

DIPLOMARBEIT

Qualitative Bewertung gekoppelter Software-Systeme des
Bauingenieurwesens auf der Grundlage von Schema-Analyse und
Schema-Mapping

Sebastian Grolla

Bauhaus-Universität Weimar
Professur Baustatik
Graduiertenkolleg 1462

Oktober 2011

Erstprüfer: Prof. Dr.-Ing. C. Könke
Zweitprüfer: Dipl.-Ing. T. Fröbel

Diplomarbeit B/2011/3

Bearbeiter: Sebastian Grolla, B/95/I, 950 888

Thema: Qualitative Bewertung gekoppelter Software-Systeme des Bauingenieurwesens auf der Grundlage von Schema-Analyse und Schema-Mapping

Fachgebiet: Graduiertenkolleg 1462 - Bewertung gekoppelter numerischer Partialmodelle im Konstruktiven Ingenieurbau

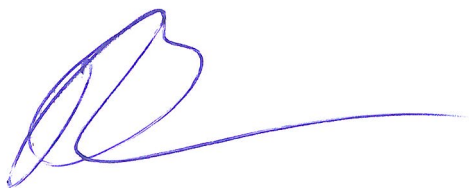
Erstprüfer: Prof. Dr.-Ing. C. Könke, Professur Baustatik

Zweitprüfer: Dipl.-Ing. T. Fröbel, Graduiertenkolleg 1462

Bearbeitungsdauer: 12 Wochen

Ausgabedatum: 14.07.2011

Abgabedatum: 06.10.2011

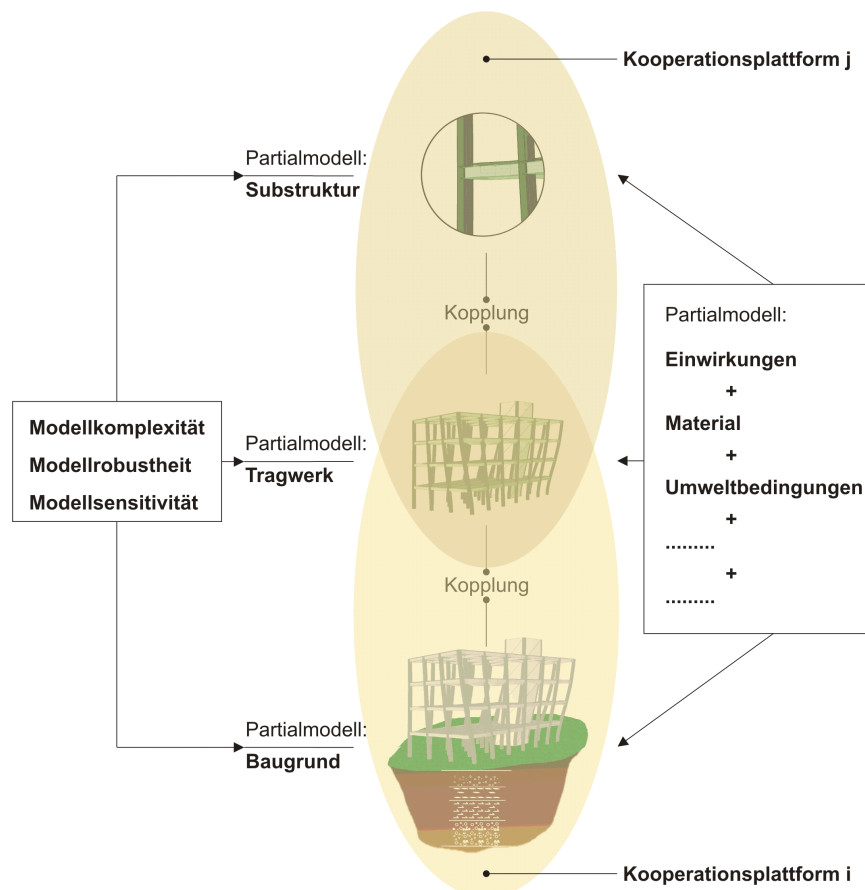


Prof. Dr.-Ing. Timon Rabczuk
Vorsitzender des Prüfungsausschusses

Aufgabenstellung

Thema: Qualitative Bewertung gekoppelter Software-Systeme auf der Grundlage von Schema-Analyse und Schema-Mapping

Aufgabe: Das Bauingenieurwesen beschäftigt sich mit der Errichtung von Bauwerken. Hierfür sind verschiedenste Aufgaben, wie die Konzipierung, Planung, und Bemessung des Bauwerks notwendig. Infolge der Komplexität und des Unikatcharakters von Bauwerken werden jedoch die Aufgaben in Teilaufgaben zerlegt, welche dann von verschiedenen Ingenieuren und mit unterschiedlichen Anwendungen räumlich getrennt bearbeitet werden können. In einem weiteren Schritt müssen die entstanden Lösungen wieder gesamtheitlich zusammengeführt werden. Dies führt zu einem hohen Grad an Kommunikation zwischen den beteiligten Ingenieuren und den verwendeten Anwendungen. Eine entscheidende Rolle für eine erfolgreiche Zusammenarbeit und für die Lösung der Aufgabe spielt der fehlerfreie Datenaustausch zwischen den unterschiedlichen Softwareanwendungen. Diese wird jedoch durch die verschiedenen meist proprietären Datenschemas und Austauschformate erschwert.



Ziel der Diplomarbeit ist die qualitative Bewertung gekoppelter Software-Systeme des Bauingenieurwesens auf der Grundlage von Schema-Analyse und Schema-Mapping am Beispiel des Datenaustauschformates IFC und dem FEM-System Ansys. Die Bewertung soll an einem Referenzobjekt des Graduiertenkollegs (Skeletttragewerk) durchgeführt werden.

- Die IFC – ISO 16739 (Industry Foundation Classes) ist ein offener Standard zur digitalen Beschreibung von Gebäudemodellen. Mit Hilfe der IFC können vor allem logische Gebäudestrukturen und Bauteile wie z.B. Wände, Fenster, Geschosse, Türen aber auch dazugehörige Materialien, Geometrien und Eigenschaften modelliert werden.
- Das FEM-System Ansys wird unter anderem zur Bearbeitung von linearen und nichtlinearen Aufgabenstellungen aus den Bereichen der Struktur-, Fluid- und Thermodynamik eingesetzt. Es verwendet eine Vielzahl von Elementtypen für die Modellierung von 1-, 2-, und 3-dimensionalen Aufgaben und unterstützt verschiedenste Materialien und Stoffgesetze.

Folgende Aufgaben sind zu bearbeiten:

1. Stand der Technik im Bereich der qualitativen Bewertung des Datenaustausches von gekoppelten Softwareanwendungen.
2. Durchführung von Schema-Analyse und Schema-Mapping an einem Referenzobjekt des Graduiertenkollegs.
3. Qualitative Bewertung des Datenaustausches (basierend auf Punkt 2) für verschiedene Modellierungsgrade.
4. Zusammenfassung der Arbeit und Ausblick auf weitere denkbare Untersuchungen.

Für die Betreuung der Arbeit wird eine regelmäßige Konsultation mit Vorstellung der aktuellen Ergebnisse verlangt.

Es sind drei Exemplare und eine digitale Fassung der Dokumentation der Arbeit abzugeben. Der Quellcode ist ausreichend zu kommentieren und der Dokumentation beizufügen.

Die Arbeit wird durch eine Präsentation mit anschließender Diskussion verteidigt.

Inhaltsverzeichnis

Einleitung	1
Motivation	1
Ziel der Arbeit	1
1. Stand der Technik	3
1.1. Kopplung von Anwendungen	3
1.2. Arten der Qualitätsbewertung	4
1.3. Formalisierung des Schema-Mappings	5
1.4. Qualitative Bewertung	7
1.5. Bewertetes Schema-Mapping	9
2. Das Referenzobjekt	12
2.1. Hintergrund	12
2.2. Konstruktion	12
3. Die Industry Foundation Classes	14
3.1. Überblick	14
3.2. Die EXPRESS-Definition	15
3.2.1. Basisdatentypen	15
3.2.2. Deklarationen von Datentypen	16
3.2.3. Entities	16
3.2.4. Attribute	17
3.2.5. Weitere lokale Regeln	18
3.2.6. Globale Funktionen	19
3.2.7. Verknüpfung mit anderen Schemata	20
3.3. Aufbau der IFC	20
3.3.1. Grundlegendes	20
3.3.2. Gesamtarchitektur	21
3.3.3. Projektcontainer und Grundeinstellungen	22
3.3.4. Räumliche Struktur	23
3.3.5. Platzierung	25
3.3.6. Geometrische Repräsentation	25

3.3.7. Gebäudeelemente	28
3.3.7.1. Stützen, Balken und Lastglieder	30
3.3.7.2. Wände	30
3.3.7.3. Platten	31
3.3.8. Strukturmodell	31
3.3.9. Material	35
3.4. Das Open IFC Toolkit	36
4. Ansys	37
4.1. Überblick	37
4.2. Modellierung in Ansys	38
4.3. APDL	40
5. Modellierung des Referenzobjektes	41
5.1. Allgemeines	41
5.2. Modellierungsmöglichkeiten	41
5.3. Das Referenzobjekt	42
6. Schema-Analyse	44
6.1. Zweck einer Schema-Analyse	44
6.2. Der EXPRESS-Analyzer	44
6.3. Analyse	46
6.3.1. Geometrie	46
6.3.1.1. Randbedingungen	46
6.3.1.2. IfcColumn	47
6.3.1.3. Anwendung auf andere Elemente	50
6.3.2. Material	53
6.3.3. Strukturmodell	56
7. Schema-Mapping	58
7.1. Generelles	58
7.2. Geometrie	60
7.3. Material	62
7.4. Strukturmodell	64
7.5. Resultate	65
8. Qualitative Bewertung	67
8.1. Bewertungsvarianten	67
8.2. Parameterbasierte Bewertung	67
8.3. Querschnittsbasierte Bewertung	70

8.4. Betrachtungen zur Bewertung des Strukturmodells	72
9. Zusammenfassung und Ausblick	74
9.1. Zusammenfassung	74
9.2. Ausblick	74
 Literaturverzeichnis	 I
Abbildungsverzeichnis	III
Anhang	V
A. Kurzübersicht über EXPRESS-G	VI
B. Schema des EXPRESS-Parsers in Backus-Naur-Form	VII
Selbstständigkeitserklärung	X

Einleitung

Motivation

Die Errichtung von Bauwerken erfordert im Vorfeld eine Reihe von Aufgaben, wie die Konzipierung, Planung, Konstruktion und Bemessung derselben. Die zuständigen Ingenieure bearbeiten jeweils eine Teilaufgabe, deren Ergebnisse später wieder in das gesamte Modell einfließen. Dies führt zu einem hohen Grad an Kommunikation zwischen den beteiligten Ingenieuren und den verwendeten Anwendungen. Voraussetzung für eine erfolgreiche Zusammenarbeit und die daraus resultierende Lösung der Aufgabe ist der fehlerfreie Datenaustausch zwischen den unterschiedlichen Softwareanwendungen. Diese wird jedoch durch eine Vielzahl von proprietären Datenformaten erschwert. Daher spielt die Untersuchung eines Datenaustausches zwischen den Anwendungen in Bezug auf die Qualität des Austauschs eine wichtige Rolle.

Ziel der Arbeit

Diese Arbeit soll dem Leser einen Überblick über die Möglichkeiten zur Qualitätsbewertung von Anwendungskopplungen verschaffen. Anhand eines Beispiels wird ein Objekt mit einem Standard modelliert, analysiert, in einen anderen Standard übertragen und diese Übertragung bewertet.

Im ersten Kapitel wird ein Einblick in den Stand der Technik gegeben. Als Nächstes wird das der Modellierung zugrundeliegende Referenzobjekt kurz vorgestellt. Mit den Industry Foundation Classes (IFC) steht ein offener Standard zur Verfügung, mit dem ein Bauwerk unter mehreren Aspekten modelliert werden kann. Dieser Standard wird in Kapitel 3 vorgestellt, wobei der Schwerpunkt auf die geometrische Modellierung gelegt wird. Das System, in welches das Modell übertragen wird, ist Ansys. Darauf wird in Kapitel 4 eingegangen.

Um eine Grundlage für die Untersuchungen zu schaffen, soll ein Referenzobjekt auf Basis der IFC modelliert werden. Diese Modellierung wird in Kapitel 5 vorgestellt.

Durch eine Schema-Analyse wird in Kapitel 6 untersucht, welche Klassen und Beziehungen zwischen den Klassen innerhalb der IFC benötigt werden, um alle Eingangsparameter des Referenzmodells zu erfassen.

Beim Schema-Mapping in Kapitel 7 werden Analogien zwischen dem IFC-Standard und den Datenstrukturen in Ansys gefunden und eine Kopplung entwickelt, um das Modell von den IFC nach Ansys zu transferieren.

Danach erfolgt im Kapitel 8 eine qualitative Bewertung des Mappings unter zwei verschiedenen Gesichtspunkten. Die Bewertung wird für unterschiedliche Varianten des Referenzmodells durchgeführt, wobei jeweils ein gesamtheitlicher Qualitätswert ermittelt wird.

Anschließend werden im letzten Kapitel die gewonnenen Erkenntnisse und Ergebnisse zusammengefasst und ein Ausblick auf weitere Möglichkeiten und Verwendungen der Verfahren gegeben.

Zu Beginn soll jedoch grundlegend erläutert werden, weshalb eine Bewertung der Qualität von Kopplungen von Softwaresystemen nötig ist und welche Verfahrensweisen es dafür gibt.

1. Stand der Technik

1.1. Kopplung von Anwendungen

Seit die Gebäudeplanung fast nur noch ausschließlich am Computer stattfindet, herrscht ein reger Datenaustausch zwischen unterschiedlichen Applikationen. Teilaufgaben werden mit den entsprechenden Anwendungen bearbeitet und ihre Ergebnisse untereinander weitergegeben.

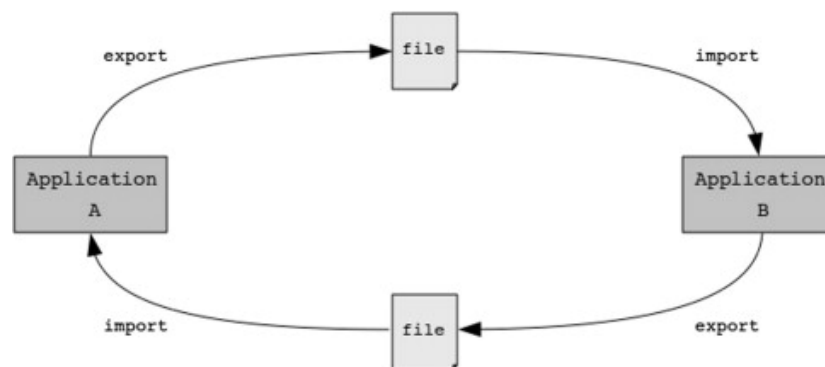


Abb. 1.1.: Dateibasierter Datenaustausch zwischen Applikationen. Quelle: [FFK11]

Die Programme nutzen oftmals eigene, auf sie zugeschnittene Datenstrukturen. Die Art der Strukturen hängt von den Anforderungen der Programme ab, die damit arbeiten. Dabei spielt es eine Rolle, wie komplex die zu abstrahierenden Informationen sind und in welcher Programmiersprache die Programme geschrieben wurden.

Um Daten zwischen Anwendungen auszutauschen, müssen sie vom Modell einer Anwendung auf das Modell einer anderen Anwendung abgebildet - gemappt - werden. Die Modelle sind oft sehr komplex und verschieden und besitzen Zusammenhänge, die in einem anderen Modell so nicht abbildbar sind. Es entstehen Verluste und Fehler. Gerade im Ingenieurwesen, wo es bei Konstruktionen und Berechnungen um äußerste Genauigkeiten geht, wirken sich diese Fehler gravierend aus. Daher ist eine qualitative Bewertung von Kopplungen bzw. Mappings wichtig und nötig [FFK11].

1.2. Arten der Qualitätsbewertung

Die Bewertungsverfahren lassen sich in zwei grundlegende Gruppen einteilen: In *Aposteriori*-Verfahren und in *Apriori*-Verfahren.

Aposteriori-Verfahren werden angewandt, wenn das Mapping vollzogen wurde und die Daten in der konvertierten Form vorliegen. Eine triviale Anwendung dieses Verfahrens ist die Sichtprüfung. Der Nutzer vergleicht am Monitor und/oder auf dem Ausdruck die Daten. Ein Praxisbeispiel wäre der DXF-Export einer Zeichnung aus AutoCAD in eine Datei und der Import dieser Datei in ein anderes CAD-Programm. Betrachtet man sich die jeweiligen Ausdrücke, ist dies eine Form von Qualitätsbewertung. Allerdings lässt diese zu wünschen übrig, da bei komplexen Modellen der Anwender schnell den Überblick verliert und ihm viele kleinere Fehler wohl eher verborgen bleiben. [Gei01] hat unter diesem Aspekt mehrere CAD-Systeme verglichen und bezüglich der Dateigrößen und Anzahl der Objekte die Kopplungsqualitäten bestimmt.

Eine andere Aposteriori-Variante ist der Dateivergleich. [MHCA06] beschreibt den Vergleich zweier Dateien, die jeweils das gleiche IFC-Modell enthalten sollen. Datei A ist die Originaldatei. Datei B wird in eine Anwendung importiert und durch den Nutzer unverändert wieder in das gleiche Format exportiert. Danach wurden beide Dateien mit einer selbst entwickelten Analysesoftware namens EVASYS verglichen. Schon vor dem genauen Vergleich war durch die unterschiedlichen Dateigrößen ein Qualitätsverlust von A nach B zu erkennen. Dieser sagte allerdings nichts über den exakten Qualitätsverlust aus. Durch EVASYS wurden die eindeutig identifizierbaren Objekte innerhalb der Modelle miteinander verglichen. Dabei wurden Unterschiede in der Menge und in der Attributzusammensetzung der Objekte gefunden. Eine Gesamtbewertung und Wichtung der Fehler wurde dabei nicht vorgenommen.

Das Apriori-Verfahren wird durchgeführt, bevor die Daten umgewandelt werden. In [FFK11] wird eine Variante beschrieben, die als „bewertetes Schema-Mapping“ bezeichnet wird. Ein Schema enthält Datenstrukturen mit ihren Beziehungen untereinander. Ein Schema-Mapping enthält ein Quellschema, ein Zielschema und eine Menge von Beziehungen zwischen diesen beiden Schemata.

Bevor ein Schema-Mapping durchgeführt werden kann, muss geprüft werden, ob die beiden Schemata überhaupt Gemeinsamkeiten haben. Beim sogenannten Schema-Matching können drei Fälle auftreten, wie in Abbildung 1.2 zu sehen ist.

Links werden Schema A und Schema B direkt verglichen. Die Daten und Beziehungen, die in beiden Schemata vorhanden sind, bilden die Schnittmenge $A \cap B$. In der Mitte und rechts wird ein Datenstandard in Form eines dritten Schemas S eingeführt. Je nachdem, wie gut A und B den Standard implementieren, treten weitere Verluste auf (Mitte), so dass $A \cap B \cap S$ übrig bleibt. Wird der Standard vollständig implementiert, entspricht der

1. Stand der Technik

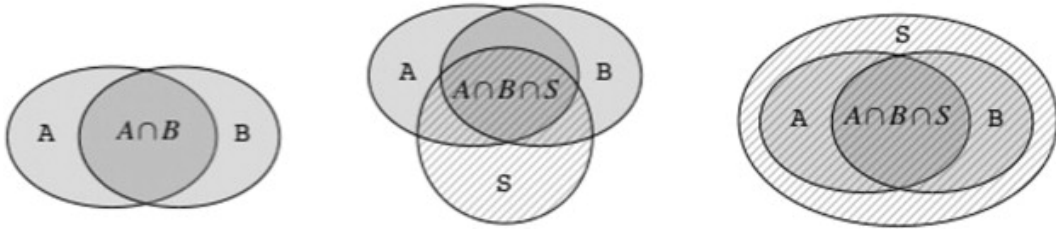


Abb. 1.2.: Schema-Matching. Quelle: [FFK11]

Verlust dem des direkten Schema-Matchings (rechts). Es ist daran zu erkennen, dass die Einführung eines Standards die Menge der zwischen den Anwendungen austauschbaren Daten nicht erhöht.

Im Gegensatz zu Aposteriori-Verfahren wird die Bewertung des Schema-Mappings nicht an den Daten des Modells vollzogen, sondern am Schema, das die Datenstruktur beschreibt.

