcheler 2005). BIM can be seen as the acronym for either Building Information Modeling or Building Information Model. The first one is the preferred meaning by Eastman, since the term ‘modeling’ emphasizes the overall procedure (in Eastman et al. 2007). To him it is crucial to take all the communication, interaction and advancements into account, as they will induce broad process changes in construction. Building Information Modeling in its core still needs an agreement on the digital data model to use and therefore BIM is sometimes referred to as the acronym of Building Information Model. Fully agreeing on the importance of seeing BIM as a way of acting, working and organizing, the focus of this paper lies on the data model.

Collecting the necessary use cases of the data model and looking at the many different stakeholders and tasks to fulfill in design and construction up to the needs of operating the final structure, many different kinds of data have to be integrated into the overall model. The data in BIM is not just a collection of independently gathered und unrelated information. It is more a very complex and highly coupled system of objects and object-properties. A change in one property of a single object often affects properties of many different other objects. Thus ensuring consistency of the whole model is one of the big issues. From the architectural point of view there is clearly the need to store the shape, the geometric properties, the dimensions,

1 http://www.graphisoft.com/
the proposed materials and required qualities of the components in the model. Since architectural design is more of an iterative process and especially the construction industry is very open to late time changes of their products, these pieces of data have to be organized and formulated in a highly parametric way - respecting and reflecting all the interconnections of the components. For preparing the bidding phase as well as for planning work on the construction site the model has to store material properties of each component, required quality and the production techniques applicable to realize the composite structures. Each single component has to be clearly identifiable. Correct and complete quantified lists of required building material have to be derivable. A huge part of data originates in the inclusion of building equipment comprising HVAC, electrical and communication installations and other supply lines. Turning towards a more operational point of view the resulting structure has to be partitioned into manageable zones and units. Those partitions then can be associated ranging from their available features, their characteristics up to financial operational data like rental rates and performance measurements.

Looking at today’s software landscape there is not a single software solution available which could serve as a data model for all those different kinds of data. The major CAE software products such as Autodesk’s Revit® family\(^2\), Graphisoft’s ArchiCAD®\(^1\) or Bentley’s Architecture V8i\(^3\) – labeled today as BIM software – are very powerful geometrical parametric modelers for building components. They provide some more flexibility in storing some selected different kinds of additional data from the above mentioned variety but fail to serve as a truly all-embracing model. Data about production techniques and performance values is stored in company’s common databases often organized in a distinct relational way. For sequencing and scheduling specialized software products are the preferred choice of tools. Structural and energy analysis are carried out in very specialized software components which store their datasets in their own particular way. Bills of material and contractual data are stored in specialized data formats quite independently of other information concerning the building project.

![Specialized software provides many different data sources](image)

Figure 1: Specialized software provides many different data sources

\(^2\) [http://usa.autodesk.com/revit-architecture/](http://usa.autodesk.com/revit-architecture/)

There are efforts to unify the scattered data in a single data model. The Industry Foundation Classes (IFC), developed by the buildingSMART International Ltd.⁴, is a standardized data model with the most advanced approach. It can be traced back to the original efforts around the STEP initiative and is now an ISO standard of its own (ISO IFC2x, 2005). Looking at the different fields above, the IFC standard can only cover a subset of all the capabilities of the specialized data models of the software (Lee 2011). So if a user wants to make use of all capabilities a software-tool provides, the IFC data model cannot be used as a persistent storage format. Out of this reason it is not possible to establish a complete data exchange on the basis of IFC. The same is true to concepts of so called BIM Servers. A BIM server takes a role as a central database where each application can check out a specific view and write back its own data. On the one hand centralized storage of all project data gives a single point for consistency checking but on the other hand it may not always provide the storage capabilities a specific application might need. This either leads to storing the same data twice as a common shared but limited version on the server and a detailed version somewhere locally – and therefore undermines the concept of single point of consistency checking – or it leads to a limited usage mode of the software tool at hand avoiding the creation of the necessary extra data to store and thereby limits the ability to find the best possible solution.

In this paper an approach is outlined which fully accepts the existing heterogeneous software landscape. Having BIM and other models of the desired real-world object at hand this question arises:

“How can many different data sources efficiently be included in the process of computationally solving a specific task?”

This paper presents a solution adopted from the field of computationally aided software engineering (Bézivin 2001). It will be shown how a system of formal model definitions can be applied to integrate different data sets in a construction project.

One focus of research in the last decade lied on the semantic meaning of a model and its parts. This lead to the development of ways to describe the semantics found in models, called ontologies. Frameworks were developed to store ontologies and tools to process and to interrelate different ontologies.

Staub-French et al. (2001) utilize an ontology to relate construction features with activities to calculate costs. Nepal et al. (2009) use ontologies to analyze design conditions found in IFC models in order to support management functions.

Steel and Drogemuller (2009) describe their experience with model interoperability in BIM using the IFC model.

A meta-model based approach is reported by (Beetz et al. 2010) describing the implementation of an open source IFC-based BIM server. They utilize EMOF technology to efficiently implement server-functionality to the IFC-data model.

