PRÜFUNG VON OPENBIM GEBÄUDEMODELLEN FÜR DIE THERMISCHE GEBÄUDESIMULATION

A. Geiger[1] und V. Hagenmeyer[1]

[1] Institut für Automatisierungstechnik und Angewandte Informatik, Karlsruher Institut für Technologie, Karlsruhe, Deutschland

KURZFASSUNG

Der Einsatz von Softwarewerkzeugen für die thermische Simulation von Gebäuden spielt eine wesentliche Rolle bei der Einhaltung gesetzlicher Vorgaben für deren Energieeffizienz. Mit dem Datenmodell Industry Foundation Classes (IFC) stellt buildingSMART ein allgemeines Building Information Model (BIM) bereit, das alle Voraussetzungen zur Unterstützung von thermischen Gebäudesimulationen erfüllt. Um in der Praxis einen verlässlichen Austausch zu gewährleisten, müssen bestimmte Anforderungen - sogenannte Model View Definitions (MVD) - definiert und eingehalten werden. Die Formalisierung dieser Anforderungen erfolgt mit dem von buildingSMART entwickelten Format mvdXML. Der vorliegende Beitrag beschreibt die prototypische Entwicklung spezifischer MVD Regeln, die technische Umsetzung dieser Regeln in der Anwendung IFCExplorer und zeigt deren Einsatz an einem realen Gebäudemodell.

ABSTRACT

The use of software tools for thermal simulation of buildings is an essential part of compliance with legal requirements for their energy efficiency. With the Industry Foundation Classes (IFC), buildingSMART provides a general Building Information Model (BIM) that satisfies all requirements for supporting thermal building simulations. In order to ensure a reliable exchange of information in practice, specific requirements - so-called Model View Definitions (MVD) - must be defined and observed. These requirements are formalized in the mvdXML format also developed by buildingSMART. The present paper describes the prototypical development of MVD rule sets, their implementation in the application IFCExplorer and demonstrates their usage by a real building model.

EINLEITUNG

Die gesetzlichen Anforderungen zur Energieeffizienz von Gebäuden, sowohl beim Neubau wie auch bei umfangreichen Sanierungsmaßnahmen, sind in den letzten Jahren deutlich gestiegen. Daher spielt der Einsatz von Softwarewerkzeugen zur thermischen Simulation von Gebäuden eine wichtige Rolle bei der Einhaltung dieser Vorgaben. In der Baupraxis setzt sich zunehmend die BIM-Methode (Building Information Modelling) für einen digitalisierten Bauprozess durch (Borrmann et al., 2015). Die internationale Organisation buildingSMART entwickelt unter dem Leitwort openBIM dafür offene Standards und Spezifikationen. Die zentrale Rolle spielt dabei das Datenmodell IFC (Industry Foundation Classes) für Bauwerke, das seit der Version IFC4 ein offizieller ISO Standard (ISO 16739:2013) ist. Um spezielle Anforderungen eines Datenaustauschprozesses zu spezifizieren wurde darüber hinaus das Information Delivery Manual (IDM) entwickelt, das ebenfalls als ISO Norm (ISO 29481-1:2010) verfügbar ist. Das IDM beschreibt grundlegend den Umfang und die Detailtiefe der Informationen. Hiermit lassen sich die Prozessverantwortung, die Rollen der Beteiligten und die technischen Bedingungen im Rahmen der Prozessplanung definieren. Es handelt sich um standardisierte Template Dokumente, die in

einem weiteren Schritt mit konkreten Klassen, Attributen und Eigenschaften aus dem IFC Datenmodell ergänzt werden. Diese konkreten Formulierungen werden als Modellansichten oder Model View Definitions (MVD) bezeichnet (Hietanen, 2006). Sie stellen eine Teilmenge des IFC Datenmodells dar, das für einen bestimmten Anwendungsbereich - wie beispielsweise die thermische Gebäudesimulation - notwendig ist und von Softwareprodukten unterstützt werden muss.