1.3. Formalisierung des Schema-Mappings

Um die Zusammenhänge formal zu beschreiben, werden als Voraussetzung eine Menge Q definiert, die alle Schemata enthält und eine Menge C , die alle Schema-Kopplungen enthält.

$$Q := \{q | q \text{ ist ein Schema}\}$$

$$C := \{(a, b) \in Q \times Q | \text{Schema } a \text{ wird gekoppelt mit Schema } b\}$$

Wie in Abbildung 1.2 erläutert, gibt es zwei Wege, Schemata zu koppeln. Werden m Schemata direkt gekoppelt, kann die Menge n der Schemakopplungen aus $n = m \cdot (m - 1)$ ermittelt werden. Geschieht der Austausch über ein Standardschema, errechnet sich n aus $n = 2 \cdot m$.

[FFK11] beschreibt ein Beispielszenario (siehe Abbildung 1.3), in dem verschiedene Kopplungsarten verwendet werden. Hier werden Daten von einem Architekten A in einer früheren Planungsphase an einen Ingenieur B übertragen, der in Zusammenarbeit mit einem weiteren Ingenieur C einen beidseitigen Datenaustausch über einen Datenstandard betreibt. In diesem Falle würde die Menge der Schemata $Q = \{a', b', c', s\}$ sein. Die Menge der Kopplungen ergibt $C = \{(a', b'), (b', s), (s, b'), (s, c'), (c', s)\}$.

Ein Schema S wird als Menge von Datenstrukturen beschrieben:

$$S := \{e | e \text{ ist eine Datenstruktur}\} \in Q$$

1. Stand der Technik

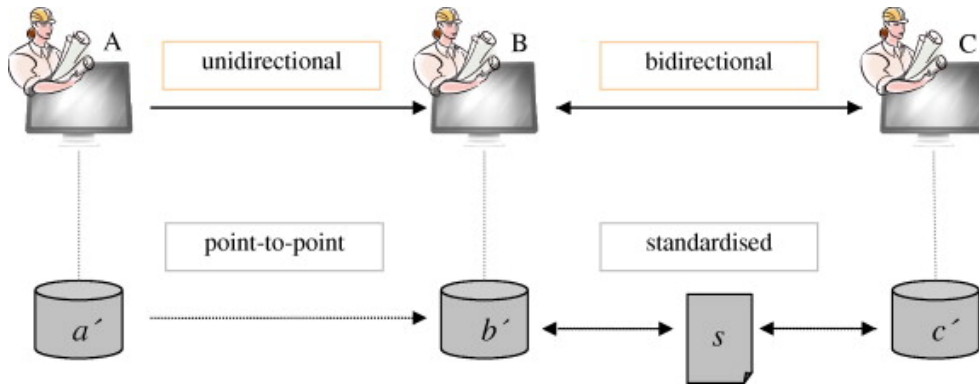


Abb. 1.3.: Beispielszenario mit Kopplungen. Quelle: [FFK11]

Die Beziehungen der Datenstrukturen innerhalb von S bildet eine Potenzmenge:

$$P(S) := \text{ist Potenzmenge von Schema } S \in Q$$

Die Potenzmenge ist die Menge aller Teilmengen von S . Da die leere Menge Teilmenge einer jeden Menge ist, gehört sie auch zu $P(S)$. Für n Elemente, die in S enthalten sind, ergibt sich die Anzahl der in $P(S)$ enthaltenen Elemente aus $|P(S)| = 2^n$.

Das Mapping m lässt sich nun durch eine Funktion beschreiben, die alle Beziehungen von Schema A auf die Beziehungen zu Schema B abbildet:

$$m_{AB} := P(A) \longrightarrow P(B) \text{ mit } A, B \in Q$$

Im Folgenden wird aus dem Kopplungsszenario die Kopplung (c', s) aus der Menge C näher betrachtet. Ein Schema mit seinen Datenstrukturen und Beziehungen lässt sich gut auf objektorientierte Strukturen anwenden. Abbildung 1.4 zeigt die Objektdiagramme der beiden Schemata c' und s , die miteinander gekoppelt werden sollen. Beide enthalten Datenstrukturen in Form von Objekten, ihre Parametern und Beziehungen.

Die Datenstrukturen beschreiben eine Wand, die die Dimensionen l, w und h hat und einen Einfügepunkt mit den kartesischen Koordinaten x, y , und z besitzt. Außerdem verfügt sie über zwei Materialeigenschaften e und k (Elastizitätsmodul und Wärmeleitfähigkeit).

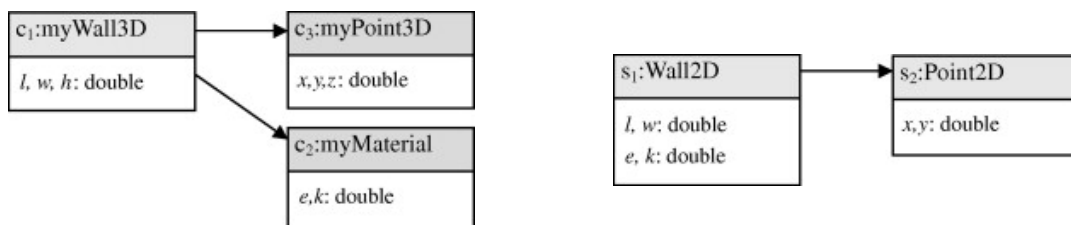


Abb. 1.4.: Schemata c' und s . Quelle: [FFK11]

1. Stand der Technik

Die Beschreibung der enthaltenen Datenstruktur lässt sich folgendermaßen ausdrücken:

$$c' = \left\{ \begin{array}{l} c1 : \text{myWall3D} \\ c2 : \text{myMaterial} \\ c3 : \text{myPoint3D} \end{array} \right\} \in Q \quad s = \left\{ \begin{array}{l} s1 : \text{Wall2D} \\ c2 : \text{Point2D} \end{array} \right\} \in Q$$

Für die beiden Schemata werden die Potenzmengen gebildet und $P(c')$ mittels $m_{c',s}$ auf $P(s)$ abgebildet. Das jeweilige Mapping kann in fünf Fälle, sogenannte Mapping Patterns, eingeteilt werden, die in Abbildung 1.5 dargestellt sind.

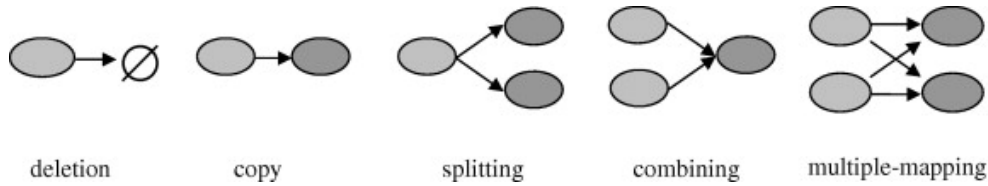


Abb. 1.5.: Mapping Patterns beim Schema-Mapping. Quelle: [FFK11]

Das Schema-Mapping und die Anwendung der Mapping Patterns sind in Abbildung 1.6 zu sehen.

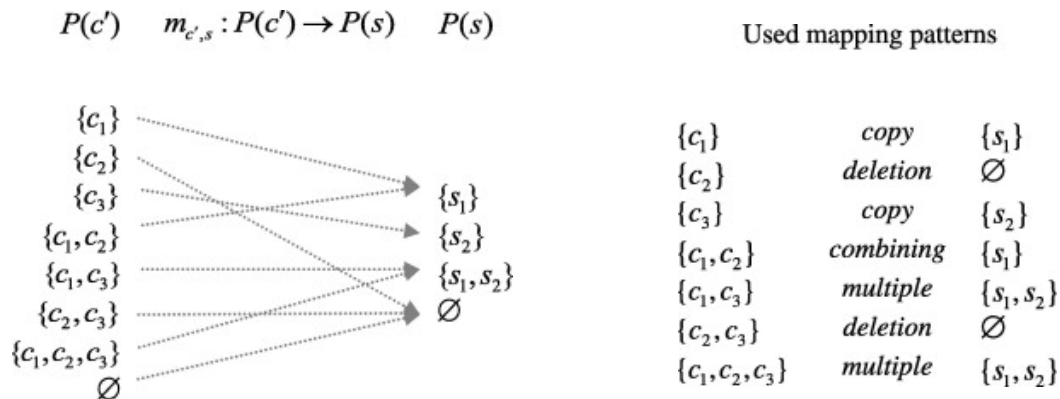


Abb. 1.6.: Schema-Mapping von c' nach s . Quelle: [FFK11]

1.4. Qualitative Bewertung

Wenn Informationen informationstechnisch verarbeitet werden sollen, müssen sie je nach Aufgabengebiet zu Datenstrukturen abstrahiert werden. Andersherum ist es nötig, die Daten wieder zu interpretieren, sollen sie in der Praxis genutzt werden. Diese Transformationen beherbergen immer ein Risiko, dass dabei Informationen verloren gehen.

Als Fehlerquelle kommen verschiedene Ursachen in Frage. Bei Anwendungen, in denen große Mengen an Dezimalzahlen verwendet werden, treten numerische und Näherungsfehler auf.

1. Stand der Technik

Numerische Fehler können zum Beispiel Rundungsfehler sein, wie sie auftreten, wenn Ganzzahlenwerte als Dezimalwert abgelegt werden oder wenn mathematisch exakte Formulierungen durch einen Computer dargestellt werden.

Näherungsfehler treten zum Beispiel auf, wenn Datenstrukturen durch verschiedene Modelle am Computer realisiert werden. Dazu gehören die Darstellung von gekrümmten Oberflächennetzen, wo der Fehler je nach Feinheit und Vernetzungsalgorithmus variiert und die Erzeugung von Farben aus unterschiedlichen Farbsystemen (CMYK und RGB).

[FFK11] beschreibt in einem Beispiel die Qualitätsbewertung einer Stütze anhand der Geometrie. Die Stütze mit dem Volumen V_1 wird von einem Modeller A geometrisch genau beschrieben. Modeller B, zu dem die Daten übertragen werden sollen, kann aber nur Stützen mit einer n-eckigen Grundfläche und dem Volumen V_2 darstellen (siehe Abb. 1.7). Um die Qualität der Kopplung zwischen Modeller A und B zu ermitteln, wird daher erst die Volumendifferenz δ_v zwischen den Volumina der Stützen errechnet. Der Fehler ermittelt sich dann aus:

$$\varepsilon = \frac{\delta_v}{V_1}$$

Daraus ergibt sich eine Qualität von:

$$Q = 1 - \varepsilon$$

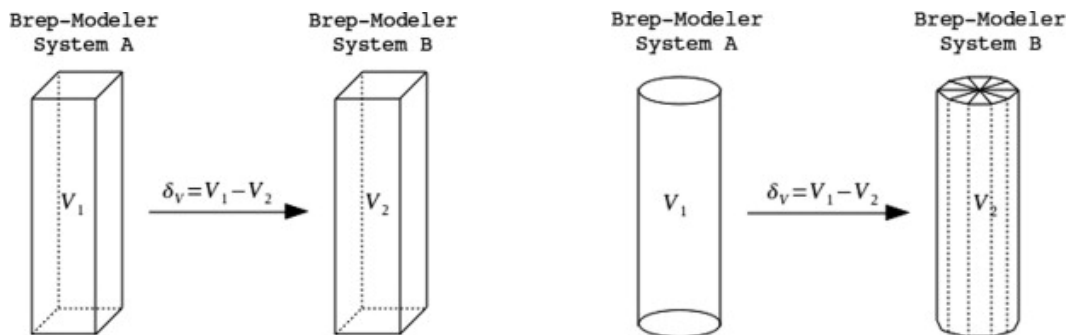


Abb. 1.7.: Übertragung der Stützen von Modeller A zu Modeller B. Quelle: [FFK11]

Im Falle einer rechteckigen Stütze ist die Differenz $\delta_v = 0$. Dementsprechend ist auch der Fehler $\varepsilon = 0$. Daraus resultiert eine Qualität $Q = 1$, was bedeutet, dass die Stütze in beiden System exakt gleich dargestellt wird.

Wird die Stütze von Modeller A dagegen rund modelliert, ergibt sich eine Volumendifferenz, die nicht mehr Null sein kann. Der Modeller B kann die Grundfläche der Stütze nicht exakt kreisrund darstellen und behilft sich mit der Näherung durch die Modellierung der Grundfläche durch ein N-Eck. Die Volumendifferenz δ_v und letztendlich der

1. Stand der Technik

Fehler ε hängen von der Anzahl der Ecken n ab und berechnen sich folgendermaßen:

$$\delta_v = \left| \pi \cdot r^2 \cdot h - \frac{r^2 \cdot n}{2} \cdot \sin \frac{360}{n} \cdot h \right| \quad \varepsilon = 1 - \left| \frac{\frac{n}{2} \cdot \sin \frac{360}{n}}{\pi} \right|$$

Würde nun eine Stütze mit rundem Querschnitt übertragen werden und der Modeller B den Kreisquerschnitt durch ein N-Eck mit einer Eckenanzahl von $n = 20$ näherungsweise beschreiben, würde sich ein Fehler $\varepsilon = 0.016$ ergeben, woraus eine Qualität $Q = 0.984$ resultierte.

1.5. Bewertetes Schema-Mapping

Für die Bewertung wird eine Menge R definiert, die durch einen Dezimalwert zwischen 0 und 1 ausdrückt, wie gut ein Mapping zwischen zwei Datenstrukturen ist. R wird in [FFK11] als Untermenge der Reellen Zahlen im Intervall $[0,1]$ definiert¹. Ein Wert von 1 bedeutet kein Informationsverlust, alle Werte darunter drücken eine Unvollständigkeit der Informationen aus.

$$R := \{r \in \mathbb{R} \mid 0 < r \leq 1\}$$

Das bereits bekannte Schema-Mapping $(a, b) \in m_{A,B}$ zweier Schemata $A, B \in Q$ wird durch einen Qualitätsfaktor $r \in R$ zu einem bewerteten Schema-Mapping $\bar{m}_{A,B}$ erweitert.

$$\bar{m}_{A,B} : m_{A,B} \longrightarrow R$$

Dies definiert die Qualität auf Schema-Level. Ein Modell nach einem Schema hat aber immer eine spezifische Zusammensetzung von Instanzen. Der Begriff Instanz stammt aus der objektorientierten Programmierung. Zum Verständnis wird noch einmal die Abbildung 1.4 herangezogen. Dort existiert eine „Vorlage“ einer Datenstruktur c1 mit der Bezeichnung myWall3D. Von dieser Vorlage werden die eigentlich verwendeten Instanzen erzeugt. Sie besitzen dieselbe Struktur, ihre Parameter sind aber mit individuellen Werten belegt. Daher ist eine weitergehende Qualitätsbewertung auf Instanzenebene nötig. Von den auszutauschenden Instanzen wird eine Instanzmenge I_A angelegt:

$$I_A := \{i \mid i \text{ ist eine Instanz einer Datenstruktur } e \in A \in Q\}$$

Die Beziehungen der Instanzen $i \in I_A$ folgen exakt den Beziehungen, die im Schema $A \in Q$ definiert sind. Daraus folgt, dass die Instanzenbeziehungen ebenso durch eine

¹Es wird aber darauf hingewiesen, dass R nicht zwingend numerisch definiert sein muss, es ist auch eine linguistische Beschreibung denkbar.

1. Stand der Technik

Potenzmenge beschrieben werden können:

$$type := P(I_A) \longrightarrow P(A)$$

Die Qualität des Mappings der Instanzenmenge I_A des Schemas $A \in Q$ zum Schema $B \in Q$ kann über $\bar{m}_{A,B}$ berechnet werden, wobei $n_{type(i)}$ die Anzahl der Instanzen eines Types $i \in I_A$ ist.

$$Quality := f(\bar{m}_{A,B}, I_A) := \sum_{i \in P(I_A)} \sum_{j \in P(B)} \frac{\bar{m}_{A,B}(type(i), j)}{n_{type(i)}}$$

Die bewertete Mapping-Funktion $\bar{m}_{A,B}$ liefert einen Qualitätswert für das Mapping einer Instanz i , erzeugt auf der Basis eines spezifischen Datentyps $type(i)$ des Ausgangsschemas A zum zugehörigen Schema-Element $j \in P(B)$ des Zielschemas B .

Der letztendliche Qualitätswert eines bewerteten Mappings $(a, b) \in \bar{m}_{A,B}$ wird durch die Übertragbarkeit seiner Attribute bestimmt. So ist zum Beispiel die Qualität des Mappings von $c_1 : myWall3D$ auf $s_1 : Wall2D$:

$$Q_{C_1, S_1} = \frac{\bar{m}_{A,B}(l, l) + \bar{m}_{A,B}(w, w) + \bar{m}_{A,B}(h, \emptyset)}{1 + 1 + 1} = \frac{1.0 + 1.0 + 0.0}{3} \approx 0.7$$

Im Prinzip lässt sich der Prozess auf die Attribute innerhalb einer Datenstruktur genauso anwenden, wie es mit den Datenstrukturen innerhalb eines Schemas gemacht wurde. Auf diese Weise lässt sich eine Mapping-Matrix aufstellen, zu sehen in Abbildung 1.8, wo dies in beiden Richtungen durchgeführt wurde.

$\bar{m}_{c',s}$	$\{s_1\}$	$\{s_2\}$	$\{s_1, s_2\}$	$\emptyset$
$\{c_1\}$	0.7	0	0	0
$\{c_2\}$	0	0	0	1
$\{c_3\}$	0	0.7	0	0
$\{c_1, c_2\}$	1.0	0	0	0
$\{c_1, c_3\}$	0	0	1.0	0
$\{c_2, c_3\}$	0	0	0	1
$\{c_1, c_2, c_3\}$	0	0	1.0	0
$\emptyset$	0	0	0	1

$\bar{m}_{s,c'}$	$\{c_1\}$	$\{c_2\}$	$\{c_3\}$	$\{c_1, c_2\}$	$\{c_1, c_3\}$	$\{c_2, c_3\}$	$\{c_1, c_2, c_3\}$	$\emptyset$
$\{s_1\}$	0	0	0	1.0	0	0	0	0
$\{s_2\}$	0	0	1.0	0	0	0	0	0
$\{s_1, s_2\}$	0	0	0	0	0	0	1.0	0
$\emptyset$	0	0	0	0	0	0	0	1

Abb. 1.8.: Matrizen der Schema-Mappings $\bar{m}_{C',S}$ und $\bar{m}_{S,C'}$. Quelle: [FFK11]

Liegt nun ein konkretes Modell entsprechend einem Schema mit Instanzen vor, wie es ein Beispiel in Abb. 1.9 zeigt, lässt sich endgültig die Gesamtqualität eines Mappings mit

1. Stand der Technik

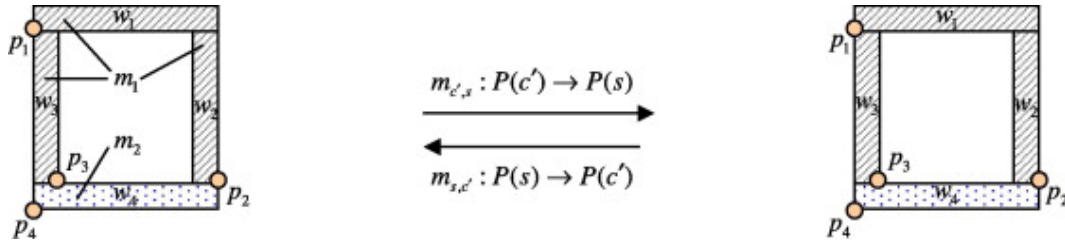


Abb. 1.9.: Mapping mit mehreren Instanzen. Quelle: [FFK11]

diesem Modell bestimmen:

$$Q_{C',S} = \frac{4 \cdot \bar{m}(\{c_1\}, \{s_1\}) + 4 \cdot \bar{m}(\{c_3\}, \{s_2\}) + 2 \cdot \bar{m}(\{c_2\}, \{\emptyset\}) + 4 \cdot \bar{m}(\{c_1, c_2, c_3\}, \{s_1, s_2\})}{4 + 4 + 2 + 4} = 0.686$$

$$Q_{S,C'} = \frac{4 \cdot \bar{m}(\{s_1\}, \{c_1, c_2\}) + 4 \cdot \bar{m}(\{s_2\}, \{c_3\}) + 4 \cdot \bar{m}(\{s_1, s_2\}, \{c_1, c_2, c_3\})}{4 + 4 + 4} = 1.0$$

Damit wurde die Gesamtqualität bestimmt.

