Model Driven Engineering Methods for Integrated Building Performance Optimization

Sebastian Stratbücker¹, Sicheng Zhu¹, Matthias Mitterhofer¹
¹Fraunhofer Institute for Building Physics IBP, Valley, Germany

Abstract
Tool interoperability is a vital technical requirement for any integrated building performance assessment in early building design stages. Architects and planners use tools with proprietary geometric representations and domain specific computational intelligence. On the other side, there is a multitude of sophisticated tools for simulating and assessing building performance. For obtaining reliable results, such tools need to be operated by qualified personnel only. In order to bridge the gap between expert domains, this study proposes the deployment of a central building information model (BIM) on top of a conventional database system. We present novel methods and a comprehensive tool chain that supports a multidisciplinary design optimization process. A new way of addressing the challenges of poor software interoperability is to apply existing solutions from model driven engineering. The main focus of the research is on the formulation of model-to-model transformations that can be reused for BIM based tool coupling in general.

Background
Integrated building performance assessment in early design stages requires a high level of software interoperability (Rahmani Asl et al. 2015). Architects and planners use computer aided design (CAD) tools with geometric representations and operations, where engineers and researchers use building performance simulation (BPS) tools with specific domain knowledge and computational complexity. Emerging technologies for building information modeling (BIM) are deemed to be key enablers for tool coupling in a collaborative building design process (Singh et al. 2011). However, there is no broadly adopted method, which is coherently applying BIM as basis for BPS tools in order to support multidisciplinary design optimization (MDO) (Sanguinetti et al. 2012; Clarke and Hensen 2015; Hyun et al. 2015; Negendahl 2015).

In terms of building energy performance simulation (BEPS) Bazjanac (2008) introduced a software for transformation of BIM data to generate IDF input files for EnergyPlus. The transformation rule types include data set reduction and simplification, data translation and interpretation, in which the latter are considered the most challenging ones. The rule set for semi-automatic generation of input data for BEPS is embedded in the software code of the Space Boundary Tool (SBT). According to Gourlis and Kovacic (2016) progress has been made in data transfer automation, though this requires custom software development or specific design guidelines for modeling according to the BEPS demands. In practice, most building energy models are reworked by domain experts with the risk of arbitrary definitions depending on personal skills and understanding. The current state of industrial implementations of BIM interfaces for BEPS is summarized in Nasyrov et al. (2014). Except for geometry, the level and quality of transferable data still need major improvements.

The authors in Cemesova et al. (2013) describe the development of a converter and data mapper for a passive house planning tool, after extension of the IFC schema with energy related data that is then extracted by Java application. In Cao et al. (2015) implementation language features from a C++ class library are used for accessing meta-information at runtime to map model elements from SimModel (ODonnell et al. 2011) to Modelica (Modelica Association 2014). Recently the authoring tool Autodesk Revit and its open and flexible .NET application programming interface (API) was used in many cases as BIM platform to manage data for performance based building design (Jeong et al. 2014; Kota et al. 2014; Jalaei et al. 2015; Kim et al. 2015; Rahmani Asl et al. 2015).

Augenbroe (2011) states that the currently implemented BIM interfaces do not support the main requirements from performance based design optimization. Instead one can observe that elements are haphazardly collected to be processed and mapped in proprietary software interfaces.

There are various databases relating to material properties, weather data, and modeling entities, which are all represented in proprietary formats and that are not kept in a central repository. Further there exist plenty of specialist tools for almost every building related domain such as acoustics, IAQ, life cycle assessment (LCA) and other, but the integration of BPS tools is progressing rather slow. Based on that findings Clarke and Hensen (2015) conclude that the absence of an overarching data model could be the principal barrier.

Though there are plenty of examples for BIM and BPS integration based on proprietary but also standardized software technologies, still there is no systematic
principle for a model based transformation between several tools, domain specific data representations and disciplines involved in the AEC industry. To entirely support model driven performance based design and optimization a verified, formalized and rule-based method for complex transformation tasks should be established. Other disciplines like software engineering address interoperability issues alike. There the notion of model transformation languages is widely used for automated code generation, model refinement strategies or sophisticated translation problems.