Seit der Version IFC4 entwickelt die buildingSMART darüber hinaus das Datenformat mvdXML (Liebich et al., 2011). Es wurde als neutrales Format zur vollständigen formalen Beschreibung von MVDs entwickelt, und unterstützt seit der Version mvdXML 1.1 (Liebich et al., 2014, Chipman et al., 2016) auch eine Regelsyntax. Damit erlaubt das Format mvdXML die Definition und Beschreibung von Regeln, um die Datenqualität im Austauschprozess zu gewährleisten. Ausgehend von der Teilmenge des IFC Datenmodells, das durch eine MVD beschrieben wird, lassen sich die verwendeten Klassen und Attribute festlegen oder einschränken, Eigenschaftssets definieren, und Gültigkeitsbereiche von Attributen festlegen. Es existieren bereits einige Softwarewerkzeuge mit Unterstützung des mvdXML Datenformates. In diesem Zusammenhang ist insbesondere das von der buildingSMART bereitgestellte Werkzeug ifcDoc (buildingSMART, 2012) zu erwähnen. Es ist das zentrale Werkzeug zur Entwicklung von MVDs, der Definition von Prüfregeln, sowie zur Prüfung der Regeln gegen IFC Instanz Dokumente. Zusätzlich wird mvdXML im Rahmen der IFC4 Zertifizierung von buildingSMART eingesetzt. Dabei werden die Anforderungen der einzelnen Test Cases auf Basis von mvdXML definiert und mit Hilfe des IFC Framework der Firma apstex validiert (buildingSMART, 2018). Weiterhin existieren aktuell zwei IFC Analysewerkzeuge mit mvdXML Unterstützung. Dies ist zum einen das von der Northumbria Universität, Newcastle als Open-Source bereitgestellte Werkzeug Xbim Xplorer (Lockley et al., 2017, Weise et al., 2016), sowie Simplebim (Hietanen, 2014). Beide Werkzeuge stellen ein Plug-In für die mvdXML basierte Regelprüfung bereit. Im akademischen Umfeld entstand zudem im Rahmen der Open BIM Plattform Services, der TU-Dresden ein Modul zur Konvertierung von mvdXML Dateien in die selbst entwickelte Abfragesprache ifcQL (Baumgärtel, 2016) und für das Open BIMserver Projekt wurde die prototypische Implementierung eines Regelprüfungsmoduls realisiert (Zhang, 2014).

Im Rahmen des vorliegenden Beitrags werden die Möglichkeiten und der Umfang der von mvdXML bereitgestellten Regelsyntax für den Anwendungsfall der thermischen Gebäudesimulation analysiert. Dazu werden geeignete Prüfregeln definiert, um zu gewährleisten, dass der Informationsgehalt eines IFC Datensatzes die minimalen Anforderungen an eine Gebäudesimulation erfüllt.

MVDXML METHODIK UND GRUNDLAGEN

Das Datenformat mvdXML bildet die Anforderungen und Definitionen einer MVD ab. Abbildung 1 stellt stark vereinfacht die Struktur und die Beziehungen der zentralen Elemente von mvdXML dar, die im weiteren näher erläutert werden.

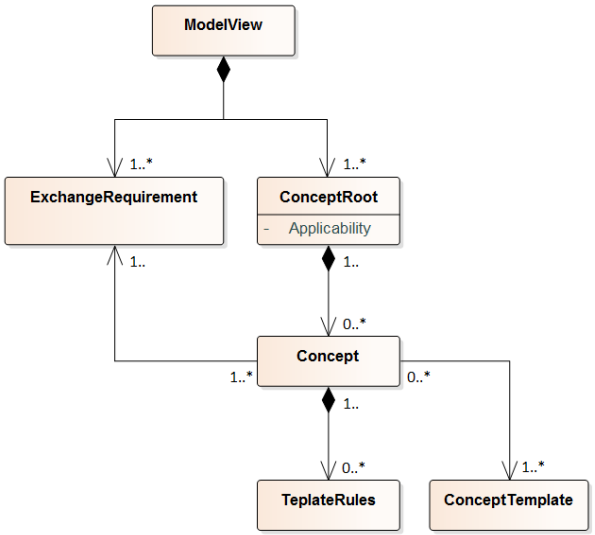

Abbildung 1: Vereinfachte schematische Darstellung des mvdXML Datenformates

Das Element *ModelView* beschreibt eine MVD. Es enthält alle Definitionen und bezieht sich auf eine spezifische Version des IFC Datenmodells. Zusätzlich enthält es Elemente vom Typ *ExchangeRequirement* und *ConceptRoot*. Bei der Abbildung einer MVD ist zu beachten, dass die enthaltenen Klassen und Typen einer MVD nicht explizit abgebildet werden. Vielmehr werden diese indirekt über die *TemplateRules* des jeweiligen *Concept* in *ConceptRoot* abgebildet und beschreiben dadurch alle Elemente im Kontext einer MVD.

Ein *ExchangeRequirement* beschreibt die für ein konkretes Austauschszenario notwendigen Daten. Dies bezieht sich entweder auf den Import oder den Export. Zusätzlich lassen sich weitere Bedingungen für die verwendeten *Concepts* angeben.