2. Das Referenzobjekt

2.1. Hintergrund

Da diese Arbeit im Rahmen des Graduiertenkollegs angefertigt wird, wurde als Referenzobjekt ein Bauwerk gewählt, das von den Mitarbeitern des Kollegs entworfen und für eigene Untersuchungen verwendet wurde. Der Aufbau wurde aus [Reu11] entnommen.

2.2. Konstruktion

Das Referenzobjekt stellt ein fünfgeschossiges Skeletttragwerk dar. Die primären Strukturelemente sind Stützen und Balken aus Stahl S235. Für die Stützen wurde Profile vom Typ IPB 240 verwendet, für die Mittelbalken IPB 200 und für die restlichen Balken IPE 200. Als Sekundärelemente wurden vertikale Aussteifungen in zwei möglichen Varianten umgesetzt: Einmal als Zugstäbe aus Stahl S235 und einmal als Scheiben aus Mauerwerk bzw. Beton mit einer Dicke von 10 Zentimetern. Weiterhin wurden Geschossdecken eingebaut, deren Existenz in der Quelle allerdings nur angedeutet wurde. Sie besitzen hier eine Dicke von 10 Zentimetern.

Der Rasterabstand zwischen den Stützen beträgt 6 Meter, so dass das Gebäude 6 Meter breit und 12 Meter lang ist. Die Geschosshöhe des Erdgeschosses beträgt 5.10 Meter, die der Obergeschosse jeweils 3.50 Meter. Das Gebäude ist über die sechs Stützen im Boden eingespannt.

Abbildung 2.1 zeigt Ansichten und Grundriß der Variante (a) mit Aussteifung durch Zugstäbe und der Variante (b) mit Wandaussteifung. Die Bemaßung stellt die Systemmaße dar.

2. Das Referenzobjekt

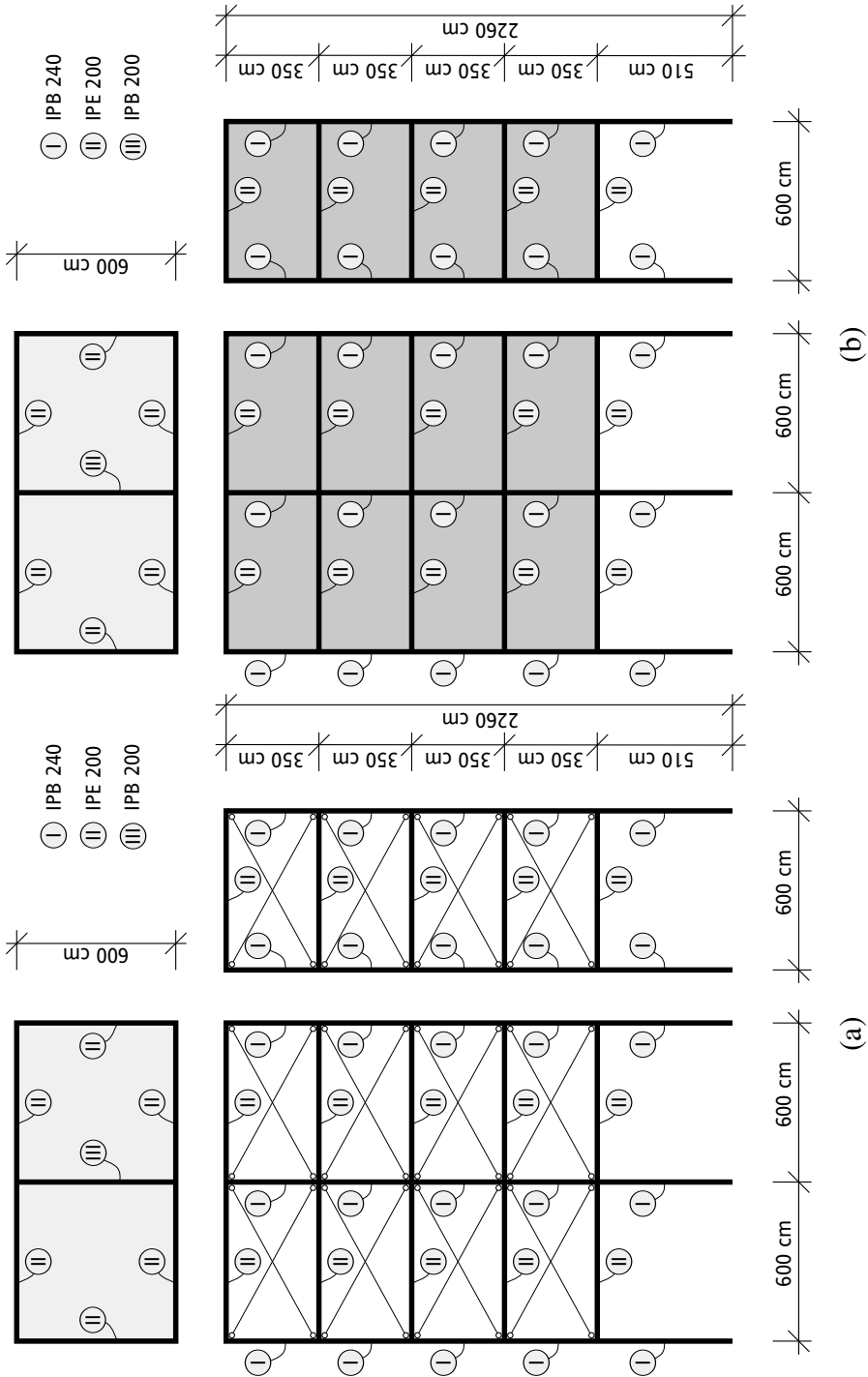


Abb. 2.1.: Referenzobjekt. Quelle [Reu11]

3. Die Industry Foundation Classes

3.1. Überblick

Im Jahre 1995 gründeten mehrere amerikanische Firmen unter der Federführung von Autodesk nach einjähriger Entwicklungsarbeit die *International Alliance for Interoperability*, kurz IAI, um einen Standard für Software-Interoperabilität zu entwickeln. Ziel war es, die Projektabwicklung mittels effizienter Methoden integrierter Informationsverarbeitung durchgängiger und effektiver zu gestalten. 2005 benannte sich die IAI in *buildingSMART* um. Im Mittelpunkt steht seitdem Building Information Modeling (BIM) als neue Planungsmethode auf der Basis digitaler Bauwerksmodelle. Für eine offene BIM-Anwendung konzipiert und zertifiziert buildingSMART Standards wie IFC, entwickelt Anforderungsprofile für neue Prozesse, Rollen und Leistungsbilder und definiert Bildungsstandards. [Bui11].

Um die Struktur der IFC festzulegen, wurde die Definitionssprache *EXPRESS* entwickelt. Mit ihr können Datenstrukturen aufgebaut, miteinander verknüpft und Beziehungen zwischen ihnen festgelegt werden. Des Weiteren sind Regeln und Funktionen definierbar, um eine gewisse Logik in der Gültigkeit beim Aufbau der Datenstrukturen zu erzeugen.

Zur Unterstützung entwickelte man eine grafische Notation namens *EXPRESS-G*. Mit ihr können die Datenstrukturen und deren Beziehungen anschaulich dargestellt werden. Einige Abbildungen in dieser Arbeit enthalten Diagramme in dieser Notation. Zum besseren Verständnis wurde eine Kurzübersicht der Symbole in den Anhang A der Arbeit eingefügt.

Um die nach diesem *EXPRESS*-Schema aufgebauten Datenstrukturen auszutauschen, war es notwendig, ein einheitliches Austauschformat zu entwickeln. Dies wurde mit dem *STEP*-Format realisiert. *STEP* steht für *STandard for the Exchange of Product model data*. Die Standardform ist textbasiert und gut lesbar. Es existiert noch eine weitere, XML-basierte Form und eine platzsparende, im ZIP-Format gepackte Version. Das *STEP*-Format wurde durch ISO 10303 zertifiziert. Somit ist eine Standardisierung dieses Datenaustauschformates gegeben.

Die IFC wurden letztendlich in einem *EXPRESS*-Schema verfasst und die in diesem Schema vorliegende Bauwerksmodelle können im *STEP*-Format ausgetauscht werden.

3. Die Industry Foundation Classes

Im Zeitraum der Entstehung dieser Arbeit waren die IFC in der Version 2x3 die aktuellste Releaseversion, die von verschiedenen Programmen gelesen und geschrieben werden kann. Momentan in der Entwicklung befindet sich die Version 2x4, die sich noch im Release-Candidate-Status befindet.

3.2. Die EXPRESS-Definition

EXPRESS gehört zu den konzeptionellen Schema-Sprachen. Sie beschreibt Objektklassen mit ihren Attributen und Einschränkungen, definiert Beziehungen zwischen den Klassen und legt Regeln für den Wertebereich der Attribute und Gültigkeit der Klassen fest. Weiterhin können mit EXPRESS eigene Datentypen erzeugt werden, sowie Funktionen und allgemein anwendbare Regeln. Objektklassen werden im EXPRESS-Kontext als Entities bezeichnet.

Wie bei vielen Programmiersprachen wird auch bei EXPRESS ein Ausdruck mit einem Semikolon abgeschlossen. Zeilenumbrüche und Leerzeichen dienen der besseren Lesbarkeit, spielen aber für das Parsen durch Programme keine Rolle. Das komplette Schema einer EXPRESS-Datei wird durch die Schlüsselwörter SCHEMA und END_SCHEMA eingebunden, wobei dem Startschlüsselwort als Parameter die Bezeichnung bzw. Version des Schemas mitgegeben wird.

```
SCHEMA IFC2X3;  
...  
END_SCHEMA;
```

3.2.1. Basisdatentypen

Quasi vordefiniert sind Datentypen, die eine einzelne Information enthalten können. EXPRESS bietet folgende Basisdatentypen an:

INTEGER	Ein ganzzahliger Wert
REAL	Eine Dezimalzahl, die ein Komma enthalten muss
NUMBER	Entweder eine Ganzzahl oder eine Dezimalzahl
BOOLEAN	Ein logischer Wert, der entweder wahr (TRUE) oder falsch (FALSE) sein kann
LOGICAL	Ein logischer Wert, der zusätzlich zu wahr und falsch auch unbekannt (UNKNOWN) sein darf
STRING	Eine Zeichenkette mit optional angegebener Länge
BINARY	Eine Sequenz von Binärwerten mit angegebener Länge

3.2.2. Deklarationen von Datentypen

In EXPRESS lassen sich eigene Datentypen deklarieren. Die Deklaration muss mit dem Schlüsselwort `TYPE` beginnen, enthält den Bezeichner und den Basisdatentyp und muss mit dem Kommando, das das Schlüsselwort `END_TYPE` beinhaltet, abgeschlossen werden. Die Syntax für die Definition eines Dezimalzahlendatentypes sieht demnach wie folgt aus:

```
TYPE IfcAreaMeasure = REAL;  
END_TYPE;
```

Eine weitere Möglichkeit sind Aufzählungen. Als Typ wird in diesem Fall das Schlüsselwort `ENUMERATION` verwendet, gefolgt von einer Liste möglicher Werte, die der Datentyp annehmen kann. Dabei handelt es sich genaugenommen um Zeichenketten. Hier ein Syntaxbeispiel für einen derartigen Datentyp:

```
TYPE IfcArithmeticOperatorEnum = ENUMERATION OF  
    (ADD,DIVIDE,MULTIPLY,SUBTRACT);  
END_TYPE;
```

Die dritte Variante verwendet den `SELECT`-Typen. Auch hier wird aus einer Liste ein Wert ausgewählt, wobei es sich in diesem Fall um den Namen eines Datentypes oder eines Entities, also einer Objektklasse, handelt. Es wird also eine Art Referenz erzeugt. Dazu sei folgendes Beispiel angegeben:

```
TYPE IfcActorSelect = SELECT  
    (IfcOrganization,IfcPerson,IfcPersonAndOrganization);  
END_TYPE;
```

3.2.3. Entities

Ein Entity stellt eine Objektklasse in EXPRESS dar. Ähnlich wie in einer objektorientierten Programmiersprache besitzen Entities Attribute und können sich durch Vererbung voneinander ableiten. Mit den Schlüsselwörtern `ENTITY` und `END_ENTITY` wird die Klasse definiert.

```
ENTITY IfcDateAndTime;  
    ...  
END_ENTITY;
```

Oft ist es nötig, spezialisierte Klassen zu definieren, die durch Vererbung Attribute ihrer Elternklassen erwerben. Die Elternklasse zeigt durch das Schlüsselwort `SUPERTYPE`

3. Die Industry Foundation Classes

OF und einer Liste an, an welche Unterklassen sie ihre Attribute vererbt. Wird der Bezeichner ABSTRACT dem SUPERTYPE noch vorangestellt, kann die Elternklasse nicht instanziiert werden, es dürfen also keine Objekte der Elternklasse erzeugt werden. Im Gegenzug muss in den Kinderklassen das Schlüsselwort SUBTYPE OF mit dem Namen der Elternklasse stehen.

```
ENTITY IfcConic
  ABSTRACT SUPERTYPE OF (ONEOF
    (IfcCircle,IfcEllipse))
  ...
END_ENTITY;
```

```
ENTITY IfcCircle
  SUBTYPE OF (IfcConic);
  ...
END_ENTITY;
```

Grundsätzlich ist es in EXPRESS möglich, von mehreren Elternklassen zu erben. Für die IFC wurde dies aber laut [Wix00b] ausgeschlossen, Mehrfachvererbung kommt dort nicht zum Einsatz.

3.2.4. Attribute

Um eine Klasse mit Daten zu füllen, sind Attribute notwendig. Ein Attribut steht in einer Beziehung zu seiner Klasse. Im einfachsten Falle bedeutet dies, dass die Klasse ein Attribut mit einem bestimmten Bezeichner besitzt und dieses Attribut genau eine Instanz eines anderen Entities oder genau einen Wert eines Datentypes besitzt. Im folgenden Beispiel sind sowohl IfcCalendarDate als auch IfcLocalTime selbst Klassen mit eigenen Attributen.

```
ENTITY IfcDateAndTime;
  DateComponent : IfcCalendarDate;
  TimeComponent : IfcLocalTime;
END_ENTITY;
```

Es kommt vor, dass die Angabe von Attributen nicht zwingend notwendig ist. Dafür gibt es das Schlüsselwort OPTIONAL, das dem Datentyp vorangestellt werden muss.

3. Die Industry Foundation Classes

```
ENTITY IfcDistributionPort
...
FlowDirection : OPTIONAL IfcFlowDirectionEnum;
END_ENTITY;
```

Weiterhin lassen sich in EXPRESS Aggregationsbeziehungen formulieren. Unter einer Aggregation versteht man die Zusammenfassung mehrerer gleichartiger Objekte zu einer nach außen hin geschlossenen Einheit. Beispielsweise bildet ein Gebäude die Aggregation seiner Stockwerke [Fli03]. EXPRESS unterstützt vier Arten von Aggregationen:

ARRAY	Eine Kollektion von Objekten fester Größe und Reihenfolge
BAG	Eine ungeordnete Kollektion von Objekten, die auch Duplikate enthalten darf
LIST	Eine geordnete Kollektion von Objekten mit fester Reihenfolge
SET	eine ungeordnete Kollektion von Objekten ohne Duplikate

Eine Besonderheit bilden inverse Attribute. Damit kann eine Beziehung in umgekehrter Reihenfolge nachvollzogen werden. Ein inverses Attribut wird mit dem Präfix INVERSE versehen. Im folgenden Beispiel greift das inverse Attribut des Entities IfcDocumentReference auf ein Attribut der Klasse IfcDocumentInformation zu.

```
ENTITY IfcDocumentInformation;
...
DocumentReferences : OPTIONAL SET [1:?] OF IfcDocumentReference;
...
END_ENTITY;
```

```
ENTITY IfcDocumentReference
...
INVERSE
ReferenceToDocument : SET [0:1] OF IfcDocumentInformation
FOR DocumentReferences;
...
END_ENTITY;
```

3.2.5. Weitere lokale Regeln

Unique-Regel

Manchmal soll sichergestellt werden, dass eine Klasse ein Attribut mit einem einzigartigen Wert enthält, den keine andere Klasse besitzen soll. Das Schlüsselwort UNIQUE

3. Die Industry Foundation Classes

zeigt diesen Bedarf an. Es muss programmiertechnisch sichergestellt werden, dass diese Regel eingehalten wird.

```
ENTITY IfcRoot
...
UNIQUE
  UR1 : GlobalId;
END_ENTITY;
```

Derive-Regel

Attribute, die dem Schlüsselwort DERIVE folgen, sind abgeleitete Attribute, die sich meistens aus anderen Attributen der Klasse errechnen lassen und auf diese Art einfach zurückgegeben werden können.

```
ENTITY IfcSweptSurface
...
  Position : IfcAxis2Placement3D;
DERIVE
  Dim : IfcDimensionCount := Position.Dim;
...
END_ENTITY;
```

Domain-Regel

Mit Domain-Regeln definiert man Wertebereiche von Attributen. Das Schlüsselwort lautet WHERE, und für jedes Attribut, das diesem Schlüsselwort unterstellt ist, gibt es eine Syntax mit logischen und/oder arithmetischen Operatoren, die den Wertebereich festlegt. Domain-Regeln sind die einzigen Regeln, die auch in Typdefinitionen auftreten können.

```
ENTITY IfcCurveStyleFontPattern;
  VisibleSegmentLength : IfcLengthMeasure;
  InvisibleSegmentLength : IfcPositiveLengthMeasure;
WHERE
  WR01 : VisibleSegmentLength >= 0.;
END_ENTITY;
```

3.2.6. Globale Funktionen

Manche Domain-Regeln erfordern eine etwas größere Menge an Quellcode als nur eine Zeile. In diesem Fall besteht die Möglichkeit, komplexeren Code in eigenständigen Funktionen unterzubringen, die aus der Regel heraus aufgerufen werden. Dies wäre auch

3. Die Industry Foundation Classes

vorteilhaft, sollte die Funktion in Regeln verschiedener Klassen verwendet werden. Die Verwendung von Funktionen soll an dieser Stelle nur erwähnt werden, auf die komplette Syntax wird in diesem Zusammenhang nicht weiter eingegangen.

```
ENTITY IfcLocalTime;  
    ...  
WHERE  
    WR21 : IfcValidTime (SELF);  
END_ENTITY;  
  
FUNCTION IfcValidTime (Time: IfcLocalTime) : BOOLEAN;  
    IF EXISTS (Time.SecondComponent) THEN  
        RETURN (EXISTS (Time.MinuteComponent));  
    ELSE  
        RETURN (TRUE);  
    END_IF;  
END_FUNCTION;
```

3.2.7. Verknüpfung mit anderen Schemata

Klassen, die in einem anderen Schema definiert wurden, lassen sich über ein EXPRESS-Interface ansprechen. Das Schlüsselwort dafür lautet REFERENCE FROM. Man kann die Klassen auf diese Weise in seinem eigenen Schema nutzen, ohne dass diese dort selbst definiert wurden.

```
REFERENCE FROM IfcGeometryResource  
    (IfcCartesianPoint, IfcBounded_curve,  
     IfcPolyline, IfcTrimmed_curve,  
     IfcCompositeCurve, IfcPlacement,  
     IfcLine, IfcConic,  
     IfcCircle, IfcEllipse));
```

3.3. Aufbau der IFC

3.3.1. Grundlegendes

Die IFC sind ein Schema, welches im EXPRESS-Format vorliegt. Es wurde aus Elementen erzeugt, die in 3.2 beschrieben wurden. Softwaretechnik ausgedrückt bedeutet das: EXPRESS definiert die Syntax, IFC dagegen beschreibt die Semantik. Mit dem Schema

3. Die Industry Foundation Classes

der IFC soll ein Bauwerk über dessen gesamten Lebenszyklus abgebildet werden. Die Struktur ist modular aufgebaut, einzelne Komponenten sind wiederverwendbar. Somit lassen sich die IFC leichter pflegen und erweitern. Durch die Beibehaltung der essentiellen Grundstruktur soll zudem eine Aufwärtskompatibilität innerhalb einer Majorversion angestrebt werden¹.

Eine Grundidee der IFC ist es, Details mit einem hohen Abstraktionsgrad abzubilden. Das bedeutet, dass der Aufbau der IFC stark gegliedert ist, um eine möglichst große Flexibilität in der Beschreibung von Objekteigenschaften zu erreichen. Dadurch soll die Konsistenz des Modells gewahrt und Redundanzfreiheit angestrebt werden.

3.3.2. Gesamtarchitektur

Die IFC werden in vier Ebenen, die sogenannten *Layer*, eingeteilt. Der Zugriff der Ebenen untereinander funktioniert nach dem Gravitationsprinzip. Das heißt, eine Ebene kann nur auf seine eigenen Informationen und die aller unter ihr angeordneten Ebenen zugreifen.