The next section introduces a common formal modeling technique found in software engineering. The standards and an existing framework are introduced. Section 3 focuses on detailed procedures in setting up a multi-model, while section 4 gives insights into an example application. The paper closes with conclusion and outlook.

⁴ http://www.buildingsmart.com/
2 Formal Modeling Techniques

By strictly expressing the data model structure of existing software solutions in a standardized way a model-mapping solution can be written. The model structure is expressed as a meta-model defined in the Meta Object Facility (MOF) issued by the Object Management Group (OMG 2010). In particular the Ecore technology of the Eclipse Modeling Framework (EMF)\(^5\) by the Eclipse Foundation will be used. This enables the use of a rich set of software tools conforming to the Query/View/Transform (QVT) standard by OMG (OMG QVT 2008). These tools allow the definition of uni- and bidirectional mappings on the meta-model-level. These mappings can be automatically applied to instances of the meta-models resulting in a model-instance network storing the relevant data for the task to solve.

Subsequently the users can solve the task by providing additional input and executing algorithms on this data-network. Depending on the task, the output can be stored in a separate model or the relevant pieces of data are written back to the originating data sources.

2.1 Meta-Modeling Standards

The Meta Object Facility was originally developed by the Object Management Group (OMG) to express the Unified Modeling Language (UML) in a formal system. UML expressed in MOF results in a layered architecture with four abstraction layers M3 to M0 (Figure 2). By modeling with elements of the M3-layer it is possible to describe the structure of UML as an M2-layer model. Software components such as classes and packages are modeled as UML-models and hence are instances of the M1-layer. Instances of these classes are called objects and are part of the M0 layer as defined in MOF. The M3-layer is self-contained in a way that it can be expressed by the elements defined in it.

![Figure 2: Layers of abstraction](image)

Accompanying with the standardization of MOF several tools were standardized which streamline the process of software-development. The XML Metadata Interchange (XMI) (OMG XMI 2007) is an XML-based interchange format for MOF models and model-instances. Because any MOF-based model can be transmitted via an XMI-instance this standard gives a basic foundation for information exchange and provides a persistent storage mechanism. The MOF Query/View/Transform is a technology neutral specification of “1.) queries on models; 2.) views on metamodels; and 3.) transformations of models” (OMG QVT

\(^5\) http://www.eclipse.org/modeling/emf/
The transformation methods specified in QVT standardize some powerful concepts of (meta-) model mappings. The Object Constraint Language (OCL) (OMG OCL 2010) accompanies the standards above by providing guard-/invariant- and attribute derivation-expressions.

The above mentioned standards altogether form the basis for the Model Driven Architecture (MDA) approach in the software industries. They are applied very successfully in embedded software development where a streamline of software products is produced out of software components with different modifications and configurations for each single product. This application resulted in a toolset now available as open-source implementations.

2.2 Meta-Modeling Frameworks

The above described standards are all technology neutral. To use the standardized modeling techniques a concrete framework has to be deployed. A very mature framework – EMF – was developed by the Eclipse Project (Steinberg et al. 2008). This project provides the modeling capabilities as well as sophisticated code-generation mechanisms. All models can be compiled into fast and type-safe Java implementations. The ecosystem nature of the Eclipse projects sometime leads to different implementations of the same standard while it fails to provide an implementation at all for others. The MOF standard comprises two parts: Essential MOF (EMOF) and Complete MOF (CMOF). Over the last years the development of the Eclipse Modeling Framework was aligned with the development of the MOF standard (Steinberg et al. 2008). As a consequence the Eclipse Ecore package provides a full implementation of the EMOF standard. Full XMI support by EMF enables the creation of true multi-models. The capabilities found in the XMI-Standard allow for cross-referencing object instances over XML-files. EMF transparently resolves model-elements in different files so that a creation of an interconnected model does not increase the complexity of the implementation.

Model-mapping according to the QVT operational mapping language is provided by the Model-To-Model (M2M) subproject.

3. Building a comprehensive Meta-Model

In order to beneficially apply the existing modeling tools the data structure of our software solution has to be divided into reasonably sized models.

An analysis of required information will be undertaken. The aim is to create an interacting-model-system which is well balanced between the information-structure already found and the structure required to efficiently solve the problem. In the context of model-driven software development the model-parts of the desired data-model representing input-information will be identified.

A listing of all participating existing software solutions has to be set up along with a compilation of the information they provide. This input-information is transferred into the desired data-model by building an initial model.

To unify data-processing and to clearly separate data-fusion from collecting available data these two concerns are separated. For each participating existing software solution the following is carried out to obtain the data-extraction mechanism:
1. **Analysis:** Gathering of the relevant information types found in each single native data source and the relationships found in between.

2. **Modeling:** Transition of the information and relationships into an Ecore-backed model for the information collected from step 1. These models are called raw-models.

3. **Coding:** Writing a mechanism (plug-in, loading-algorithm ...) capable of creating a model-instance of the model defined in step 2 and transferring all data from a running application into this model-instance.