Ein *ConceptRoot* fasst verschiedene *Concepts* einer spezifische IFC Klasse, die dieselben Bedingungen erfüllen sollen, zusammen. In den meisten Fällen bezieht sich ein *ConceptRoot* auf IFC Klassen, die von der Basisklasse *IfcRoot* abgeleitet sind (z. B. Projektstruktur, Bauteile, Relationen, ...). Darüber hinaus lassen sich mit Hilfe des Elements *Applicability* zusätzliche Bedingungen definieren, die von einer solchen Klasse erfüllt werden müssen. Erst wenn eine Instanz diese Bedingungen erfüllt, müssen die zugeordneten *Concepts und TemplateRules* eingehalten werden.

Ein *Concept* beschreibt die Verwendung von Attributen und Beziehungen einzelner Klassen und wird in der Regel so definiert, dass es wiederverwendbar und auf möglichst viele MVDs anwendbar ist. Sind für ein *Concept* Regeln - sogenannte *TemplateRules* - definiert, so enthält es zusätzlich eine Referenz auf das entsprechende *ConceptTemplate*. Die Beziehung zwischen einer *TemplateRule* und den in den *ConceptTemplates* definierten Attributen und Klassen erfolgt über eine sogenannte *RuleID*. Zusätzlich stehen Schlüsselwörter zur Verfügung, die den Typ einer Bedingung festlegen:

- *Value*: Prüft den Wert eines Attributes
- *Size*: Prüft die Anzahl der Elemente einer Aggregation
- *Type*: Prüft den Typ eines Attributes
- *Unique*: Prüft, ob ein Wert innerhalb eines Dokumentes eindeutig ist
- *Exists*: Prüft ob ein Attribut oder eine Klasse vorhanden ist

Die einzelnen Bedingungen der *TemplateRules* benutzen die Konzepte der Aussagenlogik, wobei jede Bedingung (*TemplateRule*) immer einen Wahrheitswert – richtig oder falsch – repräsentiert. Die *TemplateRules* sind zusätzlich durch logische Operatoren (and, or, not, nand, nor, xor und nxor) miteinander verknüpft und bilden einen Regelbaum.

Ein *ConceptTemplate* bildet die Attribute und Beziehungen entsprechend des verwendeten Datenschemas für eine spezifische Klasse ab. Dabei ist es nicht zwingend erforderlich, die vollständige Definition des Datenschemas anzugeben. Es werden nur die Attribute und Relationen angegeben, die innerhalb eines *Concepts* erforderlich sind. Im Unterschied zu den Definitionen ist es hier auch möglich, Attribute und Relationen in der Definition einer Basisklasse anzugeben, die erst in abgeleiteten Klassen zu Verfügung stehen. Beispielsweise ist dies ein *ConceptTemplate* zur Prüfung aller Bauteile (IfcBuildingElement), mit dem ein spezifisches Attribut der Klasse Wand (z. B. *PredefinedType*) überprüft wird. Damit ist die korrekte Interpretation der Prüfregeln in den *TemplateRules* nur in Verbindung mit dem entsprechenden IFC Schema möglich.

MVD PRÜFREGELN FÜR DIE THERMISCHE GEBÄUDESIMULATION

Das vorrangige Ziel des vorliegenden Beitrags ist nicht die Entwicklung einer neue MVD. Dazu existieren im BLIS MVD Katalog bereits eine Reihe von informellen Definitionen von Projektgruppen, die sich ausführlich mit den unterschiedlichen Aspekten der thermischen Gebäudesimulation befasst haben (BLIS, 2018). Diese Definitionen liegen in Form von formalisierten Dokumenten vor und sind somit nur bedingt für eine automatisierte Datenverarbeitung verwendbar. Für keine dieser MVDs existiert aktuell eine mvdXML. Der Ansatz des vorliegenden Beitrags besteht deshalb darin zu analysieren, inwieweit es möglich ist, auf Basis von mvdXML sinnvolle Regeln zu definieren, um IFC Daten als Eingabedatum für die thermische Gebäudesimulation zu nutzen. Diese Regeln werden primär auf Grundlage der offiziellen MVDs für IFC2x3 (CoordinationView 2.0) und IFC4 (ReferenceView, DesignTransferView) entwickelt. In einem zweiten Schritt ist geplant, im Rahmen der Arbeit innerhalb eines sogenannten „buildingSMART BuildingRooms“, diese Erkenntnisse und Erfahrungen in die Definitionen einer offiziellen Model View einzubringen.

Definition von Prüfregel

Das IFC Datenmodell ist prinzipiell in der Lage, Eingabedaten für eine thermische Gebäudesimulation zu repräsentieren. Gerade mit der Version IFC4 sind eine Reihe von Überarbeitungen und Erweiterungen eingeflossen, die speziell für die energetische

Betrachtung von Gebäuden relevant sind. Dazu gehören die Erweiterungen zur Unterstützung beliebiger geographischer Bezugssysteme, um die genaue Lage eines Gebäudemodells zu beschreiben, die Überarbeitung der Materialeigenschaften, sowie Erweiterungen bei der Verwendung von Raumbegrenzungselementen (Geiger et al., 2016).