Die unterste Ebene bildet der *Resource Layer*. In ihm befinden sich grundlegende und allgemeingültige Informationen. Beispiele dafür wären Datum, Zeit, Materialien, Geometrie oder Bemaßung.

Die nächsthöhere Ebene ist der *Core Layer*. Diese enthält den Kern (*Core*) mit seinen Erweiterungen (*Core Extensions*). Der Kern stellt grundlegende Funktionalitäten bereit, um Objekte und ihre Beziehungen zueinander zu beschreiben. Spezifische Klassen, die in Layern weiter oben definiert werden, haben hier ihre Wurzeln. Objekte bekommen eine eindeutige ID, eine Nutzerhistorie, können Eigenschaftssets erhalten, es können ihnen Typdefinitionen zugeordnet werden und mehr. Die Kern-Erweiterungen spezifizieren die im Kern definierten Klassen noch weiter. Weitere Klassen verfeinern die Kernklassen ihrem Metier entsprechend. Beispielsweise stellt die *Product Extension* unter anderem Klassen zur hierarchischen Gebäudeunterteilung sowie zur groben Einteilung in Bauelemente zur Verfügung.

Die dritte Ebene ist der *Interoperability Layer*. Sie enthält Klassen und Typen, die mehreren Domänen² zugeordnet werden können. Wie der Name schon sagt, wird damit die Interoperabilität zwischen unterschiedlichen Domänen gewährleistet. Die Klassen der *Shared Building Elements* seien an dieser Stelle hervorgehoben, da sie mit Elementen wie *IfcColumn* oder *IfcSlab* auch die Klassen enthalten, die Ausgangspunkte für Schemaanalyse und -mapping sind.

Die oberste Ebene ist der *Domain Layer*. Diese Ebene enthält die unterschiedlichen Fachgebiete, die in einem Bauwerksinformationsmodell vertreten sind. Sie werden im Kon-

¹Zum Beispiel von Version 2x3 auf 2x4.

²Eine Domäne ist als ein Fachgebiet anzusehen. Siehe dazu den folgenden Absatz zum Domain Layer.

3. Die Industry Foundation Classes

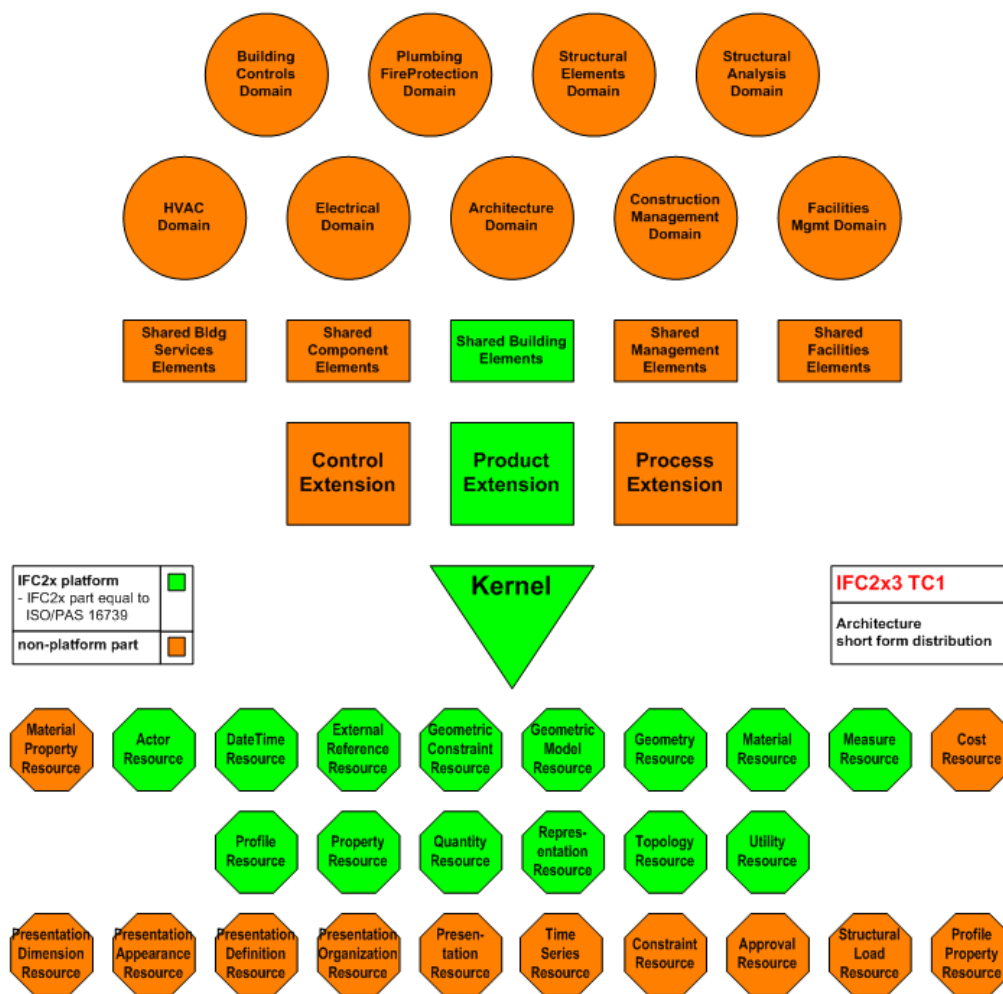


Abb. 3.1.: Layerstruktur der IFC. Quelle: [MSG07]

text der IFC Domänen genannt. Es gibt Domänen wie z.B. die *Architectural Domain*, die *Structural Analysis Domain* oder die *Facilities Management Domain*. Klassen dieser Domänen bilden in der Regel die instanziierten Klassen, vergleichbar mit den Blättern eines Baumes, der seine Wurzeln im Core Layer hat.

3.3.3. Projektcontainer und Grundeinstellungen

Jedes IFC-File besitzt genau eine Instanz des Objekts *IfcProject*. Auf diese Instanz beziehen sich direkt oder indirekt alle anderen Objekte, insbesondere gebäude- und bauteilrelevante Klassen. Dem Projektcontainer ist genau eine Instanz entweder vom Typ *IfcSite*, einer Baufläche, oder vom Typ *IfcBuilding*, einem Gebäude, zugeordnet, die das übergeordnete räumliche Element in diesem Projekt darstellt. Die Zuordnung erfolgt nicht als direktes Attribut von *IfcProject*, sondern über eine Beziehungsklasse namens *IfcRelAggre*

3. Die Industry Foundation Classes

gates. Beziehungsklassen verknüpfen andere Klassen miteinander.

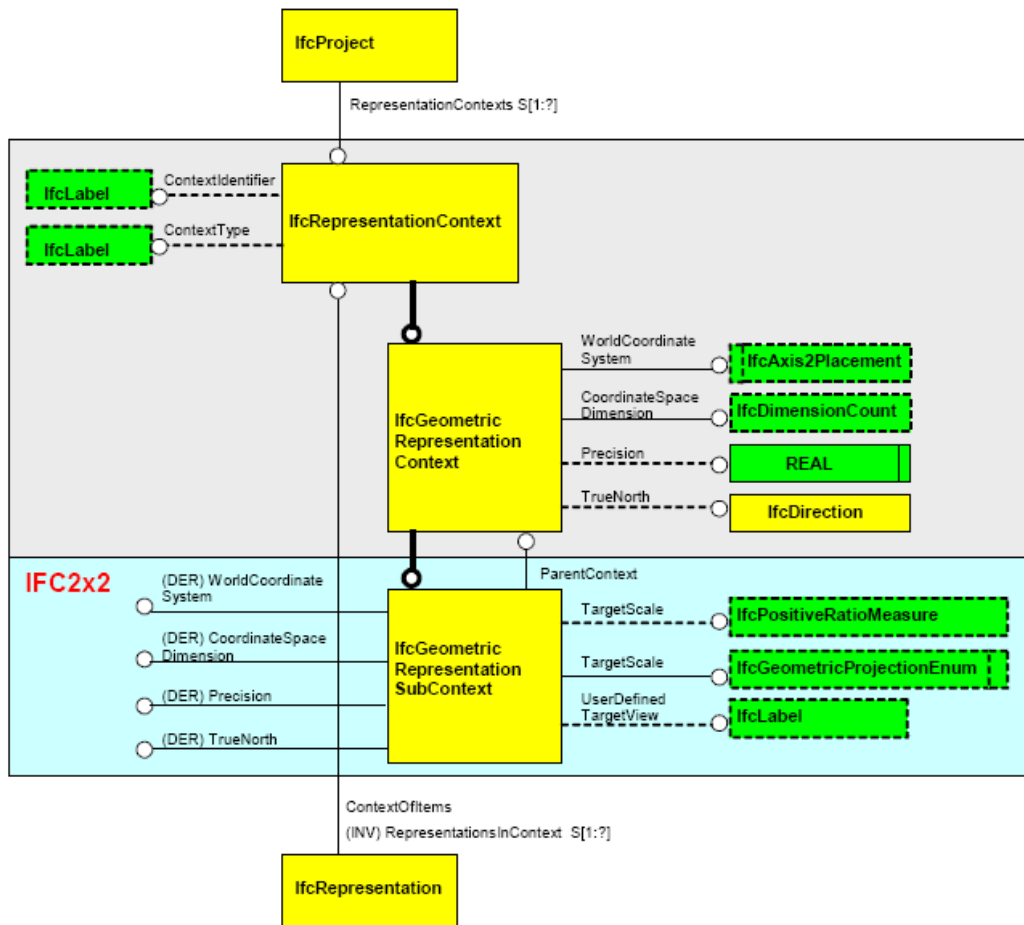


Abb. 3.2.: Struktur des IfcGeometricRepresentationContext. Quelle: [Lie09]

Weiterhin wird werden dem Projekt eine eindeutige ID, einen Namen, die Bezeichnung der aktuellen Projektphase, eine Nutzerhistorie und ein Einheitensystem zugewiesen. Außerdem bekommt das Projekt eine Referenz auf mindestens einen geometrischen Kontext. Im diesem sind Attribute wie Weltkoordinatensystem, Dimension, Genauigkeit der Einheiten sowie Nordausrichtung definiert.

Es gibt verschiedene Typen von geometrischen Kontexten. Man spricht in diesem Zusammenhang auch von *Views*. Der Hintergrund des Konzeptes von multiplen Views ist die Idee, ein Modell je nach Bedarf nach bestimmten Kriterien darzustellen. Zum Beispiel muss eine Entwurfszeichnung nicht so komplex sein wie eine Ausführungszeichnung.

3.3.4. Räumliche Struktur

Ein Projekt ist in handhabbare, räumlich abgrenzbare Untereinheiten aufgeteilt. Diese Einheiten stammen alle von der Elternklasse *IfcSpatialStructureElement* ab:

3. Die Industry Foundation Classes

- *IfcSite* stellt die Baufläche dar
- *IfcBuilding* steht für ein Gebäude
- *IfcBuildingStorey* ist die Klasse für ein Geschoss innerhalb eines Gebäudes
- *IfcSpace* wird für einen Raum innerhalb eines Gebäudegeschosses verwendet

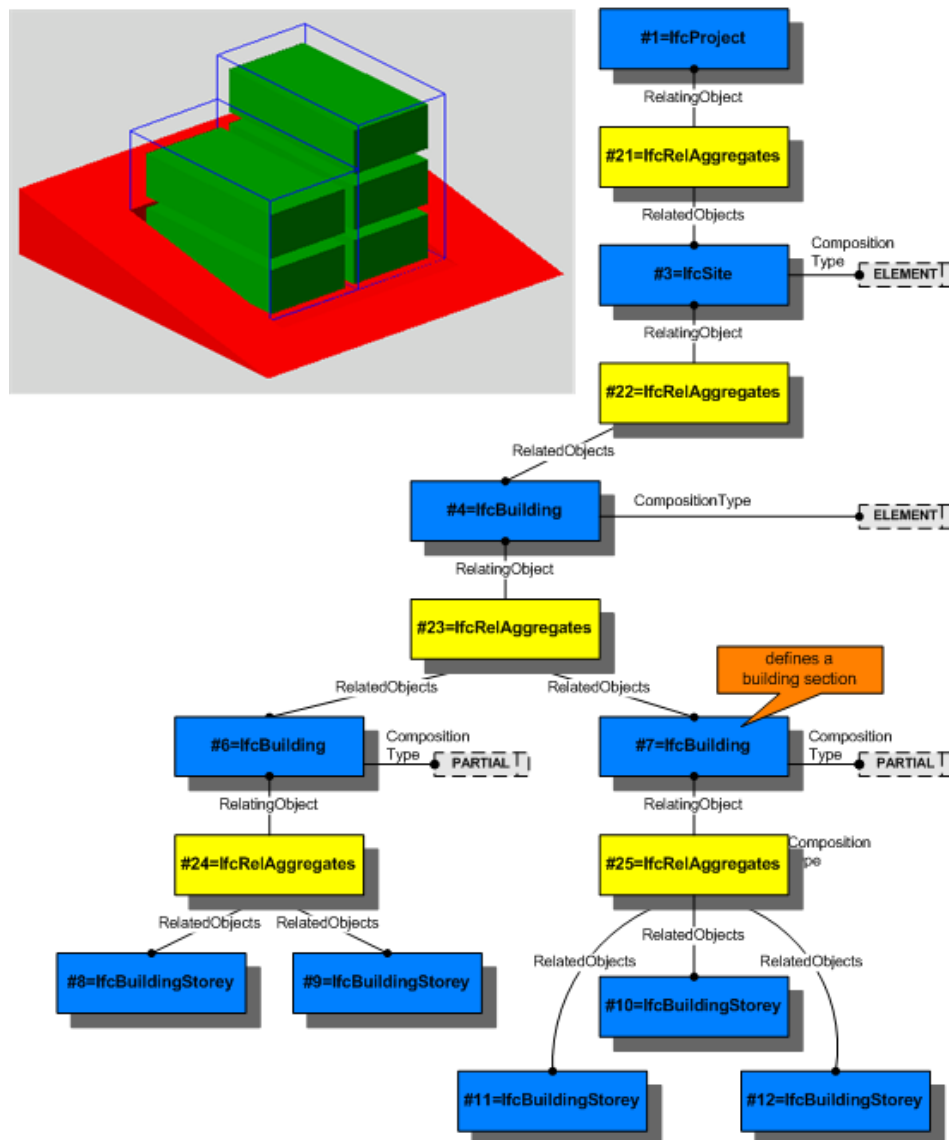


Abb. 3.3.: Exemplarische Unterteilung räumlicher Strukturen. Quelle: [MSG07]

Es ist zu erwähnen, dass es auch eine Unterteilung innerhalb der selben Untereinheiten geben kann. Grafik 3.3 demonstriert anhand eines Beispiels Möglichkeiten der Untertei-

3. Die Industry Foundation Classes

lung. Dabei legt man über das Attribut *CompositionType* die untergeordnete Unterteilung fest.

3.3.5. Platzierung

Jedes Objekt, das in der Raumstruktur eine Rolle spielt, sei es ein Stockwerk oder ein Bauelement, ist direkt oder indirekt von *IfcProduct* abgeleitet. Somit benötigt es über ein Attribut dieser Klasse eine Platzierung, die durch die Klasse *IfcObjectPlacement* realisiert wird. Dabei kann die Platzierung absolut oder relativ zu einer anderen Platzierung (von anderen Objekten) erfolgen. Eine derartige Platzierung erfordert eine Position im Koordinatensystem und eine Ausrichtung der Achsen. Weiterhin gibt es die Möglichkeit, Objekte an virtuellen Schnittpunkten von Achsen erzeugter Gitter zu platzieren.

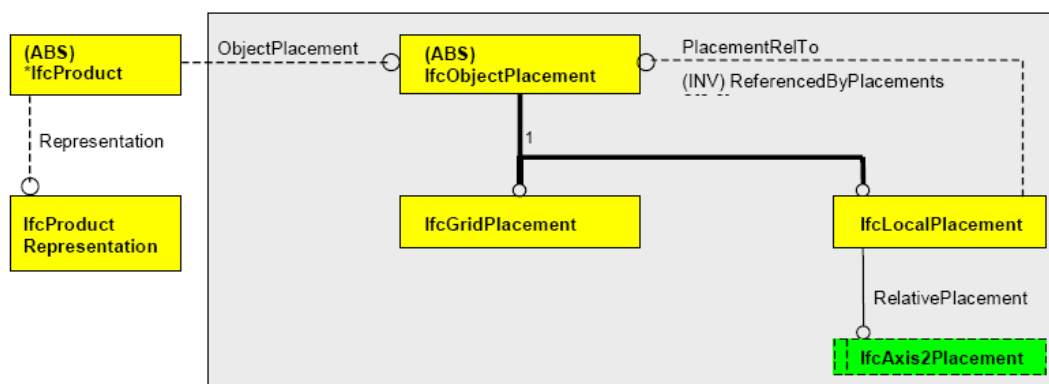


Abb. 3.4.: Struktur von *IfcObjectPlacement*. Quelle: [Lie09]

3.3.6. Geometrische Repräsentation

Der zweite wesentliche Aspekt von Objekten, die von *IfcProduct* abgeleitet werden, ist neben der Platzierung die Möglichkeit, eine oder mehrere Repräsentationen bereitzustellen, die Bezug auf den in Kapitel 3.3.3 angesprochenen geometrischen Kontext nehmen. Objekte, die eine Platzierung besitzen, benötigen in der Regel immer die Klasse *IfcProductDefinitionShape* als Spezialisierung von *IfcProductRepresentation*. Für jeden geometrischen Kontext enthält *IfcProductDefinitionShape* als Container jeweils eine entsprechende Repräsentation, die durch die Klasse *IfcShapeRepresentation* realisiert wird. Abbildung 3.5 verdeutlicht die Zusammenhänge.

Die jeweilige Repräsentation enthält nun ihrerseits eine Anzahl von *IfcRepresentationItem*-Klassen. Diese Klassen setzen letztendlich den Inhalt der Repräsentation um. Es gibt drei Spezialisierungen, um Geometrien zu beschreiben:

- Durch direkte Modellierung über Volumen- oder Flächenmodelle (*IfcGeometricRepresentationItem*)

3. Die Industry Foundation Classes

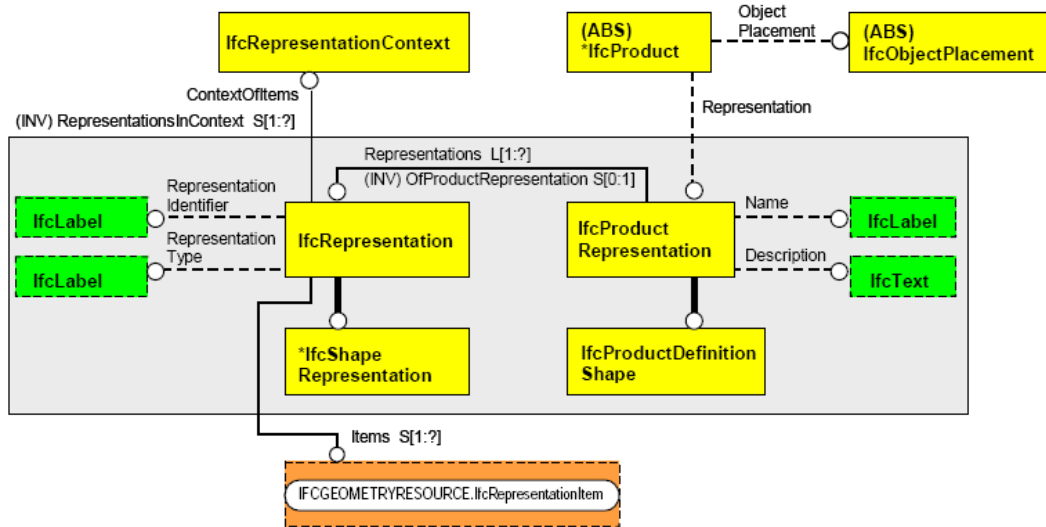


Abb. 3.5.: Definition von geometrischen Repräsentationen. Quelle: [Lie09]

- Durch Definition über topologische Daten (*IfcTopologicalRepresentationItem*)
- Durch die transformierte Wiederverwendung einer Repräsentationsvorlage (*IfcMappedItem*)

In dieser Arbeit wurden die Bauteile direkt modelliert. Die IFC bieten eine große Anzahl an Möglichkeiten der direkten Modellierung. Abbildung 3.6 gibt einen Überblick über die Einbindung und grundlegende Aufteilung von *IfcGeometricRepresentationItem*. Es gibt dafür folgende Basismodellierungsvarianten:

- *Curve2D*: Zweidimensionale Elemente werden mit Hilfe von Linien entlang eines Pfades dargestellt.
- *GeometricSet*: Objekte werden durch Punkte, Kurven und Flächen erzeugt. Eine Unterversion namens *GeometricCurveSet* beschränkt die Darstellung nur auf Punkte und Kurven.
- *SurfaceModel*: Die Beschreibung eines Objektes geschieht mittels Flächen bzw. Polygonzügen. Diese können offen und geschlossen sein. Gut für die Darstellung von filigranen Objekten und Landschaftsmodellen [Fli03].
- *SolidModel*: Generierung von Volumenkörpern. Dafür gibt es drei Varianten:
 - *Brep* (Boundary Representation): Oberflächenmodelle, die sich aus Polygonen mit gerichteten Normalen zusammensetzen.
 - *CSG* (Constructive Solid Geometry): Darstellung von Körpern, die auf primitiven Volumina basieren und mittels boolescher Operationen verschnitten werden. *Clipping* wird dabei als Unterform unterstützt.