**Method**

By applying well-established methods and tools from model driven engineering (MDE) (Mohagheghi et al. 2013; Object Management Group 2014), a prototype framework for multi-disciplinary building information modeling and performance assessment is presented in this work. According to Mohagheghi et al. (2013) the benefits of implementing MDE include:

- helpful abstraction of complex systems at multiple levels and from different aspects
- improvement of interdisciplinary collaboration and communication by domain-specific models
- support of simulation and testing
- performance-related decision making
- system design improvements
- possibility to merge different tools in a single development environment

The main concept for MDE is to strictly use models throughout the whole design and development process, starting from a high level of abstraction. Models, metamodels, and metametamodel form a three-level stack, with levels named M\(_1\), M\(_2\), and M\(_3\) respectively. Modeling languages consist of an abstract syntax (i.e. metamodel), at least one concrete syntax and semantics. A metamodeling language is able to describe structural aspects of models on lower levels by concepts like classes and references.

Entities on level M\(_0\) are real objects that are relevant to the application domain. Classes and objects that provide abstraction to these entities are modeled on level M\(_1\). The next higher modeling level is M\(_2\), which is the metamodel that describes how to formalize M\(_1\) entities and relations.

The Meta-Object Facility (MOF) serves as self-describing root metamodel (see Figure 1). So called model-to-model (M2M) transformations are used in this study to describe logical operations on level M\(_1\), while conforming to the corresponding metamodel on level M\(_2\). The Eclipse Modeling Framework (EMF) fully supports the concepts of MDE and implements its own Essential MOF (EMOF) named Ecore (the Eclipse Foundation 2016). The metamodel stack in Figure 1 depicts languages and models that are used in the present work.

**Data Requirements Modeling**

In a first step, the domain specific data requirements from a set of BPS tools are analyzed and elaborated using the Unified Modeling Language (UML) (Object Management Group 2014). As described by Sanguinetti et al. (2012) the next step is to decide upon a target design model that serves as basis supporting a broad range of information modeling requirements, while reflecting the individual nature of construction projects. We introduce the term Unified Data Model (UDM) as being the conceptual domain model on level M\(_3\), which is deployed as centralized BIM data-service. BPS tools on the other hand are developed and deployed with Specific Data Models (SDM). The main challenge is to formulate rules and operations that safely convert data from UDM to SDM without information loss, need for manual modifications or time-consuming inspections.

![Figure 1: Modeling stack used for data analysis and prototype framework](image1)

![Figure 2: Subset of the Ecore model entities and relations (Budinsky et al. 2004)](image2)
Model Transformation Language

The ATLAS Transformation Language (ATL) is a model transformation language and toolkit, which is used to transform a set of source models into a set of target models. Developed in the Eclipse platform, ATL conforms to the Ecore metamodel and executes transformations between models on the levels M₁ and M₂. The procedure of a model transformation consists of two parts. First, the source and target models are generated with help of Ecore. Rules for transforming models on the same level are expressed using ATL. The generated model transformation logic conforms to a translation language on level M₂ and is based on the source and target metamodels on M₁. All metamodels conform to a common meta-object facility. In the case of ATL the MOF is represented by Ecore.

![Figure 3: Model transformation pattern](image)

ATL is an open source tool based on Java and provides complete support of horizontal transformations between models of the same abstraction level, as well as vertical transformations that include refinement of model data. ATL features exogenous, i.e. heterogeneous transformations between source and target models that are expressed in different metamodels. A full comparison of model transformation tools can be found in Kahani and Cordy (2015). Further properties of ATL are:

- Object Constraint Language (OCL 2.0) for queries and navigation in source models
- semi-automatic unidirectional transformations
- support of all cardinalities
- implicit and explicit formulation of rules (while selection of rules is non deterministic)
- recursion based iterations
- modularity and inheritance based reuse of code
- support of user-defined traceability between models (based on explicit tracing links)

Since ATL is a hybrid transformation language, it supports imperative as well as declarative constructs. According to Jouault et al. (2008) the latter gives advantages for model developers, since the methods of specifying relations between source and target patterns is more intuitive. The declarative expressions are encoding the inter-model relations using a simple syntax, while the transformation system is in charge of handling underlying complexity (e.g. related to selection of source elements, rule triggering and ordering, dealing with traceability, etc). Two types of transformation rules are defined in ATL. On one hand, matched rule can transform corresponding classes or elements from source model to target model. On the other hand, the called rule can generate target model elements from imperative code. That means the elements that do not exist in source model can be instantiated during the procedure of transformation. Besides ATL there exist plenty of modeling and transformation tools (e.g. based on XSLT or QVT). ATL is used consistently in the present study to demonstrate interoperability of BIM and BPS tools by means of model-driven data integration.