Abbildung 2 stellt den Umfang der für eine thermische Gebäudesimulation notwendigen Informationen dar. Dieser Umfang lässt sich prinzipiell auch im IFC Datenmodell bilden. Die offiziellen buildingSMART MVDs für die Versionen IFC2x3 (CoordinationView 2.0) und IFC4 (ReferenceView bzw. DesignTransferView) decken dabei primär den linken Teil ab, der sich auf die Abbildung der Gebäudestruktur, der Bauteilgeometrien und der Definition von Räumen konzentriert.

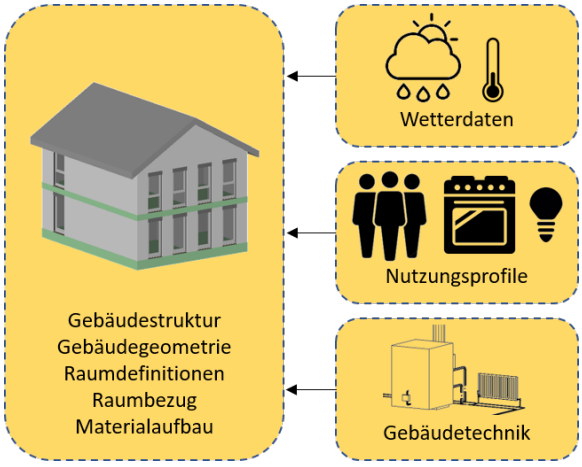

Abbildung 2: Informationsumfang für eine thermische Gebäudesimulation

Zu den Bauteilen kommen zusätzlich noch Angaben über die zugeordneten Material- und Schichtinformationen. Darüber hinaus kann ein Gebäudemodell noch über eine geographische Lage und eine Ausrichtung verfügen. Im Gegensatz dazu werden die ebenfalls für die thermische Gebäudesimulation notwendigen Informationen zu den klimatischen Bedingungen, den Nutzungsbedingungen der Räume bzw. thermischen Zonen und der umfassenden Beschreibung der Gebäudetechnik nicht im erforderlichen Umfang von den bestehenden Implementierungen unterstützt. Der vorliegende Beitrag konzentriert sich deshalb auf den linken Teil, der gegenwärtig von den gängigen Werkzeugen unterstützt wird. Der Umfang an Regeln soll sicherstellen, dass die minimalen Anforderungen für die Datenübergabe von der Architektur zu einem Simulationswerkzeug erfüllt werden, und prüft hierzu die Existenz von Attributen und Werten, sowie teilweise deren Wertebereich.

Tabelle 1: Formale Definition von Prüfregeln

Mindestens ein Gebäude muss vorhanden sein
Mindestens ein Raum muss vorhanden sein und der Raum muss einem Gebäude zugeordnet sein
Jeder Raum muss eine Raumgeometrie enthalten
Die räumliche Lage und Ausrichtung muss definiert sein
Raumbegrenzungselemente müssen vorhanden sein
Mindestens ein Bauteil muss vorhanden sein
Alle Bauteile müssen eine Volumengeometrie besitzen
Bauteile müssen Materialinformationen enthalten
Materialien müssen physikalische Parameter enthalten: - Wärmeleitfähigkeit (ThermalConductivity) - Dichte (Density) - Wärmekapazität (SpecificHeatCapacity)

Ausgehend von diesen Randbedingungen lassen sich die in Tabelle 1 aufgelisteten minimalen Anforderungen an den Informationsgehalt des IFC Modells formulieren. Im Rahmen des vorliegenden Beitrags wird beispielhaft eine Regel zur Überprüfung der physikalischen Parameter eines Materials näher betrachtet.

Tabelle 2: Auszug einer mvdXML Regel zum Prüfen der physikalischen Parameter eines Materials

```
<TemplateRules operator="and">
  <TemplateRule Parameters ="Material[Exists]=TRUE"/>
  <TemplateRules operator="and">
    <TemplateRule Parameters="MaterialProperties[Size]>=3"/>
    <TemplateRule Parameters="MaterialName[Exists]=TRUE"/>
    <TemplateRules operator="and">
      <TemplateRule Parameters="PropertySetName[Value]=
        'Pset_MaterialThermal' and SimplePropertyName[Value]=
        'ThermalConductivity' and NominalValue[Value] > 0.0"/>
      <TemplateRule Parameters="PropertySetName[Value]=
        'Pset_MaterialThermal' and SimplePropertyName[Value]=
        'SpecificHeatCapacity' and NominalValue[Value] > 0.0"/>
      <TemplateRule Parameters="PropertySetName[Value]=
        'Pset_MaterialCommon' and SimplePropertyName[Value]=
        'MassDensity' and NominalValue[Value] > 0.0"/>
    </TemplateRules>
  </TemplateRules>
</TemplateRules>
```