3. Die Industry Foundation Classes

- *SweptSolid*: Grundflächen werden entlang eines Vektors oder kreisförmig extrudiert. In einer erweiterten Variante ist dies auch entlang eines Pfades möglich.
- *BoundingBox*: Eine Repräsentation für eine simple 3D-Darstellung von Körpern. Dabei wird der umschließende Quader dargestellt. BoundingBox ist für Anwendungen gedacht, die keine komplexen Geometrien darstellen können.
- *SectionedSpine*: Ein Volumenkörper wird anhand von Querschnitten, die entlang eines Pfades verlaufen, dargestellt. Die Bereiche zwischen den Querschnitten werden mit Linien verbunden, die sich an Fixpunkten der Querschnitte orientieren.

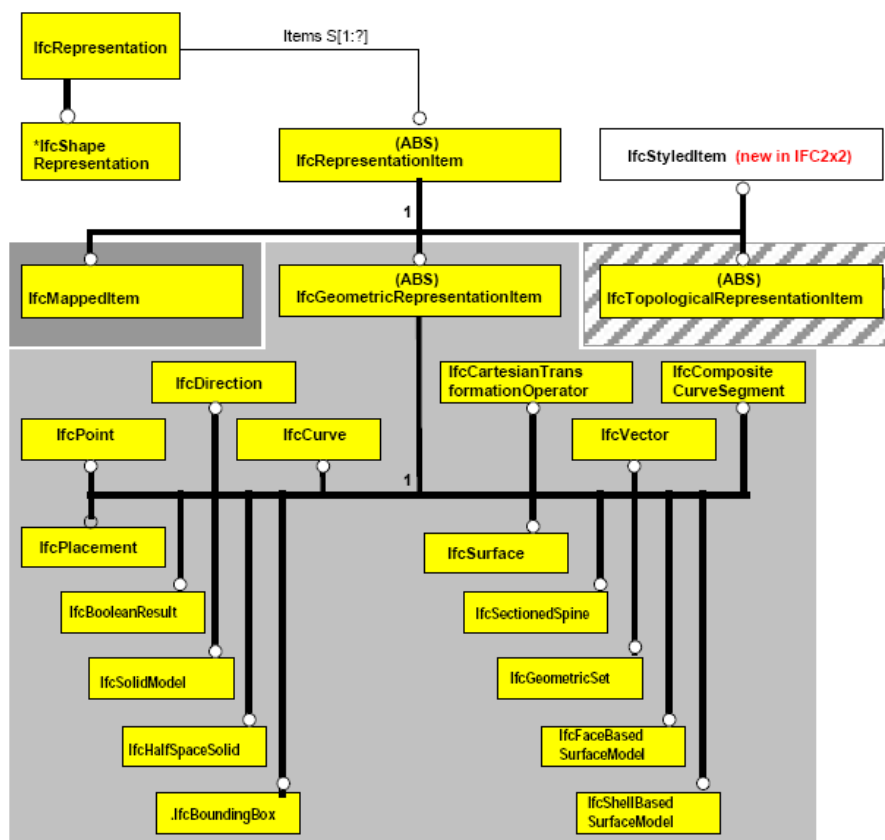


Abb. 3.6.: Varianten von IfcRepresentationItem. Quelle: [Lie09]

Für die Bauelemente des Referenzobjektes wurde die Generierung von Volumenkörpern gewählt, und zwar durch Extrusion aus einer Profilfläche. Zur Darstellung der Profilflächen stellen die IFC einen komplexen Klassenbaum zur Verfügung, der Möglichkeiten zur Erzeugung vordefinierter parametrisierter Profile, aus selbstdefinierten 2D-Flächen erstellter Profile, daraus kombinierter oder transformierter Profile bietet.

3.3.7. Gebäudeelemente

Alle Gebäudeelemente leiten sich direkt von *IfcBuildingElement* ab, das seinerseits aus *IfcElement* hervorgeht. Letztere ist die Elternklasse aller Bauteile. Neben Gebäudeelementen trifft dies unter anderem auch auf Möbel oder elektrische Elemente zu, für die jeweils eine eigene Klasse definiert ist, die sich aus *IfcElement* ableitet.

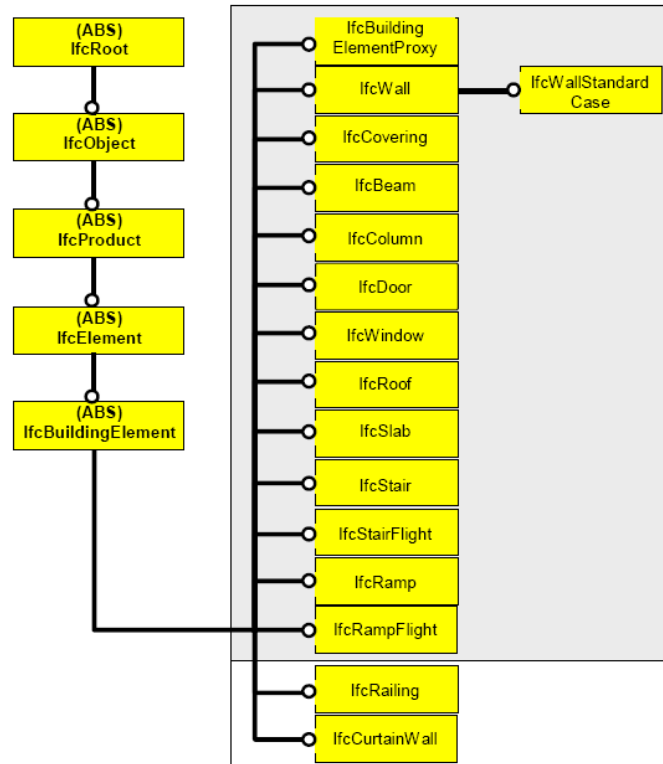


Abb. 3.7.: Vererbungsstruktur von *IfcBuildingElement*. Quelle: [Lie09]

IfcElement bietet eine Reihe wesentlicher (ererbter) Eigenschaften:

- Identifikation und Änderungsmanagement
- Verknüpfungen durch Beziehungen wie Aggregation oder Assoziationen (beispielsweise zu Materialien)
- Geometrische Repräsentation
- Platzierung
- Zuordnung von Typen und Eigenschaften

IfcBuildingElement selbst besitzt keine neuen Attribute, diese Klasse dient zur logischen Abgrenzung und ist abstrakt. Abbildung 3.7 zeigt die weitere Unterteilung von

3. Die Industry Foundation Classes

IfcBuildingElement, die letztendlich zu den Klassen führt, die die Gebäudeelemente repräsentieren. Über die Beziehungsklasse *IfcRelContainedInSpatialStructure* werden Bauelemente einer räumlichen Einheit zugeordnet. Siehe dazu Abbildung 3.8.

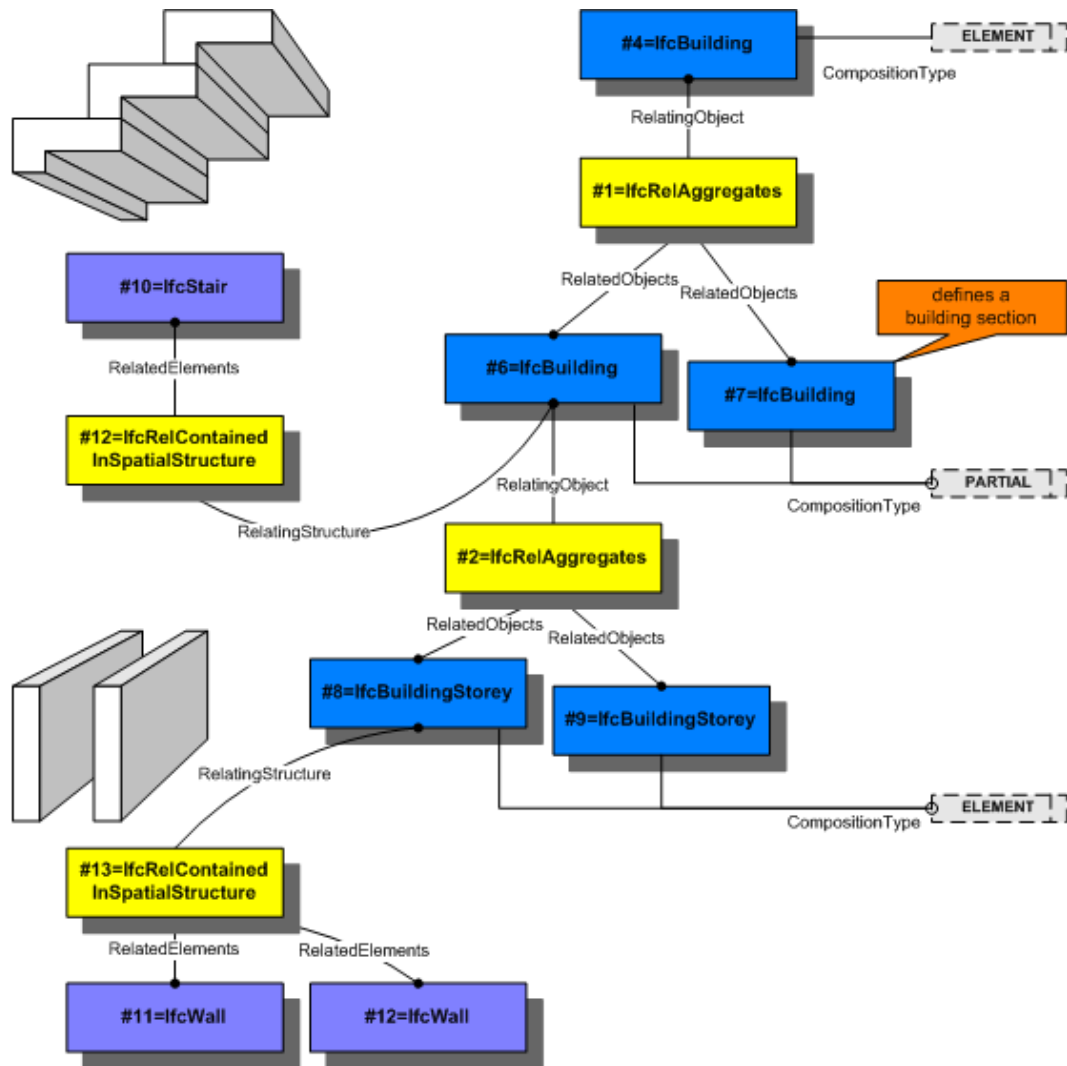


Abb. 3.8.: Eingliederung von Bauelementen in räumliche Strukturen. Quelle: [MSG07]

Im Folgenden wird auf die Bauelemente eingegangen, die zur Modellierung des Referenzobjektes verwendet wurden. Die Erläuterungen folgen den Definitionen der IAI. Es sind Richtlinien, die bestimmen, welcher Art die Attribute für das jeweilige Bauelement sein muss, also welche Klassen dafür gültig sind und welche nicht. Wird diesen Richtlinien nicht entsprochen, hängt es von den jeweiligen Interpreten der Software ab, ob und wie weit die jeweiligen Elemente dargestellt werden. Nur wenn die Objekte strikt nach Vorgabe modelliert werden, ist sichergestellt, dass sie in jeder Software gleich aussehen.

3.3.7.1. Stützen, Balken und Lastglieder

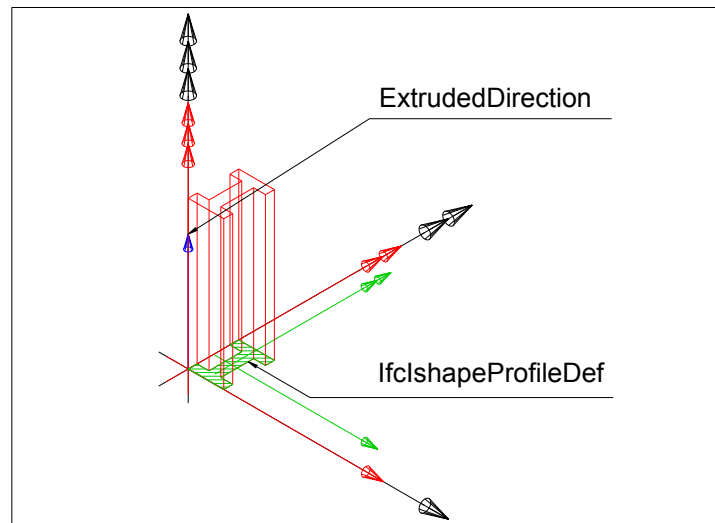


Abb. 3.9.: Extrusionsdarstellung einer Stütze mit einem I-Profil. Quelle: [MSG07]

Die im Referenzobjekt primär verwendeten Bauelemente sind Stützen und Balken, die durch die Klassen *IfcColumn* und *IfcBeam* repräsentiert werden. Die IFC definieren Stützen und Balken annähernd gleich, nur dass eine Stütze vorwiegend senkrecht und ein Balken vorwiegend waagrecht zu konstruieren ist. Lastglieder, die im Referenzmodell als Sekundärelemente zur Aussteifung verwendet werden, unterliegen diesbezüglich keinen Konventionen.

Die Regeln für die grafische Darstellung lassen die meisten der in 3.3.6 vorgestellten Modellierungsarten zu (SweptSolid, Clipping, MappedRepresentation, SurfaceModel, Brep, BoundingBox). Speziell für die Erzeugung von Volumenkörpern durch lineare Extrusion aus Profilflächen existieren Beschränkungen hinsichtlich der Auswahl der Klassen, die die Profilflächen definieren.

3.3.7.2. Wände

Wände, verkörpert durch die Klasse *IfcWall*, werden laut IAI als (meist) vertikale, lasttragfähige Konstruktionen beschrieben. Sie gehören zu den komplexesten Gebäudeelementen der IFC, daher steht mit *IfcWallStandardCase* eine Unterklasse für vereinfachte Wandkonstruktionen zur Verfügung, die einen Großteil der Modellierungsfälle abdeckt. Diese haben eine konstante Materialdicke entlang ihres Pfades, der entweder eine Gerade oder einen Kreisbogen beschreibt. Ihr Querschnitt ist rechteckig, was exakt senkrechte

3. Die Industry Foundation Classes

Wandflächen bedeutet.

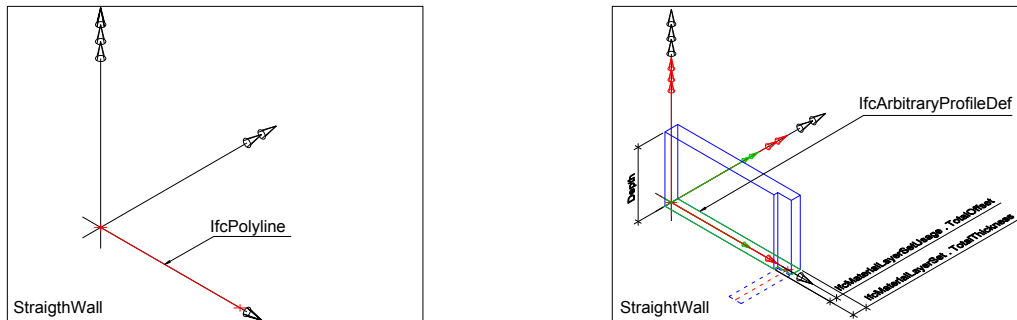


Abb. 3.10.: Darstellung einer Standardwand in den zwei erforderlichen Kontexten. Quelle: [MSG07]

Die geometrische Repräsentation von *IfcWallStandardCase* ist speziell festgelegt. Standardwände müssen zwei Kontexte besitzen. Zum einen ist dies eine zweidimensionale *Curve2D*-Darstellung des Pfades der Wand, zum anderen wird eine dreidimensionale Modellierung als *SweptSolid* oder *Clipping* verlangt.

3.3.7.3. Platten

Platten (*IfcSlab*) werden modellierungstechnisch ähnlich gehandhabt wie Stützen, Balken und Lastglieder. Konstruktiv werden sie horizontal eingebaut. Obwohl sie laut Definition für Modellierungsarten wie *SweptSolid*, *Clipping*, *MappedRepresentation*, *Brep*, und *BoundingBox* zugelassen sind, ist die Extrusionsdarstellung die praktischste Variante. Dabei fungiert die Extrusionslänge als Plattendicke, wenn die Extrusion orthogonal zum Plattenprofil erfolgt. Dies ist bei der Modellierung des Referenzobjektes der Fall.

Für Platten sollten neben selbstdefinierten Profilen nur die primitiven parametrisierten Profile wie Rechteck, Kreis oder Ellipse verwendet werden. I- oder L-Profile würden hier keinen großen Sinn ergeben.

3.3.8. Strukturmodell

Die IFC bietet in der Domänenschicht mit der *IfcStructuralAnalysisDomain* eine Reihe von Klassen, die der Umsetzung von Strukturmodellen dienen sollen. Die Domäne befindet sich noch im Aufbau und wird mit jeder neuen Version der IFC erweitert. Momentan bietet die *IfcStructuralAnalysisDomain* nach [MSG07] folgende Umsetzungsmöglichkeiten:

- Punktförmige, lineare und flächige Strukturelemente und Auflager
- Deren Anschlüsse und Auflagerbedingungen

3. Die Industry Foundation Classes

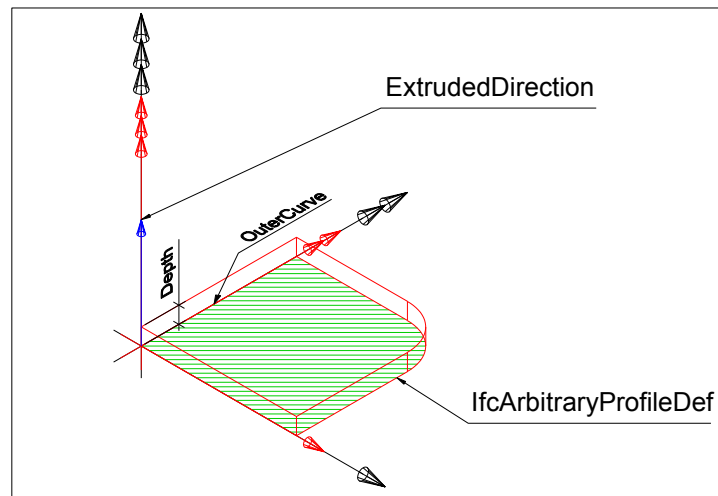


Abb. 3.11.: Extrusionsdarstellung einer Platte mit selbstdefiniertem Profil. Quelle: [MSG07]

- Definition von Punkt-, Linien- und Flächenlasten und Temperatureinwirkungen, die zu Lastgruppen, Lastfällen und Lastfallkombinationen zusammengesetzt werden können
- Spezifikationen verschiedener Statikmodelle, um unterschiedliche Gebäudebereiche zu modellieren inkl. der Abhängigkeiten der Modelle voneinander
- Integration der Berechnungsergebnisse in Form von Kräften und Verschiebungen

Ausgangspunkt für ein Strukturmodell ist die Klasse *IfcStructuralAnalysisModel*, die über eine Beziehung, verkörpert durch die Klasse *IfcRelServicesBuildings*, mit einer oder einer Gruppe von Raumstruktureinheiten (siehe Kapitel 3.3.4) verknüpft wird. *IfcStructuralAnalysisModel* enthält neben der Unterscheidung seiner Dimension eine Liste der eingesetzten Lastgruppen und eine Liste mit Gruppen von Ergebnissen.