Realizing all of the above steps results in a collection of raw-models. This data is then merged into a coherent data model by applying a series of model-to-model transformations. The transformations result in intermediate models. Finally, a last model-to-model transformation transfers the data into the model-parts of the desired data-model.

### 3.1 From an existing software component to a model

The central idea of the presented data integration approach is to extract only relevant information out of the existing software and data schemes.

Two different approaches can be considered. Firstly, access to the data can be given through the persistent data storage. If the data format is known, the relevant data can be extracted from the persistent storage. This works very well with applications utilizing relational data bases, XML-based storage formats or otherwise standardized and well structured data. For instance model-based access to IFC-P21 files can be generated from the Express definition of the IFC standard (Beetz et al. 2010).

Secondly, data in software products which provide an open API for software developing can be accessed directly.

Figure 3a depicts the data extraction modeling resulting in raw-models.
3.2 Steps towards a multi-model

Exporting data results in instances of raw data models associated with the existing software applications or storage files. These model-instances will be automatically transformed into a set of different data models instances using QVT-Operational mappings. The transformation is carried out step by step with each mapping focusing on a major aspect. Each QVT-Mapping can have multiple models as input and as output. This enables the establishing of connections between different model-instances. One goal of a mapping is to achieve a reorganization of data. This can greatly simplify subsequent mappings. If a real-world object is referred to from different existing software components the above workflow introduces a redundant model-instance for the same object, one in each raw model. Another goal of model mapping is to let the different sub-models reference only the same instance. This step connects the sub-models involved as shown in Figure 3b by “reference links”.

The QVT-mappings as well as the data-models should be authored in way that a clear separation of concerns can be maintained. It is expected that the integration of ontology-based methods will beneficially improve the authoring and the performance of the mappings.

Once the mappings are provided the model-transformation can be applied repeatedly and automatically on model instances.

The result is a data-model comprising different interconnected sub-models (Figure 4). All available task-relevant data is stored in this set of Ecore model instances. It serves as the main data structure for the upcoming application development.

![Figure 4: Generated network of models. Instances of such a model network hold all relevant application data.](image)

4 Example Accessing BIM

To solve a specific problem in the context of task sequencing of construction works an application has been developed. For this example existing software models comprise

- a BIM model provided as Revit Project,
- a relational database of production techniques (mainly consisting of process-templates),
- calendar data given as milestones in XML and
The goal of the desired software is to assign each component to the intended sub-process-chain, to incorporate additional user-provided interdependencies and finally to schedule all processes in such a way that every technological interdependency is satisfied (Huhnt, 2009).

The whole system was implemented as an Eclipse Rich Client Platform, written in the Java programming language. The developed application utilizes a workbench concept. This enables a simple change management of the involved data-sources by monitoring and visualizing their current states.

The concept of processing all necessary data as instances of specifically tailored Ecore-models was applied. For the purpose of data extraction of BIM data a Revit-plugin was authored which was linked to the main application via web service interfaces. Available XML data was mapped to Ecore models via schema based model generation. In this relatively small example relational database access and the corresponding model was directly authored. For more complex layouts mapping solutions like the Teneo project would have been utilized.

After getting access to data by creating the raw models in a second step a QVT mapping was written to combine the information of the given structuring tree and the components provided by the BIM model. This step incorporated existing building-level membership of components as well as evaluation of project-specific string properties authored by BIM users with the independently provided hierarchy elements. This step reduces the amount of data stored with

6 http://www.eclipse.org/modeling/emft/?project=teneo
the component entities and replaces references to level objects with hierarchy elements from the structuring tree.

Authoring model transformations has to be done thoroughly to ensure all relevant elements still remain identifiable. In this particular setup the provision of universal identifiers out of BIM has eliminated this potential pitfall.

The resulting data structure was used together with user input to carry out the process instantiation and scheduling of tasks. The generated objects have been stored in an additional sub-model linked to the generated structure. For the purpose of data-exchange the overall scheduling result was then exported to a Microsoft Project file. Each task was provided with its associated component ID. This enabled an automatic generation of a 4D animation of the construction work inside Autodesk’s Navisworks application taking the original BIM and the created schedule.

5 Conclusion and Outlook

It has been shown how to efficiently combine different existing software solutions to author a new solution which usually would bind a lot of software development resources.

With the proposed method above a comprehensive model interconnecting different sub-models was constructed. The sub-models were specified using MOF. This foundation enabled the usage of a uniform set of software model tools. All data could be accessed by the same means regardless of the original software product they where authored in. In a next step this comprehensive model can be used to apply special algorithms solving the desired task.

The presented solution lead to flexible software architecture to incorporate many differently organized data sources. The usage of the modeling approach enabled the deployment of available persistent storage mechanisms at no development cost. The access to existing project data was – up to the initial software start up and plug-in invocation – fully automated.

The clean model-based approach eases data-processing. The integration of the results and methods provided by ontological research activities are considered as one of the next steps towards a more automated engineering.

References


Bézivin, J. (2001): „From Object Composition to Model Transformation with the MDA“. Published in the proceedings of TOOLS'USA, Volume IEEE TOOLS-39, Santa Barbara, August 2001