Der in Tabelle 2 dargestellte Auszug einer Prüfregel soll sicherstellen, dass an jedem Material die physikalischen Parameter gesetzt und ihre Werte größer 0.0 sind. Dabei wird zunächst mit „Material[Exists]=TRUE“ geprüft, ob ein Material vorhanden ist. Anschließend wird mit „MaterialProperties[Size]>=3“ sichergestellt, dass mindestens drei Eigenschaftssätze einem

Material zugeordnet sind. Die nachfolgende Regel „MaterialName[Exists]=TRUE“ kontrolliert, dass für jedes Material ein Name vergeben wurde. Die eigentliche Prüfung der physikalischen Parameter erfolgt mit den darauffolgenden drei Regeln. Dabei wird jeweils der Name des Eigenschaftssets z.B. „Pset_MaterialThermal“, der Name der Eigenschaft z.B. „ThermalConductivity“ und der geforderte Wert, in diesem Fall „> 0.0“ überprüft.

Technische Umsetzung der mvdXML Regeln

Die Anwendung IFCExplorer ist ein allgemeines Werkzeug zur Analyse, Visualisierung, Prüfung und Integration semantischer Datenmodelle aus den Bereichen BIM und GIS (Benner et al., 2013). Für das Datenmodell IFC stehen neben einer Schemaprüfung eine Reihe weiterer Modellprüfungen zur Verfügung. Diese Prüfmechanismen sind bislang fest implementiert. Mit der Regelsyntax von mvdXML steht nun ein standardisiertes Format zur flexiblen Spezifikation solcher Prüfungen zur Verfügung. Daher und auch um die zuvor definierten mvdXML-Regeln für die thermische Gebäudesimulation anzuwenden zu können, wird ein entsprechendes Modul zur Regelprüfung implementiert (siehe Abbildung 3).

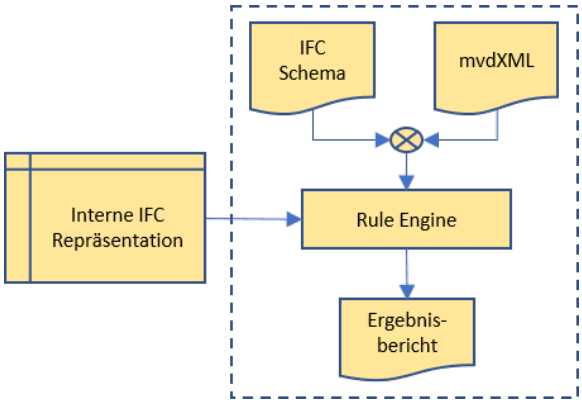

Abbildung 3: Schematische Darstellung des entwickelten Moduls zur Regelprüfung

Dieses Modul beinhaltet neben einer Reihe von Dialogen zur Steuerung des Prüfvorgangs, der Analyse von mvdXML Dokumenten und zur Präsentation der Prüfergebnisse, insbesondere eine Komponente für die Regelprüfung. Diese Komponente erlaubt sowohl eine Konsistenzprüfung der mvdXML Regeln, als auch die Prüfung eines IFC Instanz Dokumentes gegen diese Regeln. In einem separat steuerbaren Validierungsprozess wird die Konsistenz der Regeln in den *ConceptTemplates* gegen das entsprechende IFC Schema sichergestellt. Dabei müssen die in den *ConceptTemplates* verwendeten Klassen und Attribute dem verwendeten Datenschema entsprechen, sowie die im Datenschema vorgegebene Reihenfolge einhalten. Zusätzlich werden die in den *ConceptTemplates* verwendeten Bezeichner (*RuleID*) auf ihre Eindeutigkeit überprüft. Dies ist notwendig, da über eine *RuleID* die Zuordnung zwischen einer Regel aus einem *Concept* mit einem Attribut oder einer Klasse in einem *ConceptTemplate* erfolgt. Nur so kann sichergestellt werden, dass während des eigentlichen Prüfvorganges die Bedingungen pro *Concept* korrekt interpretiert und verarbeitet werden können.

Ein zentraler Dialog steuert sowohl die Analyse- wie auch die Prüffunktionalitäten (siehe Abbildung 4).