Des Weiteren wird *IfcStructuralAnalysisModel* über eine Relation mit einer Gruppe von Struktureinheiten verknüpft. Diese Gruppe kann invers aus der Analysismodellklasse abgefragt werden. Zusätzlich ist eine weitere, ebenfalls invers abfragbare Relation vorgesehen, mit der man mehrere (partielle) Modelle hierarchisch miteinander verbinden kann.

Die eben angesprochenen Struktureinheiten unterteilen sich in zwei Gruppen: In Strukturelemente der Klasse *IfcStructuralMember* und in Knoten oder Auflager, die durch die Klasse *IfcStructuralConnection* realisiert werden.

3. Die Industry Foundation Classes

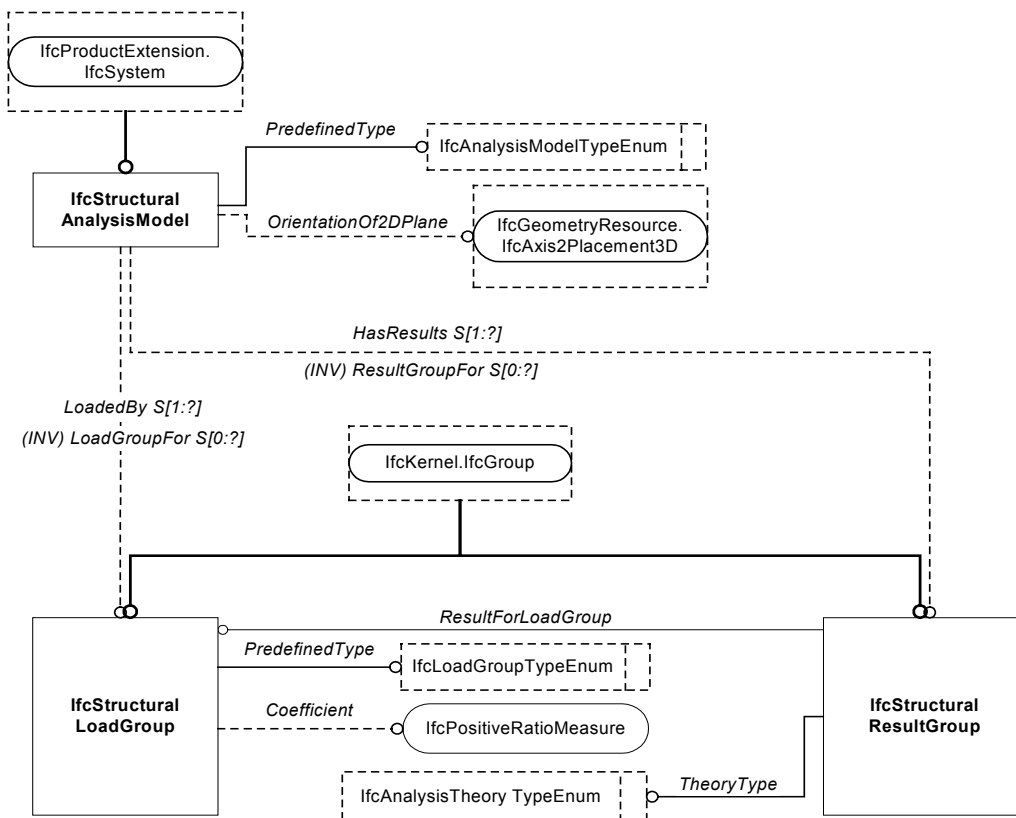


Abb. 3.12.: Relationen mit IfcStructuralAnalysisModel. Quelle: [WKLS03]

Die Klasse **IfcStructuralMember** beschreibt die strukturellen Eigenschaften eines Bauelements. Sie unterteilt sich nochmals in zwei Klassen: In linienförmige Elemente und in Flächenelemente. Ihr kann eine Material- und eine Profildefinition zugewiesen werden. Die Strukturelemente müssen einen jeweils vordefinierten Typ annehmen, der festlegt, welche Arten von Belastungen das Element aufnehmen kann. Flächenelemente können zudem noch ihre Dicke definieren. Weiterhin müssen sie eine geometrische Repräsentation vom Typ **IfcTopologyRepresentation** besitzen, in der mit topologischen Primitiven über Kanten bzw. Flächen eine Geometrie erzeugt wird.

Eine Relation namens *IfcRelConnectsStructuralElement* weist dem strukturellen Bauteil ein konstruktives Bauteil zu. Genau hier wird die logische Verbindung zwischen dem Strukturmodell und dem konstruktiven Modell geschaffen.

Die Klasse **IfcStructuralConnection** definiert Knoten oder Auflager und ist ebenfalls in Unterklassen gegliedert. Darin wird zwischen Punkt-, Linien oder Flächenverbindungen unterschieden. Für jede dieser drei Unterklassen können die jeweiligen Randbedingungen in Form von Steifigkeiten festgelegt werden.

3. Die Industry Foundation Classes

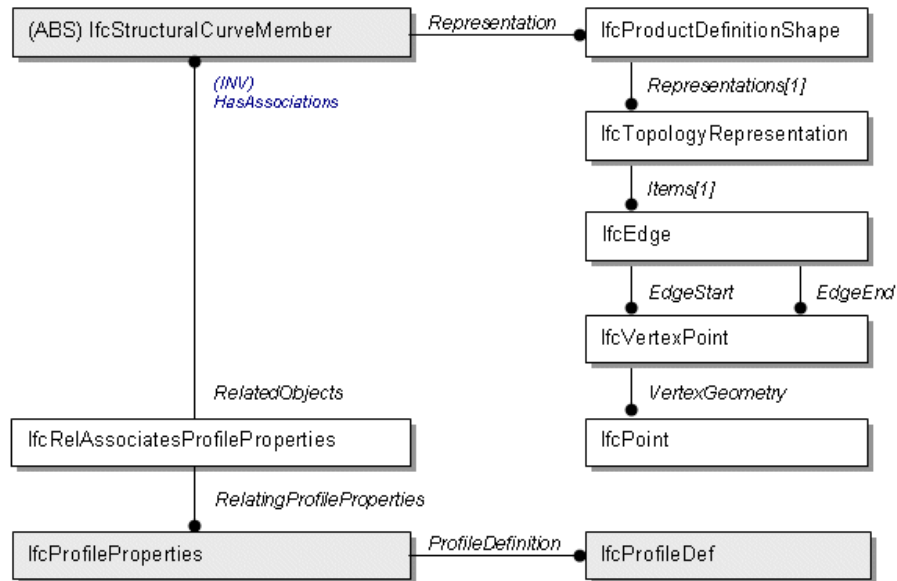


Abb. 3.13.: Aufbau des Linienelementes IfcStructuralCurveMember. Quelle: [MSG07]

Jeweils ein Strukturelement und ein Knoten bzw. Auflager werden durch eine Relation *IfcRelConnectsStructuralMember* verbunden. Werden mehrere solcher Verknüpfungen mit einer Gruppe von Elementen erzeugt, können komplexe Strukturen aufgebaut werden.

Das Prinzip der Lasten folgt der Philosophie der starken Strukturierung der IFC. Ein Strukturmodell besitzt eine Liste von Lastgruppen der Klasse *IfcStructuralLoadGroup*. Innerhalb einer Lastgruppe wird mit diversen Attributen festgelegt, um welche Art von Kombinationen der Lasten es sich handelt, welcher Art die Lasten sind (Verkehrslasten oder ständige Lasten), welcher Sicherheitsfaktor gewählt wird und wie diese Kombination bezeichnet wird. Die Klasse, die eine der Einwirkungen darstellt (*IfcStructuralAction*), spezifiziert sich durch die Dimensionalität (Punkt-, Linien- oder Flächenlast) in eine Unterklasse, die eine Platzierung, eine (topologische) geometrische Repräsentation und die physische Definition der Einwirkung in Form eines Kraft-, Verschiebungs- oder Temperaturvektors besitzt.

Zusammenfassend kann man sagen, dass ein Strukturmodell in den IFC grundlegend eigenständig aufgebaut ist. Es wird logisch an eine größere geometrische Gebäudestruktur (z.B. ein Gebäude oder Stockwerk) gekoppelt. Einzelne strukturelle Bestandteile können mit Bauteilen des geometrischen Modells assoziiert werden. Im Prinzip ist das Strukturmodell jedoch gekapselt, was dem Zweck dient, in möglichst abstrakter Form Anwendungen der statischen Berechnung zur Verfügung zu stehen.

3.3.9. Material

Objekten in einem IFC-Modell kann ein Material (*IfcMaterial*), eine Materialliste (*IfcMaterialList*), eine Materialschicht (*IfcMaterialLayer*) oder ein Set von Materialschichten (*IfcMaterialLayerSet*), das zudem noch an das Objekt angepasst werden kann (*IfcMaterialLayerSetUsage*), zugewiesen werden.

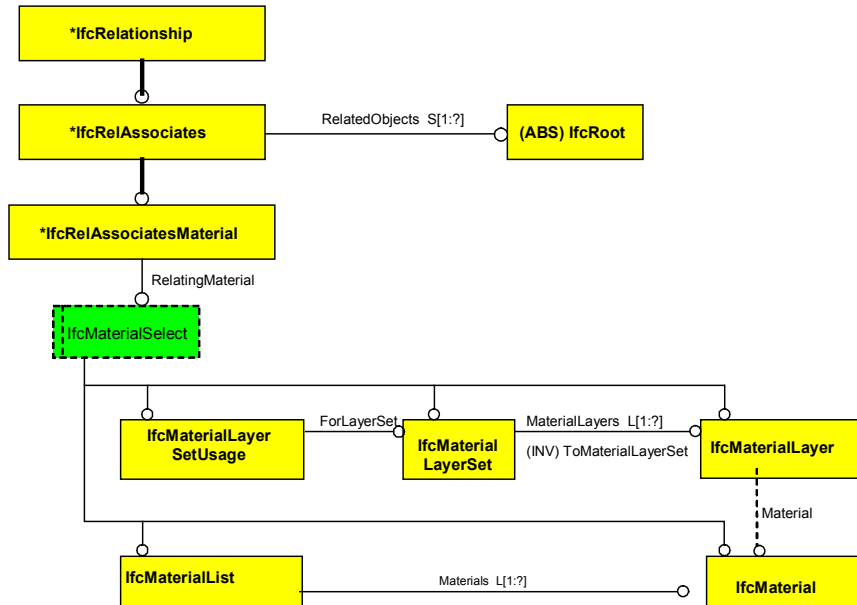


Abb. 3.14.: Funktionsweise der Materialzuweisung. Quelle: [Lie09]

Die Zuweisung erfolgt über eine Relationsklasse. Gebäudeelemente sowie alle Klassen, die sich in irgendeiner Form von *IfcElement* ableiten, besitzen ein inverses Attribut, das die zugehörige Materialrelation zurückliefert. In der Regel benutzen viele Elemente in einem Gebäude das gleiche Material bzw. den gleichen Schichtenaufbau. Daher werden diese Elemente als Liste in der Relation geführt, die sie mit dem benötigten Material bzw. dem Schichtenaufbau verknüpft. Abbildung 3.14 stellt die Zusammenhänge anschaulich dar.

Während das geometrische Handling der Materialklassen sowie das Management der Schichten durch die IFC sehr detailliert umgesetzt wurde, besitzt die Basisklasse *IfcMaterial* keine weiteren Eigenschaften bis auf den Namen. Für eine Kopplung, wie sie im Rahmen dieser Arbeit mit einem FEM-Programm wie Ansys durchgeführt werden soll, ist dies natürlich nicht ausreichend. Dafür bietet die *MaterialPropertyResource* einen Ausweg. Sie stellt Containerklassen zur Verfügung, mit denen man an Materialien Eigenschaftssets aus unterschiedlichen Fachbereichen anhängen kann. Der Mechanismus funktioniert so, dass man eine Instanz einer der Eigenschaftsklassen, wie zum Beispiel *IfcMechanicalMaterialProperties* erzeugt, ihr Attribut beispielsweise für die Querdehnzahl

3. Die Industry Foundation Classes

oder den Elastizitätsmodul definiert und in einem weiteren Attribut die entsprechende Materialklasse zuweist. So lässt sich das Material um die benötigten Attribute für z.B. statische Berechnungen erweitern.

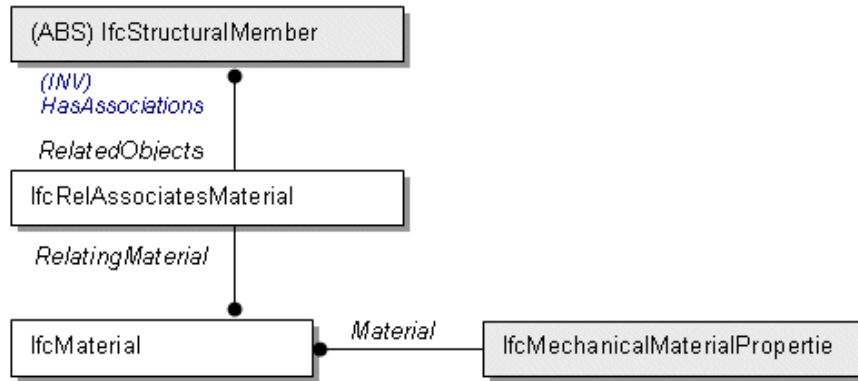


Abb. 3.15.: Materialverknüpfung eines Strukturelements. Quelle: [MSG07]

3.4. Das Open IFC Toolkit

Das Open IFC Toolkit stellt eine leistungsfähige Umsetzung der Industry Foundation Classes für Java-Programmierer dar. Java ist eine objektorientierte Programmiersprache, in der man Objekte in Java-Klassen modellieren kann. Das Toolkit stellt die Struktur der IFC in Form von Java-Klassen zur Verfügung, so dass man mittels Quellcode die Möglichkeit hat, sequenziell oder über Algorithmen IFC-Modelle aufzubauen, zu lesen, zu speichern oder weiter zu verarbeiten.

Im Rahmen der Arbeit wurde sowohl zur Generierung des Referenzobjektes als auch für den Konverter das Open IFC Toolkit in der Version 1.0 verwendet. Die Software unterliegt den OpenSource-Bedingungen und ist für nichtkommerzielle Nutzung frei verfügbar.

Im Rahmen der Entwicklung des Toolkits entstand auch ein leistungsfähiger, in Java geschriebener IFC-Modellbetrachter, der für die Entwicklung des Referenzmodells und zum Testen des Konverters unabkömmlich war.

4. Ansys

4.1. Überblick

Ansys ist eine FEM-Software und dient zur Lösung von linearen und nichtlinearen Problemen aus der Strukturmechanik, Fluidmechanik, Akustik, Thermodynamik, Piezoelektrizität, Elektromagnetismus sowie von kombinierten Aufgabenstellungen (Multiphysik). Ihre grundsätzliche Funktionsweise liegt in der Lösung von partiellen Differentialgleichungen über numerische Verfahren, der Methode der finiten Elemente. Das Objekt der Betrachtungen wird dabei in eine größere Anzahl endlich (*finit*) großer Elemente geteilt, die sich mit einer endlichen Zahl von Parametern beschreiben lassen. Die Elemente sind durch Knoten über Randbedingungen miteinander verknüpft. Durch eine für das Element bestimmte Ansatzfunktion wird zusammen mit Anfangs-, Rand- und Übergangsbedingungen ein Gleichungssystem erzeugt, das numerisch gelöst wird. Ansys besitzt eine Vielzahl von Elementtypen für ein-, zwei-, und dreidimensionale Aufgaben.

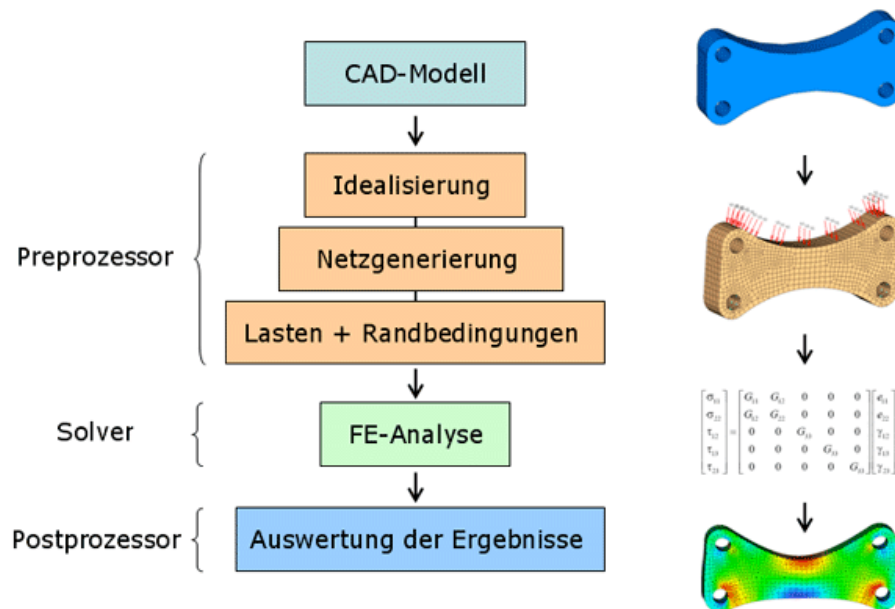


Abb. 4.1.: Ablauf einer Finite-Elemente-Analyse. Quelle: [SF211]

Eine Analyse in Ansys läuft in drei Schritten ab. Zuerst wird im Präprozessor-Modus

4. Ansys

durch den Anwender eine idealisierte Geometrie erstellt. Dazu gehört die Vereinfachung und Reduzierung der Geometrie auf relevante Strukturen. Symmetrien können beispielsweise Modelle erheblich vereinfachen. Es müssen geeignete Elementtypen gewählt und Materialdaten und zusätzliche Elementeigenschaften definiert werden. Danach wird vom Programm ein Netz aus Knoten und Elementen generiert. Die Feinheit des Netzes ist vom Nutzer steuerbar. Ansys stellt diverse Möglichkeiten zum Datenimport von geometrischen Modellen zur Verfügung. Es gibt unter anderem Schnittstellen für Formate wie DXF, IGES und Parasolid. Ein generelles Problem dabei ist, dass diese Geometriemodelle oft zu detailliert abgebildet wurden und demnach keine ideale Vorlage für FEM-Modelle liefern (Beispiel: [DOO07]). Deswegen ist beim Import in der Regel eine Nachbearbeitung in mehr oder weniger großem Umfang notwendig.

Der zweite Schritt ist der Lösungsabschnitt. Hier werden die von außen auf das Bauteil einwirkenden Einflüsse wie Randbedingungen bzw. Bindungen und Lasten festgelegt. Das Programm führt zuerst diverse Plausibilitätsprüfungen durch. Ist die Geometrie zu stark verzerrt oder fehlen nötige Materialkennwerte, wird die Berechnung gar nicht erst durchgeführt. Sind alle Ausgangsbedingungen erfüllt, baut Ansys z.B. im Falle einer statischen Berechnung aus den Daten die Elementsteifigkeitsmatrizen und daraus die Gesamtstruktursteifigkeitsmatrix und den Lastvektor auf. Die Auflösung des Gleichungssystems führt zum Vektor der Knotenverschiebungen, der eigentlichen Unbekannten einer FEM-Berechnung. Daraus werden letztendlich Verzerrungen (Dehnungen) sowie Spannungen ermittelt.

Im Postprocessing, dem letzten Schritt, werden die Ergebnisse ausgewertet und dargestellt. Früher geschah die Ausgabe ausschließlich in Form von Listen und Tabellen von Reaktionskräften und Verschiebungen. Inzwischen lassen sich Bereiche detailliert selektieren und grafisch zum Beispiel durch Isoflächen darstellen. Es ist möglich, Ergebnisdaten weiter zu verarbeiten, zu überlagern und Grenzwerte zu bilden.

4.2. Modellierung in Ansys

Um ein physikalisches Modell zu generieren, muss zuerst ein geometrisches Modell vorliegen. Die Struktur eines geometrischen Modells in Ansys besteht aus geometrischen Komponenten, die aufeinander aufbauen. Sie sind jeweils auf ihrem Primitivenlevel nummeriert, das heißt, Punkte besitzen eine eigene Nummerierung, Linien eine eigene und so fort. Die Eckpfeiler der Geometrie sind sogenannte Keypoints, Punkte, die eine geometrische Position besitzen. Jeweils zwei Keypoints bilden eine Keyline, eine Linie, die die Nummern der Keypoints kennt. Aus den Keylines besteht eine Fläche und aus Flächen wiederum ein Volumenkörper. Auch wenn es in Ansys möglich ist, zweidimensionale Modelle zu berechnen, werden die meisten Modelle wohl eher aus Volumenkörpern be-

4. Ansys

stehen.