BIM Database System

A relational database management system (RDBMS) serves as backend to provide BIM data according to the defined UDM. The UDM is defined in compliance to data dictionaries and classifications systems that are already part of the planning process for architects (e.g. German DIN 276). There are many reasons for deploying a central BIM repository with DBMS. The DBMS maintains building data and allows native applications and multiple user groups to import and export data for viewing, checking and modification. The BIM-server itself has limited built-in functionality and is mainly used to exchange information between stakeholders involved throughout the building project life cycle. It supports project based collaboration, synchronization, versioning and multi-user applications. Data can be stored and managed consistently. The notion of the application specific model view is incorporated in most DBMS. The system already provides means to deploy and manage role based data access and modification rights. Depending on the tasks, responsibilities and project phases, the roles of users define the applicability of filtering functions and grant read/write access on the level of single properties. Further details about DBMS structure and functionality are presented in Zhu (2016).

Results

The presented prototype system strives for the establishment of an adaptable UDM that is able to reproduce the semantic relationships between data objects in order to comply with the information requirements of almost arbitrary BPS tools. The UDM
defines classes and attributes aligned to existing BIM data dictionaries having a Global Unique Identifier (GUID), so that they can be shared and exchanged seamlessly between different applications (Törmä 2013). Besides addressing the need for interoperable BPS tools by means of a DBMS, we also include evaluation tools, native support for computer aided design (CAD) tools like SketchUp (Trimble Navigation Limited) and linking to semantically rich building physics data sources (see Figure 4).

**Figure 4: Prototype system for integrated building performance assessment**

Designers that use a central project-based BIM must be able to modify how properties are assigned to groupings of objects. This requires a flexible problem formulation methodology for BIM-based MDO that enables a dynamic exchange requirement structure (Welle et al. 2011). The class structure of the proposed RDBMS follows these design objectives to provide most possible flexibility and extensibility for application domain modeling (see Figure 5).

**Figure 5: Structural class model of BIM database**

The main class is plain Object that appears as sub-types Group and Entity. A Group is used to collect Object instances. The Entity represents a BIM element, which is usually imported into the database from an external CAD tool with its native Geometry representation. PropertySetContainer is a class that relates to the classification of an instance of Object. In this way, the assignment of properties must not be resolved at the Object level. Instead, all Object properties are defined by its classification, which conforms to the way planners would treat building elements in the first place. A so called Object Modeler is used to define project specific classification systems and associated property specifications. The Base Model provides a set of default elements and attributes. The Base Model is usually complemented by a Custom Model, adhering to the requirements of one particular building project. When deployed to the RDBMS each class of Base Model and the Custom Model is represented by a separate table. Since the UDM is fully supporting model driven design and development, any user defined modification or extension of the Custom Model is possible during system runtime.

For the study, three BPS tools are taken as instances, WUFI® Plus (Fraunhofer Institute for Building Physics IBP) for hygrothermal simulation and annual energy consumption calculation, a Modelica-based zonal thermodynamic simulation model VEPZO (Norrefeldt 2013; Reim et al. 2015) for indoor climate assessment and Daysim (Reinhart 2016) for daylight analysis. The main focus of this research is on the formulation of M2M procedures to be able to adjust, match, remove or enrich model structures such that BPS tools can extract required input data directly from the UDM.

**Unified Data Model**

We use EMF to model the input and output data requirements from various BPS tools. Based on the Ecore MOF the data structure of the UDM is formalized (see Figure 6). The main activities to define the UDM are:

1. Analyze basic features of the DBMS and the input data of selected BPS tools w.r.t data structure, required attributes, type of identification literals
2. Define new classes using Base Model of DBMS to adapt to the data structures of different SDM
3. Represent required attributes within UDM for robust mapping into different SDM

**Model to Model Transformation**

To demonstrate an ATL transformation process we present an example that uses UDM as source model and a specific target model used by WUFI® Plus (named SDMW and partly shown in Figure 7). The main elements in a model transformation are object classes and their identity, the relationship between objects and the attributes of each object. The GUIDs from UDM are kept in the transformation, whereas a new ID as integer value for SDMW is also created to meet the requirements of the input data. In this procedure, a combination of a certain GUID and an ID as integer are linked in order to identify the same objects.
in different models. The SDM represents all class associations, by resolving references with attributes. The structure then conforms to the input data requirements for WUFI® Plus, which is an efficient data serialization without redundancy.