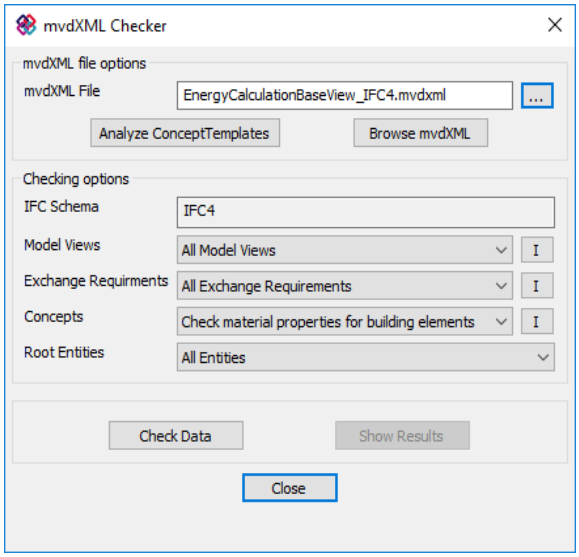

Abbildung 4: Benutzerdialog zur Steuerung der Regelprüfung

Dazu zählt insbesondere der zu wählende Kontext einer Prüfung. Wahlweise lassen sich die in einer mvdXML Datei enthaltenen *ModelViews*, *ExchangeRequirements* und *Concepts* nach Bedarf auswählen. Ein weiteres Dialogfenster präsentiert - entsprechend der für den Prüfkontext ausgewählten Konzepte - das Ergebnisprotokoll eines Prüflaufes (siehe Abbildung 6).

Prüfergebnisse

Als Praxisbeispiel dient ein in dem Architektur CAD System ArchiCAD modelliertes Gebäude. Dabei handelt es sich um ein konkretes Gebäude, das im Rahmen eines

Forschungsprojektes auf dem KIT Campus Nord errichtet wird (siehe Abbildung 5).

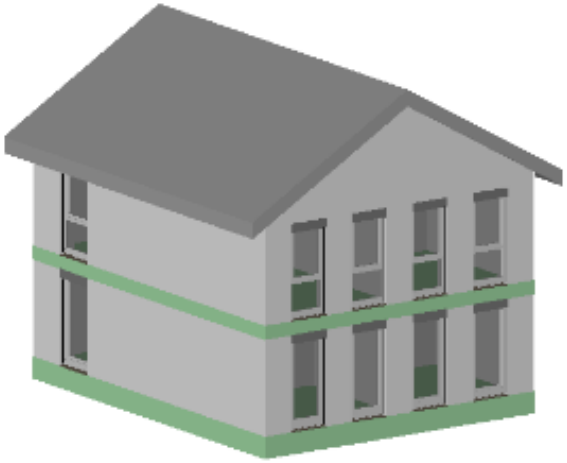

Abbildung 5: Praxisbeispiel KIT Musterhaus

Bei der Erstellung dieses Gebäudemodells wurde besonders Wert auf die korrekte Modellierung und die vollständige Definition der bauteilspezifischen Parameter in ArchiCAD gelegt. Den Bauteilen wurden entsprechende Schichtaufbauten zugewiesen, und pro Schicht die physikalischen Parameter für Wärmeleitfähigkeit λ, Dichte ρ und die Wärmekapazität C angegeben. Weiterhin wurde Wert auf den korrekten Raumbezug, wie die Lage und die Ausrichtung des Gebäudes, gelegt.

Um die Vollständigkeit der erforderlichen Parameter zu kontrollieren, wird das Gebäude aus ArchiCAD im Format IFC4 (DesignTransferView) exportiert und gegen die definierten Regeln geprüft. In Abbildung 6 ist das detaillierte Prüfprotokoll der Bodenplatte zur Überprüfung der physikalischen Parameter dargestellt. Dieses Beispiel zeigt, dass bei der Eingabe der Materialparameter in ArchiCAD vergessen wurde, für die Bodenplatte dem Parameter „SpecificHeatCapacity“ einen Wert zuzuweisen.