Um diese Körper zu erzeugen, muss nicht zwangsläufig der schrittweise Aufbau vollzogen werden. Im Gegenteil, es ist nach [MG07] sogar eher unpraktisch und wäre bei praxisbezogenen Modellen zuviel Aufwand. Die Prinzipien zur Generierung von Volumenkörpern folgen teilweise denen der geometrischen Repräsentation von Bauteilen der IFC (siehe Kapitel 3.3.6). Körper können aus Grundflächen in Rotation oder linear extrudiert werden. Es lassen sich über boolsche Operationen Körper miteinander verschneiden. Das Ganze funktioniert auch kombiniert, siehe Abbildung 4.2.



Abb. 4.2.: Verknüpfung zweier Modellierungskonzepte in Ansys. Links die Erzeugung einer Querschnittsfläche mittels boolscher Verschneidungen, rechts die Erzeugung eines Volumenkörpers durch Rotationsextrusion dieser Fläche.

Bevor die Geometrie gemesht, also in Elemente eingeteilt und vernetzt wird, muss bestimmt werden, welcher Art die Elemente sind. Ansys bietet über 100 verschiedene Elemente mit unterschiedlichen Eigenschaften. So gibt es Unterschiede hinsichtlich der Grundgeometrie und Belastungsart (ist es ein z.B. Biegebalken, eine Platte oder ein Druckstab? Soll das Element auf Temperaturänderungen reagieren?), der Anzahl der Knoten am Element, die mit anderen Elementen verbunden sind, und die betrachtete Dimension. Elementtypen können mit zusätzlichen, meist geometrischen, Parametern versehen werden, den sogenannten Real Constants. Generiert man beispielsweise einen Elementtyp für eine Platte, dann wird in einer solchen Konstante die Plattendicke angegeben. Weiterhin muss ein Materialtyp definiert werden. Durch Angabe von Materialeigenschaften wie beispielsweise Elastizitätsmodul oder Wärmeleitfähigkeit wird ein Material modelliert. Sowohl Elementtypen als auch Materialtypen besitzen jeweils wieder eine eigene Nummerierung. Über die entsprechenden Nummern werden Element- und Materialtypen beim Meshen referenziert.

Nach dem Meshen liegt ein Modell vor, das aus durch Knoten miteinander verbundenen Elementen besteht. Es benötigt nun noch die Modellierung der auf es einwirkenden Lasten. Nach [MG07] werden diese in sechs Kategorien eingeteilt: Bindungen von Freiheitsgraden, Kräfte (Punktlasten), Oberflächenlasten, Volumenlasten, Trägheitskräfte

4. Ansys

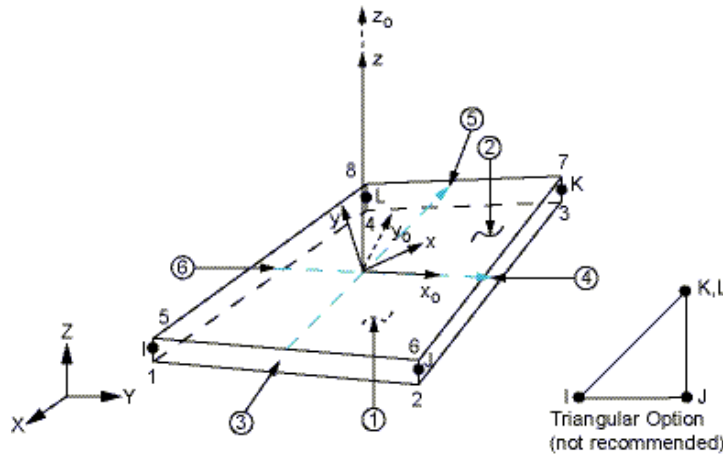


Abb. 4.3.: Element SHELL181 in Ansys. Quelle: [ANS09]

te und Lasten für gekoppelte Feldberechnungen. Auch wenn Lasten am physikalischen Modell benötigt werden, bietet Ansys die Möglichkeit, sie schon vor dem Meshen am geometrischen Modell anzubringen. Sie werden dann vor der Berechnung auf das physikalische Modell übertragen. Vorteil der Lastenaufbringung am geometrischen Modell ist einerseits die Tatsache, dass bei Änderungen am Modell die Lasten nicht neu eingegeben werden müssen, da sie sich nicht direkt auf das neu zu erstellende Netz beziehen. Außerdem ist die Eingabe über die GUI am geometrischen Modell meist einfacher. Nachteil ist, dass das Übertragen von Bindungen problematisch ist, da ein Keypoint des Geometriemodells nicht einfach mit einem Knoten am physikalischen Modell gleichgesetzt werden kann.

Die Modellierung ist damit abgeschlossen und das Modell kann nun berechnet werden. Für den Fall einer Strukturanalyse hätte man ein physikalisches Modell mit Lasten vorliegen, an dem die Knotenverschiebungen ermittelt werden würden.

4.3. APDL

Ansys bietet neben der Eingabe der Daten über die grafische Benutzeroberfläche auch die Möglichkeit zur Steuerung über eine Skriptsprache. Diese Sprache wird *ANSYS Parametric Design Language* genannt, kurz APDL. Sie ermöglicht es dem Anwender, über Befehle mit Werten und Parametern Modelle zu generieren, erweitern und zu bearbeiten. Die Funktionalität wird durch die Benutzung von Schleifen, Verzweigungen und Macros erweitert. Somit bietet APDL ein Interface, um von anderen Programmen generierte Modelle in Textform in Ansys einzugeben.

Der im Rahmen dieser Arbeit programmierte Konverter generiert Dateien mit APDL-Code, die von Ansys eingelesen und verarbeitet werden können.

5. Modellierung des Referenzobjektes

5.1. Allgemeines

Um ein Modell des Referenzobjektes im IFC-Format zu erzeugen, wurde das in Kapitel 3.4 beschriebene Open IFC Toolkit verwendet. Da die IFC eine starke Strukturierung und die Modelle einen dementsprechend komplexen Aufbau besitzen, wurde zur Unterstützung ein Hilfsmodell entwickelt, um die direkte Generierung mittels elementarer Bauteile zu vereinfachen. Das Hilfsmodell bindet die Klassen des Toolkits seinerseits ein. Es übernimmt Aufgaben wie das Erstellen von Beziehungen zwischen Objekten, Eingliederung in die Raumstruktur oder Eintragen in die Datenbank. Auf Basis des Hilfsmodells war es auf einfache Weise möglich, mit wenigen Parametern Modelle aus Bauelementen zusammenzubauen. Pro Modell ist in der Regel eine Quellcodedatei nötig, die unter Anwendung der Klassen des Hilfsmodells ein Modell aufbaut und im STEP-Format speichert. Die Datei kann dann mit einem IFC-Betrachter geöffnet und das Modell sichtbar gemacht werden.

5.2. Modellierungsmöglichkeiten

Grundlage für ein IFC-Modell sind die Bereitstellung der Grundstrukturen wie Projekt, Bauwerk und Stockwerke. Diese können erzeugt werden. Weiterhin ist die Definition von Materialien inklusive mechanischer und allgemeiner Materialeigenschaften möglich. Der zentrale Kern der weiteren Betrachtungen sind allerdings die fünf Typen von Bauelementen, die für die Modellierung verfügbar sind:

- Stützen, dargestellt durch das Element `IfcColumn`
- Balken, dargestellt durch das Element `IfcBeam`
- Wände, dargestellt durch das Element `IfcWallStandardCase`
- Zugstäbe, dargestellt durch das Element `IfcMember`
- Platten, dargestellt durch das Element `IfcSlab`

5. Modellierung des Referenzobjektes

Alle fünf Elemente definieren sich durch eine Grundfläche, die in eine Richtung linear extrudiert wird. Die Art der Grundfläche wird durch IFC-Definitionen und durch lokale Festlegungen eingeschränkt. Für Säulen, Balken und Zugstäbe sind rechteckige, kreisrunde und I-Profil-Grundflächen erlaubt, Wände und Platten dürfen nur eine rechteckige Grundfläche besitzen. Die Position und Ausrichtung der Elemente im Koordinatensystem ist frei wählbar, sollte jedoch möglichst den Konventionen der IFC entsprechen (Säulen stehen z.B. vorrangig senkrecht).

Als Materialien stehen Baustahl S235 und Beton C30/37 mit Parametern wie E-Modul, Querdehnzahl und Dichte zur Verfügung. Wände und Deckenplatten wurden aus Beton modelliert, Stützen, Balken und Zugstäbe aus Baustahl.

Primär sollte die Geometrie modelliert werden. Die Anforderungen an das Modell wurden jedoch erweitert, damit nach der Konvertierung in Ansys ein physikalisches Modell generiert werden kann. Deswegen gibt es auf struktureller Ebene zwei Erweiterungen. Erstens kann eine Einspannung an einer bestimmten Position im Koordinatensystem definiert werden. Hintergrund ist die IFC-Klasse *IfcStructuralPointConnection*, die in sechs Freiheitsgraden beschränkt werden kann. Siehe hierzu auch Kapitel 3.3.8. Die zweite Möglichkeit ist eine Einwirkung in Form einer Punktlast, ebenfalls definiert durch eine Position und einer Kombination aus sechs Lastwerten entsprechend den Freiheitsgraden. Hierzu wurde IFC-seitig die Klasse *IfcStructuralPointAction* benutzt.

Den Randbedingungen wurden aus Effizienz- und Übersichtlichkeitsgründen weitere Festlegungen getroffen:

- Es gibt ein globales Koordinatensystem, auf das sich die Positionierung jedes Elementes bezieht.
- Auf das Attribut *IfcOwnerHistory* wurde verzichtet, da es für die Konvertierung irrelevant ist.
- Der geometrische Kontext wurde ebenfalls weggelassen, er spielt für die Konvertierung keine Rolle.

5.3. Das Referenzobjekt

Es wurde in Java ein Quellcode erstellt, der das Referenzobjekt mit den im vorherigen Kapitel vorgestellten Modellierungsmöglichkeiten generiert. Über Parameter sind Variationen einstellbar:

- Die Stützen können mit Rechteck- oder I-Profil modelliert werden
- Die Aussteifung kann durch Zugstäbe oder Wände erfolgen

5. Modellierung des Referenzobjektes

Im Prinzip lassen sich fast alle Attribute parametrisieren, aber für die Bewertungen der Qualität wurde mit den oben genannten Variationen gearbeitet. Die Geometrie wurde, soweit angegeben, der Vorlage des Modells aus Kapitel 2 entnommen. An den Fußpunkten der Erdgeschoßsäulen wurden Einspannungen definiert.

Um in Ansys ein anschauliches Berechnungsergebnis zu erhalten, wurde exemplarisch eine Punktlast definiert, die an der Längsseite des Gebäudes in X-Richtung angreift.

Letztendlich wurden zwei Modellvarianten für die Qualitätsbewertung ausgewählt. Modell A besitzt Stützen mit I-Profil und wird mit runden Zugstäben ausgesteift (Abb. 5.1). Modell B besitzt Rechteckstützen und wird durch Wände ausgesteift (Abb. 5.2). In beiden Versionen wurden Deckenplatten modelliert. Für die Balken wurden jeweils I-Profile entsprechend den Vorgaben aus 2.2 verwendet.

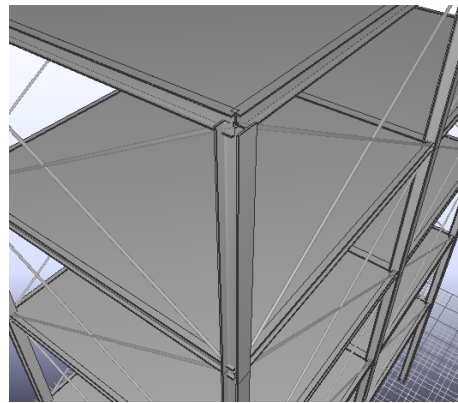
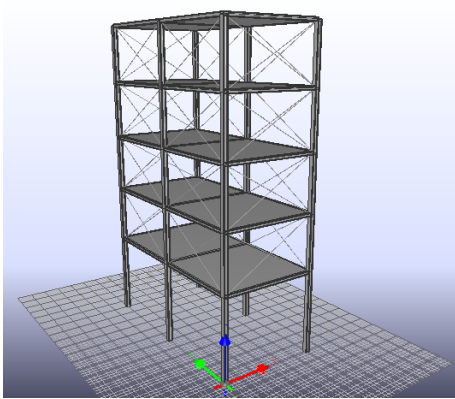


Abb. 5.1.: Darstellung des Referenzobjektes mit I-Profil-Stützen und Zugstäben im IFC-Viewer

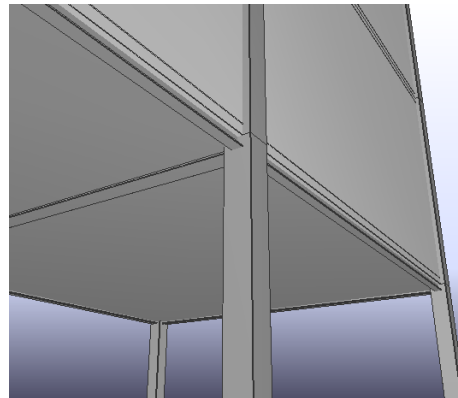
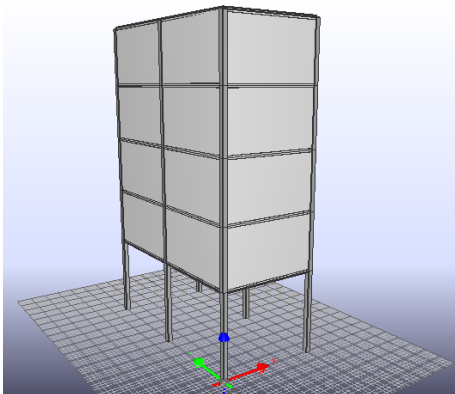


Abb. 5.2.: Darstellung des Referenzobjektes mit Rechteckstützen und Wänden im IFC-Viewer

Der compilierte Code erzeugt nach dem Starten das Modell und speichert es im STEP-Format ab.

6. Schema-Analyse

6.1. Zweck einer Schema-Analyse

In Kapitel 1.3 wurde angesprochen, dass die Beziehungen der Datenstrukturen untereinander durch eine Potenzmenge ausgedrückt werden können. Das bedeutet, erhöht sich die Anzahl der Datenstrukturen eines Schemas, explodiert die Anzahl der Beziehungen förmlich. Natürlich ist im IFC-Schema bei Weitem nicht jedes Entity-Objekt durch eine Beziehung mit anderen Entities verknüpft. Dies verringert die Beziehungsmenge. Allerdings gibt es in der IFC auch Konstrukte, in denen zwischen zwei Entities mehr als eine Beziehung besteht. Beispielsweise benötigt eine Platzierung im 3D-Raum eine Position und eine Ausrichtung. Die Position wird durch ein Punkt-Entity vom Typ `IfcCartesianPoint` realisiert. Die Ausrichtung wird über zwei Entities vom gleichen Typ `IfcDirection` definiert. Das heißt, zum gleichen Entity existieren zwei Verbindungen. Konstrukte wie diese vergrößern die Menge der Verknüpfungen wieder beträchtlich.

Mit einer Schema-Analyse schränkt man die Anzahl der Beziehungen schrittweise durch Anwendung von Regeln ein. Die Art und Menge der Regeln hängt vom jeweiligen Kontext des Anwendungsgebietes ab. Darauf wird genauer in Kapitel 6.3.1.1 im Absatz zur Reduktion eingegangen.

6.2. Der EXPRESS-Analyzer

Um die Beziehungen zwischen den Klassen der IFC zu untersuchen, wurde in Java der EXPRESS-Analyzer entwickelt. Er liest die Definition der IFC, die in einer Datei im EXPRESS-Format vorliegen, ein und parst den Code. Dafür wurde mit der Java-Erweiterung *JavaCC*¹ ein leistungsfähiger Open-Source-Parser-Generator genutzt. Da das Thema Compilerbau kein zentraler Bestandteil dieser Arbeit ist, soll nicht weiter an dieser Stelle darauf eingegangen werden. Die Programmierung erforderte allerdings einen nicht unerheblichen Zeitaufwand. Daher findet sich als Ergebnis der Bemühungen die Backus-Naur-Form des Parsers in Anhang B.

Wurde das Schema fehlerfrei eingelesen und verarbeitet, kann optional noch eine Zusatzdatei eingelesen werden, in der Entities gelistet sind, die aktiviert oder deaktiviert

¹<http://javacc.java.net>

6. Schema-Analyse

werden sollen. Standardmäßig sind sie aktiviert, aber für Analysen mit Randbedingungen ist es von Vorteil bestimmte Entities auszublenden.

Der Nutzer ist nun in der Lage, über Eingabezeile verschiedene Anfragen und Kommandos zu verarbeiten:

- Informationen über Entities können angezeigt werden. Welche (globalen) Attribute welcher Art besitzen sie, wie ist die Vererbungsstruktur, werden Regeln definiert?
- Die rekursive Verknüpfung der Attribute kann als Baumstruktur angezeigt werden.
- Die Vererbungshistorie eines Entities kann ausgegeben werden.
- Die Anzahl und Listen möglicher Attributkombination kann über Potenzmengen berechnet, angezeigt und abgespeichert werden. Dabei hat man die Wahl, ob die Kombinationsmenge hinsichtlich der nichtoptionalen Attribute bereinigt werden soll, also ob nur die Kombinationen ausgewählt werden, in denen die „Pflicht“-Attribute enthalten sind.

Exemplarisch folgt ein Auszug aus dem Screenlog eines Laufes des Analyzers, in dem zuerst alle Attribute der Klasse `IfcRectangleProfileDef`, welche ein Rechteckprofil verkörpert, abgefragt wurde. In diesem Fall wurden vier Typen und ein Entity angezeigt.

```
> eaattrgl ifcrectangleprofiledef
All attributes for IfcRectangleProfileDef (*optional @abstract #inactive)
[T] ProfileType = IfcProfileTypeEnum
[T] ProfileName = IfcLabel *
[T] XDim = IfcPositiveLengthMeasure
[E] Position = IfcAxis2Placement2D
[T] YDim = IfcPositiveLengthMeasure
```

Betrachtet man sich nun einmal die rekursive Struktur der Entity-Attribute, die hier - zugegeben - sehr übersichtlich ist, sieht man, dass das Rechteckprofil ein Attribut für die Platzierung im zweidimensionalen Raum besitzt, die ihrerseits wieder zwei Entity-Attribute, nämlich Koordinatenposition und Ausrichtung enthält, die zudem optional ist.

```
> etreer ifcrectangleprofiledef
Resolved entity tree for IfcRectangleProfileDef (*optional @abstract #inactive <>ent.select)
IfcAxis2Placement2D
  IfcCartesianPoint
  IfcDirection *
```

Der Nutzen für die Schema-Analyse wird ersichtlich, wenn man die Menge aller Relationskombinationen abfragt. Man kann dabei ignorieren, ob die Attribute optional sind:

6. Schema-Analyse

```
> erel ifcrectangleprofiledef
[IfcRectangleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint]
[IfcRectangleProfileDef, IfcAxis2Placement2D, IfcDirection]
[IfcRectangleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint, IfcDirection]
[IfcRectangleProfileDef, IfcAxis2Placement2D]
[IfcRectangleProfileDef]
Total relation count for IfcRectangleProfileDef: 5
```

Sinnvoller ist es aber meistens, es zu berücksichtigen:

```
> erelr ifcrectangleprofiledef
[IfcRectangleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint]
[IfcRectangleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint, IfcDirection]
Total restricted relation count for IfcRectangleProfileDef: 2
```

Auf diese Weise lassen sich alle gültigen Attributkombinationen ermitteln, die es für das Entity `IfcRectangleProfileDef` gibt.

Der Analyzer ist in der Lage, abstrakte Klassen weiter aufzulösen und Select-Typen nach weiteren Entities zu durchsuchen. Obwohl in das Programm eine Endlosschleifenprüfung integriert wurde, kann es durch die starke Strukturierung der IFC des Öffneren passieren, dass die Rechen- und Speicherkapazitäten des Computers überschritten werden. Gerade das Generieren von Kombinationslisten führt schnell zu einem Abbruch des Programms. Daher erfordert es Sorgfalt und Kenntnis des Nutzers über die IFC, um erfolgreiche Analysen mit dem Analyzer durchzuführen. Aus diesem Grund wurde auch das Prinzip der inaktiven Entities eingeführt.

Das Programm ist sicherlich nicht perfekt und kann durchaus noch erweitert werden. Regeln werden beispielsweise noch nicht verarbeitet. Für die Schema-Analysen in den folgenden Kapiteln war es jedoch sehr hilfreich.