The transformation rule removes all objects of class Room because it is not required for that particular BPS-tool. Another difference is found in the representation of attributes that illustrate the semantic relationship between building components and zones (i.e. inner vs. outer side of a component). The corresponding ATL code for the rule is presented in Listing 1. The helper acts like a function to define the attributes, which can be used in the following transformation rules. The rule illustrates the translation regulations from one class in original model and belonged attributes to one or multiple classes in target model.

According to the ATL rule set in Listing 1, there are two exemplary instances for source and target data given in Figure 8: sample data from UDM and the same data in SDMW after execution of the ATL code. The source and target data is expressed in XMI format. Using XSLT or XPath the target data can be finally converted to the XML input data format for the BPS tool WUFI® Plus.

---to translate classification with integer in UDM into types of building components in SDMW
helper context UDM!Component def: Typ: String =
if self.Classification1 = 10 then 'opaque'
else 'transparent' endif;

--to define the inner side and outer side of each component
helper context UDM!Component def: Outside1: String
= if self.Classification2 = 10 then 'Outer air'
else if self.Classification2 = 20 then 'Ground'
else 'Zone ' + self.Outside endif endif;

helper def: e: Integer = 0;

helper def: allComponents: Sequence(UDM!Component) =
UDM!Component.allInstances()->asSequence();

helper def: allZones: Sequence(UDM!Zone) =
UDM!Zone.allInstances()->asSequence();

-- to transform classes of type Building
rule Building { from s: UDM!Building to t: SDMW!Building (Name<-'SDMW'+s.Name,
Zone<-'thisModule'.allZones,
Component<-'thisModule'.allComponents)}

-- to transform classes of type Zone
rule Zones { from s: UDM!Zone
to t: SDMW!Zone (ID<-s.Position1)}

-- to transform classes of type Component
rule Components { from s: UDM!Component
to t: SDMW!Component (Type<-s.Typ,
Innerside<-'Zone ' + s.belongedRoom.belongedZone.Position1,
Outerside<-s.Outside1)
do { thisModule.e<-thisModule.e+1;
t.ID<-thisModule.e}" ] } }

Listing 1: Part of the ATL transformation rules to handle source classes Building, Zone and Component
Model Transformation Patterns

Several patterns of model transformation in the field of software engineering are presented in Lano (2007) and Krämmer (2013). Due to basic features of languages like ATL, modular and inheritance based reuse of code would enable a scaling up of once defined transformation rules for bridging the gap between UDM and various other BPS tools. During the study five generic patterns for M2M transformations were identified:

1. Removing an obsolete class
   Example: removing the class Room (Figure 7)

2. Creating a new class
   Example: Creating the class Material instead of saving the material properties in class Layer

3. Redefinition of an association relationship to a reference using class attribute
   Example: from inheritance to references

4. Transformation of identification literals
   Example: From references using GUID strings to references using Integer values or Name strings

5. Resolve redundant model data
   Example: The same constructions in UDM are transformed to a construction library, where the particular constructions are referenced by the components

Model Based Performance Assessment Workflow

According to Attia et al. (2012) the challenge in many BPS tools is the representation of input parameters, because of the clear separation between the architectural requirements and the language for describing building physics. A proposal to define a common performance-driven architectural design workflow is presented in Shi and Yang (2013). According to the authors the workflow must be supported by an underlying data exchange and communication system to control the entire design and analysis process. The following iterative steps within the workflow are proposed:

- Use a modeling program to generate one or multiple initial designs and parameterize them
- Define geometric, material, physical, and environmental parameters to initialize BPS tool
- Analyse the performance according to the simulation output
- Check if the predefined design objectives are met
- If needed, adjust the design and repeat the procedure
A continuous workflow from UDM to SDM using formal M2M transformations is proposed in this study to generate input for several BPS-tools and deliver results for integrated performance assessment (Figure 9).

The current status of model and software implementation helps understanding the data requirements for future decision support systems (DSS) (Jalaei et al. 2015). According to Augenbroe (2011) the performance of a building can be defined as a quantifiable expression of how well a particular building function is achieved. A derived need can be expressed as a criterion of achieving a function, thus the criterion is an expression of performance. One problem is that every part of a building will contribute to multiple functions. Performance cannot be measured unless the systems are in a particular use condition, in that terms performance always relates to a specific behaviour of the whole system in its environment, and cannot be merely defined as a property of the building or its components. So called key performance indicators (KPI) are individual target parameters that are defined in close coordination with the client. KPIs are aggregated from BPS output data, while the metrics for performance quantification are determined in advance. Based on formal definitions for KPI, a valuable DSS would provide feedback for building designers based on a cross-domain evaluation of the BPS results.