MessageLog

Message Type	Message Description
MvdXML	Checking concept: Check material properties for building elements
	Applicable for: IfcBuildingElement
	Checked entities total: 40
	Successful checked entities: 2
	Failed entities: 38
Failed	Result concept: "Check material properties for building elements" for IfcSlab[Floor]
	Type: IfcSlab[Floor] | Name: Decke-001 | OID: #217
MvdXML Conditions	
Successfull	Check condition: MaterialRelation [Size] == 1 (IfcRelAssociatesMaterial)
Successfull	Check condition: RelatingMaterialLayerSetUsage [Type] == IfcMaterialLayerSetUsage (IfcMaterialLayerSetUsage)
Successfull	Check condition: LayerSetName [Exists] == TRUE (IfcMaterialLayerSet.LayerSetName)
Successfull	Check condition: MaterialLayers [Size] >= 1 (IfcMaterialLayerSet.MaterialLayers)
Successfull	Check condition: Material [Exists] == TRUE (IfcMaterialLayer.Material)
Successfull	Check condition: MaterialProperties [Size] >= 3 (IfcMaterial.HasProperties)
Successfull	Check condition: MaterialName [Exists] == TRUE (IfcMaterial.Name)
Successfull	Check condition: PropertySetName [Value] == Pset_MaterialThermal (IfcMaterialProperties.Name)
Successfull	Check condition: SimplePropertyName [Value] == ThermalConductivity (IfcSimpleProperty.Name)
Successfull	Check condition: NominalValue [Value] > 0.0 (IfcSimpleProperty.NominalValue)
Successfull	Check condition: PropertySetName [Value] == Pset_MaterialThermal (IfcMaterialProperties.Name)
Successfull	Check condition: SimplePropertyName [Value] == SpecificHeatCapacity (IfcSimpleProperty.Name)
Failed	Check condition: NominalValue [Value] > 0.0 (IfcSimpleProperty.NominalValue)
Successfull	Check condition: PropertySetName [Value] == Pset_MaterialCommon (IfcMaterialProperties.Name)
Successfull	Check condition: SimplePropertyName [Value] == MassDensity (IfcSimpleProperty.Name)
Successfull	Check condition: NominalValue [Value] > 0.0 (IfcSimpleProperty.NominalValue)

Speichern ... Clear OK

Abbildung 6: Ergebnisprotokoll der Regelprüfung eines Bauteils

Dabei handelt es sich um einen typischen Fehler, wie er im Projektalltag auftritt. Mit Hilfe entsprechend definierter Regeln in mvdXML und einem entsprechenden Prüfwerkzeug lassen sich solche Fehler sehr einfach identifizieren und sicherstellen, dass IFC Gebäudemodelle im Datenaustausch die Anforderungen an vordefinierte Prozesse erfüllen.

Erfahrungen

Die Testergebnisse zeigen, dass mit den Regeldefinitionen der Version mvdMXL 1.1 die Qualität des Datenaustausches deutlich verbessert werden kann. In dem beschriebenen Gebäudemodell konnten sehr schnell vergessene Parameter identifiziert und ein IFC

Gebäudemodell mit den für eine thermische Gebäudesimulation erforderlichen Eingabedaten erstellt werden. Es hat sich aber auch gezeigt, dass es eine Reihe von Prüfaufgaben gibt, die mit dieser Version mvdXML 1.1 nicht oder nur über Umwege realisiert werden können.

So ist es aktuell nicht möglich, mit einer einfachen Regel zu prüfen, ob eine Instanz eines spezifischen Typs in einem IFC Instanz Dokument enthalten ist. Ein Beispiel ist die Prüfung, ob der zu prüfende IFC Datensatz über Gebäude (*IfcBuilding*) verfügt. Dies lässt sich mit der aktuellen Version von mvdXML nur über Umwege prüfen, zum Beispiel mit Hilfe eines *ConceptRoot* für IfcProject, in dem die entsprechenden Relationen der Projektstruktur analysiert werden und geprüft wird, ob innerhalb dieser Projektstruktur Gebäude enthalten sind.

Ein weiter Schwachpunkt der aktuellen Version ist die Unterscheidung der Prüftiefe innerhalb einer Liste von Elementen. Es gibt Prüfaufgaben, in denen alle Elemente einer Liste eine Bedingung erfüllen müssen, und es gibt den Fall, dass nur eine Teilmenge der Elemente diese Bedingung erfüllen muss. Dies lässt sich mit der aktuellen Definition von mvdXML nicht unterscheiden. Um das in einem Beispiel zu veranschaulichen, wird erneut ein Schichtaufbau betrachtet. Im ersten Fall müssen alle Materialien des Schichtaufbaus die physikalische Parameter enthalten (siehe Tabelle 3, oben). Im zweiten Fall soll im selben Schichtaufbau mindestens ein Material mit dem Namen „Beton enthalten“ (siehe Tabelle 3, unten).