The KPI could be the energy consumption, the criteria for thermal comfort, daylight factor among others. Performance needs to be determined either by stakeholders or by adapting criterion from sustainable building certificate systems like LEED (The U.S. Green Building Council 2016) and DGNB (German Sustainable Building Council 2016). To support the continuous and iterative building performance simulation, assessment and optimization process the underlying database model is augmented with performance related classes (see Figure 10).

As the model structure reflects all relations between UDM and performance criteria by introducing functional hierarchies, the M2M workflow is able to generate information for defining proper use conditions and simulation parameters for BPS tools, but also for post-processing output data to aggregate KPI values. That approach is considered being a major technical enabler for holistic assessment of building performance. All performance criteria providing proper KPI definitions can be automatically processed and evaluated. The effects of parametric changes in the UDM, and their impact to the user defined KPIs can be analyzed and communicated for further decision making process.

An example to illustrate the model-driven optimization process would be given by the Performance Criterion of operative temperatures, which is an important criterion to evaluate the Building Function providing thermal comfort. The calculated results from WUFI® Plus or other energy simulation tools including hourly outdoor temperature, outdoor relative humidity and indoor temperature are vital raw results to define the operative temperatures. There exist several criteria or Aggregation Methods to evaluate the limit of operative temperature, for example the limit of 26 °C for summer and 20°C for winter or different adaptive models (Park and Stratbücker 2013). With such aggregation methods, the corresponding Performance Indicator like degree hours or the length of deviation as a single indicator rather than operative temperatures for each hour during a period can also be calculated and used directly to evaluate the indoor thermal comfort.
pattern of usage from standards like DIN V 18599-10 or statistical models. All the above mentioned objects and information can be defined previously and combined with each other in several models to support the design process.

Discussion

As already stated in literature MDE is costly and requires higher expertise than ad-hoc software development with object-orientation (e.g. Java, C++) and powerful scripting languages (e.g. Python, Ruby). Rationales for using MDE for integrated building performance assessment are discussed in the following sections.

The metamodeling and code generation capabilities of the EMF are addressing fundamental input and output data requirements for coupling BPS-tools with a BIM data-server. Using ATL is an efficient means for any kind of transformation of complex data structures using both declarative and imperative language concepts. ATL enables model weaving and supports the linked-data mechanism to share information between different domains and disciplines.

The advantages of using a centralized BIM data-service based on UDM include:

- Efficient and robust data management system based on element-wise classification and allocation
- Adaptive and flexible data enrichment by user defined property sets
- Assignment of data elements to predefined Level of Detail (LOD)
- Support of interdisciplinary co-operation and multi-model management

The RDBMS also supports linking to data sets from third-party databases, e.g. for building materials. Without referring to certified single sources of information regarding the physical, technical and functional specifications of building components, a consistent and valuable provision of input data for BPS tools will depend on personal and organizational capabilities (i.e. tedious quality checks, manual data input, error-prone copy and paste solutions).

Conclusion

The analyses in the field of building physics, such as hygrothermal, indoor climate and daylight are most valuable in early stages. We explore the capabilities of a model driven integration of CAD tools, BPS tools, and a BIM data server for building performance assessment. A prototype framework for this integration demonstrates how MDE methods improve the quality and coherence of BIM based cross-domain applications by strictly using formal semantic metamodels, model transformation languages and appropriate modeling frameworks. If the proposed MDE methods were scaled up to a wide range of AEC applications, this would provide a stable technical solution for iterative and integrated performance based building optimization.

The methods adopted from MDE are able to produce an integrated set of tools and processes. In combination with optimization routines, they are the key to evaluate a multitude of design variants. Using MDE methods and technologies BPS tools could be used to full capacity, in contrast to occasional and detached applications of building simulation tools as observed in practice. Since the established system is extensible and expresses a generic data source for building design and analysis, more BPS tools and other software modules can be integrated on top of the UDM in the future.

Acknowledgement

This study has been conducted with funding from the German Federal Ministry for Economic Affairs and Energy (BMWi) under support code 03ET1177B. The authors are responsible for the content of this publication.

References


In Irvine, CA, USA, September 16-18 (pp. 412–419).
doi:10.1109/ICSC.2013.80