Tabelle 3: Beispiel für den unterschiedlichen Prüfkontext von Regeln

<TemplateRule Parameters="PropertySetName[Value]= 'Pset_MaterialThermal' and SimplePropertyName[Value]= 'ThermalConductivity' and NominalValue[Value] > 0.0"/>
<TemplateRule Parameters="MaterialName[Value]= 'Beton'"/>

ZUSAMMENFASSUNG

Im vorliegenden Beitrag wird gezeigt, wie sich auf Basis des buildingSMART Standards mvdXML spezifische Regeln für die thermische Gebäudesimulation definieren lassen. Diese Regeln können verwendet werden um Klassen und Attribute festzulegen oder einzuschränken, Eigenschaftssets zu definieren, und Gültigkeitsbereiche von Attributen festzulegen. Das Ziel dieser Regeln ist sicherzustellen, dass IFC Instanz Dokumente die für eine thermische Gebäudesimulation erforderlichen Parameter enthalten. An einem konkreten Gebäudemodell, erstellt mit ArchiCAD im Format IFC4 (Design TransferView), wird die Anwendung der Regeln demonstriert. Während des Modellierungs-prozesses des Gebäudemodells zeigt sich sehr schnell, dass sich mit Hilfe der ersten definierten Regeln Fehler, insbesondere bei den Attributen und Eigenschaftssätzen, leicht identifizieren lassen.

Mit der Version 1.1 von mvdXML steht ein Format zur Verfügung, das die Qualität des Datenaustausches im Alltag deutlich verbessern könnte. Voraussetzung ist, dass die benötigten Regeln auf Basis einer offiziellen MVD definiert werden. Leider verfügen die gängigen Modellierungswerkzeuge bislang nicht über die entsprechende Unterstützung des mvdXML Formats, und die Anzahl der verfügbaren Werkzeuge mit mvdXML Unterstützung ist noch überschaubar.

Der vorliegende Beitrag zeigt aber auch, dass die Prüfmöglichkeiten von mvdXML noch erweitert werden müssen. Entsprechende Vorschläge werden innerhalb der Arbeitsgruppe MSG (Modelling Support Group) von buildingSMART diskutiert und erarbeitet.

LITERATUR

Baumgärtel, J., Pirnbaum, S., 2016, „Automatische Prüfung und Filterung in BIM mit Model View Definitions“, 28. Forum Bauinformatik 2016, Hannover, 19. – 21.9.2016

Benner, J., Geiger, A., Häfele, K.-H., Knüppel, H., 2013, „IFCExplorer – Ein Werkzeug für die Integration unterschiedlicher raumbezogener semantischer Daten“, GEOINFORMATIK, UNIVERSITÄT HEIDELBERG

BLIS-Project, Website: http://www.blis-project.org/IAI-MVD/MVDs/

Borrmann, A., König, M., Koch, C., Beetz, J. , 2015, „Building Information Modeling - Technologische Grundlagen und industrielle Praxis“, Springer Verlag, VDI-Buch, ISBN 978-3-658-05605-6

buildingSMART, 2018, IFC4 Certification, Website: http://www.buildingsmart-tech.org/certification

buildingSMART IFC Documentation Generator Tool, 2012, OpenSource project buildingSMART, Website: http://www.buildingsmart-tech.org/downloads/accompanying-tools/ifcdoc/application

Chipman, T., Liebich, T., Weise, M., 2016, „Specification of a standardized format to define and exchange“, buildingMSART International Ltd., 15.02.2016

Geiger, A., Reichenbach, I., Häfele, K.-H., 2016, „IFC-Daten für die thermische Gebäudesimulation“, 28. Forum Bauinformatik 2016, Hannover, 19. – 21.9.2016

Hausknecht, K., Liebich, T., 2018, „BIM-Kompendium, Building Information Modeling als neue Planungsmethode“, Fraunhofer IRB Verlag, ISBN 978-3-8167-9948-1

Hietanen, J., 2006, „IFC model view definition format“, buildingSMART International

Hietanen, J., 2014, „MVD and Simplebim“, Datacubist, Website:

ISO 16739. 2013. ISO 16739:2013 Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries. 2013

Liebich, T., Geiger, A., Katranuschkov, P., Linhard, K., Steinmann, R., Weise, M., 2011, „mvdXML specification, mvdXML Schema“, buildingSMART International

Lockley, S., Benghi, C., Černý, M., 2017, „Xbim.Essentials: a library for interoperable building information applications“, JOSS, Journal of Open Source Software

Liebich T., Weise M., Scherer R.J., 2014, „Prüfung und Erweiterung von Fachmodellen am Beispiel von IFC“, In: Scherer R., Schapke SE. (eds) Informationssysteme im Bauwesen 1. VDI-Buch. Springer Vieweg, Berlin, Heidelberg

Weise, M., Nisbet, N., Liebich, T., Benghi, C., 2016, „IFC model checking based on mvdXML 1.1”

Zhang, C., Beetz, J., Weise, M., 2014, „Model view checking: automated validation for IFC building models“, European Conference on Process and Product Modelling; Wien, 17 – 19.20.2014