6.3. Analyse

6.3.1. Geometrie

6.3.1.1. Randbedingungen

Es werden nur die Entities betrachtet, die wirklich benötigt werden. Von 653 sind das 56 Entities, wobei in dieser Menge schon ein Overhead enthalten ist, um eine logische Hierarchiestruktur zu definieren. Diese ist nötig, um das Element mit einem parametrischen Profil in einem IFC-Betrachter darstellen zu können. Diese Entities sind für die eigentliche schematische Betrachtung der Säule unwichtig.

Die folgenden, eigentlich obligatorischen Entities werden zusätzlich vernachlässigt, um die Analyse auf die relevanten Attribute zu fokussieren:

6. Schema-Analyse

- IfcOwnerHistory wird weggelassen. Für die Modellierung selbst sind die Informationen dieses Abschnittes irrelevant. Darin stehen Daten über Bearbeiter, Zeit und Status des jeweiligen IFC-Objektes.
- IfcRepresentationContext bzw. IfcGeometricRepresentationContext wird weggelassen. Im Normalfall ist für ein IFC-Modell solch ein Kontext vorgesehen, um zu definieren, in welcher Form das Modell dargestellt werden soll (direkt als Modell, als ausdrückbare Zeichnung oder anderes). Weiterhin ist eine Anbindung an ein übergeordnetes Weltkoordinatensystem enthalten. Dies spielt für die Konvertierung nach Ansys keine Rolle.

Die Reduktion erfolgt in drei übergeordneten logischen Abschnitten. Als erstes werden die Schema Rules angewandt. Dies sind Regeln, die in der Definition der IFC verankert sind. Danach erfolgt die Reduktion durch die globale Modellierung. Dies sind Einschränkungen, die sich aus den Randbedingungen ergeben, die im Rahmen der Arbeit aufgestellt wurden. Zuletzt erfolgt die Reduktion infolge lokaler Modellierung. Hier wird im Wesentlichen durch die Beschränkung auf modellierte IFC-Klassen noch einmal die Anzahl der Kombinationen verringert.

6.3.1.2. IfcColumn

Die folgenden Schritte führen zu einem umsetzbaren Maß an Kombinationen von Parametern, die zur Beschreibung einer Säule durch die IFC nötig sind.

A - Schema Rules

Schritt 1: Betrachtet man alle möglichen Kombinationen der Relationen, die sich mit Hilfe der benutzten Entities von IfcColumn ausgehend bilden lassen, ergeben sich 159165 Kombinationen. Dieser Schritt wird mit Hilfe des Analysetools vollzogen, das für die Untersuchung der EXPRESS-Definition geschrieben wurde.

Schritt 2: Es gibt obligatorische und optionale Attribute. Letztere werden im EXPRESS-Schema mit OPTIONAL gekennzeichnet. Es müssen in allen Kombinationen alle obligatorischen Attribute vorhanden sein.

Nach Herausfiltern bleiben noch 7085 Kombinationen.

Schritt 3: Damit eine Säule darstellbar ist, benötigt die Säule ein IfcProductRepresentation (definiert in IfcProduct, einer Superklasse von IfcColumn). Nach [MSG07] muss laut Regel WR1 in IfcProduct jedes Objekt, das ein IfcProductRepresentation zudem ein IfcObjectPlacement besitzen. Kombinationen, die diesen Anforderungen nicht genügen, werden gestrichen.

Es bleiben 6528 Kombinationen.

6. Schema-Analyse

Schritt 4: Zur Beschreibung der Säule wird `IfcProductDefinitionShape` benutzt, eine Unterklasse von `IfcProductRepresentation`. Letztere ist nicht abstrakt und taucht deswegen auch als mögliche Klasse in den Kombinationen auf. Laut Hinweis in der Dokumentation wird `IfcProductDefinitionShape` ab der nächsten IFC-Version zur abstrakten Klasse. Kombinationen mit dieser Klasse werden eliminiert.

Analog ist es mit dem Attribut `IfcRepresentation`, welches durch die Subklasse `IfcShapeRepresentation` spezifiziert und deswegen aus den Kombinationen eliminiert wird. Auch diese Klasse wird ab der nächsten IFC-Version als abstrakt erklärt.

Es verbleiben 1632 Kombinationen.

Schritt 5: Weil mit dreidimensionalen Objekten gearbeitet wird, braucht man dementsprechend auch einen dreidimensionalen Raum. `IfcLocalPlacement` benötigt nur `IfcAxis2Placement3D`, nicht `IfcAxis2Placement2D`.

Es verbleiben 1088 Kombinationen.

Schritt 6a: Die Position der Säule wird durch einen Punkt im dreidimensionalen Raum sowie die Festlegung zweier Bezugsachsen bestimmt. Letztere können entfallen, wenn die Ausrichtung der Säule dem Standardkoordinatensystem entspricht. Weiterhin ist in `IfcAxis2Placement3D` als Regel definiert, dass entweder beide oder keine Achsendefinition vorliegen sollen.

Dadurch halbiert sich die Anzahl der Kombinationen auf 544.

Schritt 6b: `IfcExtrudedAreaSolid` benötigt eine Positionierung durch `IfcAxis2Placement3D`, die wiederum durch die Parameter `IfcCartesianPoint` und `IfcDirection` bestimmt ist. Interessant ist dies in Fällen, bei denen die `IfcRepresentation` mehrere `IfcRepresentationItem` enthält, die relativ zu dem Placement des eigentlichen Objektes ausgerichtet sind. Auch hier greift die Regelung zur Achsendefinition, die schon bei Schritt 6a angewandt wurde.

Somit bleiben 288 Kombinationen übrig.

Schritt 7: Die Definition zum Profil für `IfcColumn` nach [MSG07] schränkt die Auswahl möglicher Unterklassen von `IfcParameterizedProfileDef` ein.

Dies verringert die Anzahl der Kombinationen auf 176.

B - Globale Modellierung

Schritt 8: `IfcLocalPlacement` braucht keinen relativen Bezugspunkt, da die Säule global positioniert wird.

Damit halbiert sich die Menge der Kombinationen auf 88.

Schritt 9a: Im Prinzip ist eine Säule nur senkrecht extrudiert sinnvoll, allerdings kann man sie auch um die senkrechte Achse drehen. Aus diesem Grund werden beide

6. Schema-Analyse

IfcDirection-Parameter von IfcAxis2Placement3D in IfcLocalPlacement mit berücksichtigt.

Es verbleiben 44 Kombinationen.

Schritt 9b: Auch das IfcAxis2Placement3D von IfcExtrudedAreaSolid lässt sich vereinfachen. Da die Ausrichtung schon über das LocalPlacement erfolgt und nur ein Item vorliegt, ist somit nur der obligatorische Parameter für IfcCartesianPoint notwendig, der logischerweise (0.0,0.0) enthält. Die beiden IfcDirection-Parameter entfallen.

Somit bleibt 26 Kombinationen übrig.

Schritt 9c: Das Attribut IfcAxis2Placement2D in jedem der IfcParameterizedProfileDef-Entities benötigt einen Punkt und eine Richtung zum Positionieren der Rechteckgrundfläche. Wie in Schritt 9b ist das Placement im Prinzip unnötig und der optionale IfcDirection-Parameter kann weggelassen werden.

Es bleiben 17 Kombinationen übrig.

C - Lokale Modellierung

Schritt 10: Als Parameter von IfcShapeRepresentation ist nur IfcExtrudedAreaSolid relevant, auch wenn unter den 56 Entities andere geometrische Objekte wie IfcDirection, IfcCartesianPoint, IfcAxis2Placement2D oder IfcAxis2Placement3D dazugehören, die allerdings in dem Kontext überhaupt keinen Sinn ergeben.

Somit verbleiben 9 Kombinationen.

Schritt 11: Die Säule soll eines von drei möglichen Grundrissprofilen annehmen: IfcRectangleProfileDef, IfcIShapeProfileDef und IfcCircleProfileDef. Kombinationen mit anderen Parameterprofilen fallen somit heraus, dass nur noch 3 Kombinationen übrig bleiben.

Das bedeutet, dass Elemente, die im Referenzobjekt als IfcColumn modelliert werden, eine der drei folgenden Parameterkombinationen besitzen müssen:

```
[IfcColumn,  
  IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,  
  IfcProductDefinitionShape, IfcShapeRepresentation, IfcExtrudedAreaSolid,  
    IfcCircleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,  
    IfcDirection,  
  IfcAxis2Placement3D, IfcCartesianPoint]
```

```
[IfcColumn,  
  IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,  
  IfcProductDefinitionShape, IfcShapeRepresentation, IfcExtrudedAreaSolid,  
    IfcIShapeProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,  
    IfcDirection,
```

6. Schema-Analyse

```
IfcAxis2Placement3D, IfcCartesianPoint]
```

```
[IfcColumn,  
  IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,  
  IfcProductDefinitionShape, IfcShapeRepresentation, IfcExtrudedAreaSolid,  
    IfcRectangleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,  
    IfcDirection,  
    IfcAxis2Placement3D, IfcCartesianPoint]
```

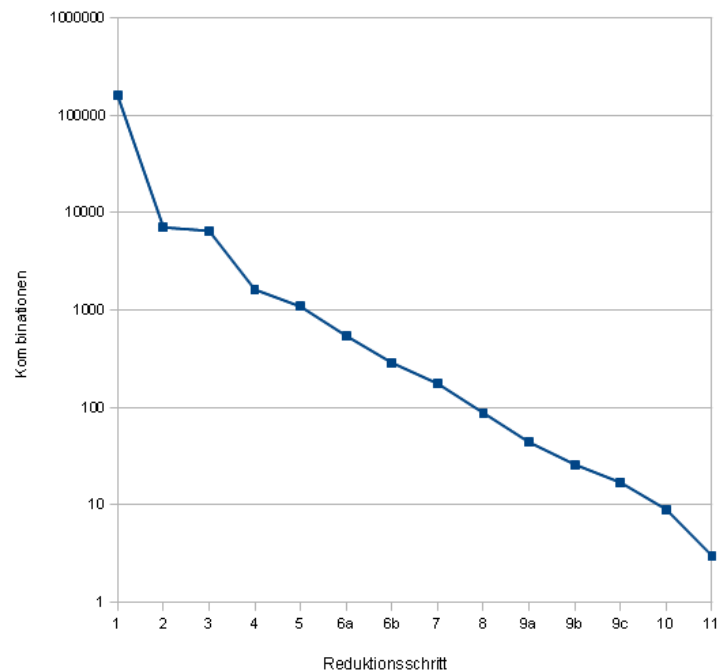


Abb. 6.1.: Reduktionsschritte der Schema-Analyse von IfcColumn. Hinweis: Die vertikale Skala ist logarithmisch eingeteilt.

Abbildung 6.2 beschreibt die Beziehungen der Klassen am Beispiel von IfcColumn mit einem Rechteckprofil untereinander auf einer detaillierteren Ebene, in der auch Vererbungshierarchien dargestellt werden.

6.3.1.3. Anwendung auf andere Elemente

Im Wesentlichen funktioniert die Analyse für die anderen ausgewählten Bauelemente analog, da sie allesamt Unterklassen von IfcBuildingElement sind. Lediglich die Auswahl der Profile im Schritt 11 ist unterschiedlich.

IfcBeam [MSG07] definiert einen Balken parametertechnisch wie eine Säule, nur dass seine Lage in der Regel horizontal verlaufen soll. Daher gibt es bei der Schema-Analyse

6. Schema-Analyse

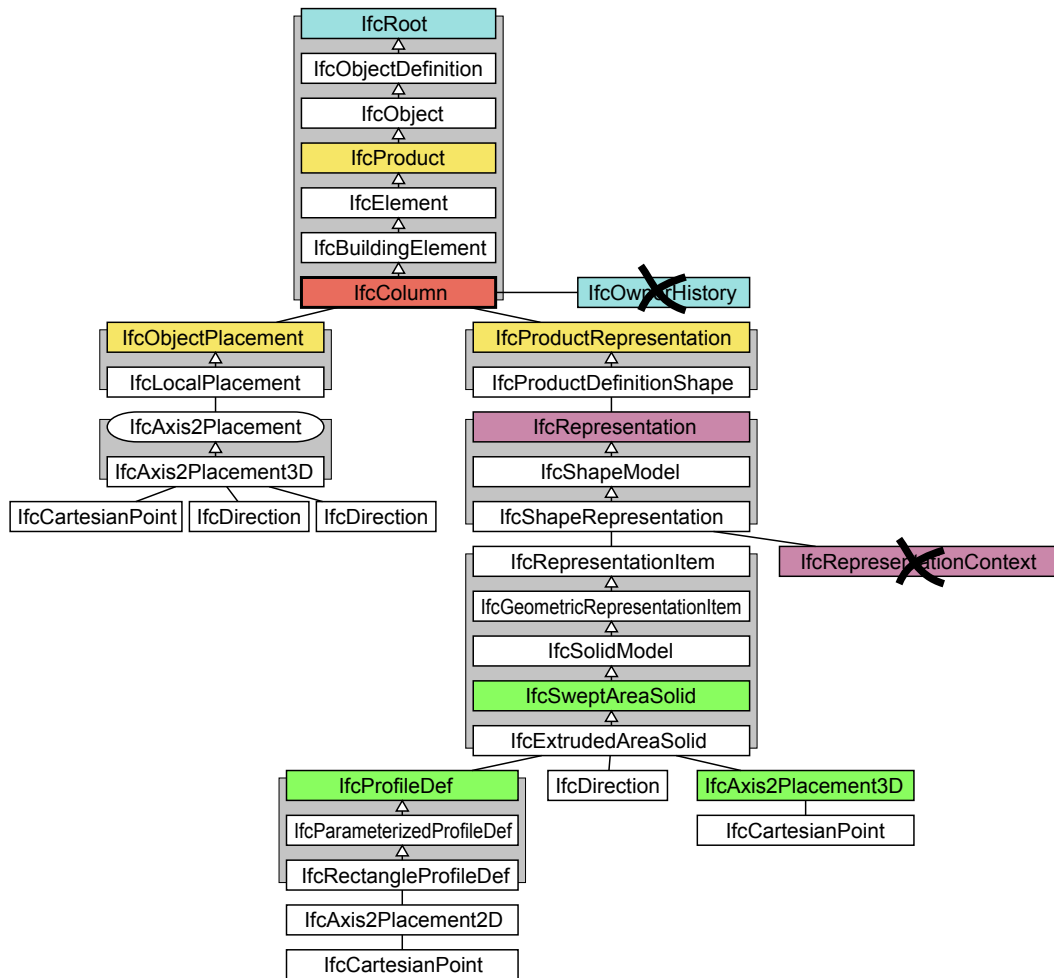


Abb. 6.2.: Schema von IfcColumn mit einem Rechteckprofil

zahlenmäßig keinen Unterschied, so dass wieder folgende drei Kombinationen übrig bleiben.

```
[IfcBeam,
  IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,
  IfcProductDefinitionShape, IfcShapeRepresentation, IfcExtrudedAreaSolid,
  IfcCircleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,
  IfcDirection,
  IfcAxis2Placement3D, IfcCartesianPoint]
```

```
[IfcBeam,
  IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,
  IfcProductDefinitionShape, IfcShapeRepresentation, IfcExtrudedAreaSolid,
  IfcIShapeProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,
  IfcDirection,
  IfcAxis2Placement3D, IfcCartesianPoint]
```

```
[IfcBeam,
  IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,
```

6. Schema-Analyse

```
IfcProductDefinitionShape, IfcShapeRepresentation, IfcExtrudedAreaSolid,  
    IfcRectangleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,  
    IfcDirection,  
    IfcAxis2Placement3D, IfcCartesianPoint]
```

IfcMember Dieses Strukturelement darf nach [MSG07] zwar alle parametrischen Profile besitzen, wird durch die lokale Modellierung jedoch auch auf die folgenden drei Varianten beschränkt.

```
[IfcMember,  
    IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,  
    IfcProductDefinitionShape, IfcShapeRepresentation, IfcExtrudedAreaSolid,  
        IfcCircleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,  
        IfcDirection,  
        IfcAxis2Placement3D, IfcCartesianPoint]
```

```
[IfcMember,  
    IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,  
    IfcProductDefinitionShape, IfcShapeRepresentation, IfcExtrudedAreaSolid,  
        IfcIShapeProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,  
        IfcDirection,  
        IfcAxis2Placement3D, IfcCartesianPoint]
```

```
[IfcMember,  
    IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,  
    IfcProductDefinitionShape, IfcShapeRepresentation, IfcExtrudedAreaSolid,  
        IfcRectangleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,  
        IfcDirection,  
        IfcAxis2Placement3D, IfcCartesianPoint]
```

IfcSlab Für Platten sind nach [MSG07] für parametrisierte Grundflächen Rechtecke und Kreise vorgesehen. Die lokale Modellierung schließt die Kreisfläche allerdings aus.

```
[IfcSlab,  
    IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,  
    IfcProductDefinitionShape, IfcShapeRepresentation, IfcExtrudedAreaSolid,  
        IfcRectangleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,  
        IfcDirection,  
        IfcAxis2Placement3D, IfcCartesianPoint]
```

IfcWallStandardCase Werden Wände in der vereinfachten Variante mit IfcWallStandardCase extrudiert modelliert, sind nach [MSG07] für Extrusionskörper nur rechteckige Grundflächen erlaubt.

```
[IfcWallStandardCase,  
    IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,  
    IfcProductDefinitionShape, IfcShapeRepresentation, IfcExtrudedAreaSolid,  
        IfcRectangleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,  
        IfcDirection,  
        IfcAxis2Placement3D, IfcCartesianPoint]
```

6. Schema-Analyse

Mit diesen Elementen in den vorliegenden Kombinationen ist es möglich, das Referenzmodell auf geometrischer Ebene als IFC-Modell abzubilden. Es muss natürlich eine logische Gebäudestruktur nach 3.3.4 vorhanden sein, damit das Modell in einem IFC-Viewer betrachtbar ist.

6.3.2. Material

Die Verwendung eines Materials geht nicht vom Material selbst aus, sondern von der Relation `IfcRelAssociatesMaterial` zwischen Element und Material, die eine vergleichsweise einfache Struktur besitzt. Daher ist die Anzahl der möglichen Anfangskombinationen auch deutlich geringer.

Die Randbedingungen werden ähnlich denen der Geometrie definiert. Die Anzahl der zu untersuchenden Entities beschränkt sich hierbei jedoch nur auf 34. `IfcOwnerHistory` wird ebenfalls weggelassen. Eine weitere Einschränkung ist, dass nur Bauelemente, also von `IfcBuildingElement` abgeleitete Objekte, mit Materialien verknüpft werden sollen. Davon wird zudem auch `IfcBuildingElementComponent` samt Unterklassen ausgeschlossen, da diese Klassen laut Definition keine ganzheitlichen Teile beschreiben.

Die Reduktion erfolgt ebenfalls in den drei bekannten Abschnitten Schema Rules, globale Modellierung und lokale Modellierung.

A - Schema Rules

Schritt 1: Betrachtet man alle möglichen Kombinationen der Relationen, die sich mit Hilfe der benutzten Entities von `IfcRelAssociatesMaterial` ausgehend bilden lassen, ergeben sich 312 Kombinationen.

Schritt 2: Es gibt obligatorische und optionale Attribute. Letztere werden im EXPRESS-Schema mit `OPTIONAL` gekennzeichnet. Es müssen in allen Kombinationen alle obligatorischen Attribute vorhanden sein.
Nach Herausfiltern bleiben noch 176 Kombinationen.

Schritt 3: Nach Definition von Regel WR22 aus `IfcRelAssociatesMaterial` dürfen mit dem Material verknüpfte Elemente nur aus einer Unterklasse der Klasse `IfcProduct` (oder `IfcTypeProduct`, welche aufgrund der Randbedingungen irrelevant ist) stammen. Somit verbleiben 160 Kombinationen.

B - Globale Modellierung

Schritt 4: Jedes Bauelement soll aus genau einem Material bestehen, zudem werden in Definitionen der Bauelemente wie z.B. bei `IfcColumn` oder `IfcBeam` die Möglichkeiten aus `IfcMaterialSelect` sowieso auf `IfcMaterial` und `IfcMaterialList` beschränkt.

6. Schema-Analyse

Für diese Analyse soll nur IfcMaterial genutzt werden.
Es bleiben 20 Kombinationen übrig.

C - Lokale Modellierung

Schritt 5: Die Bauelementklassen werden auf die exemplarisch im Projekt umgesetzten Klassen beschränkt.
So bleiben letztendlich 5 Kombinationen übrig.

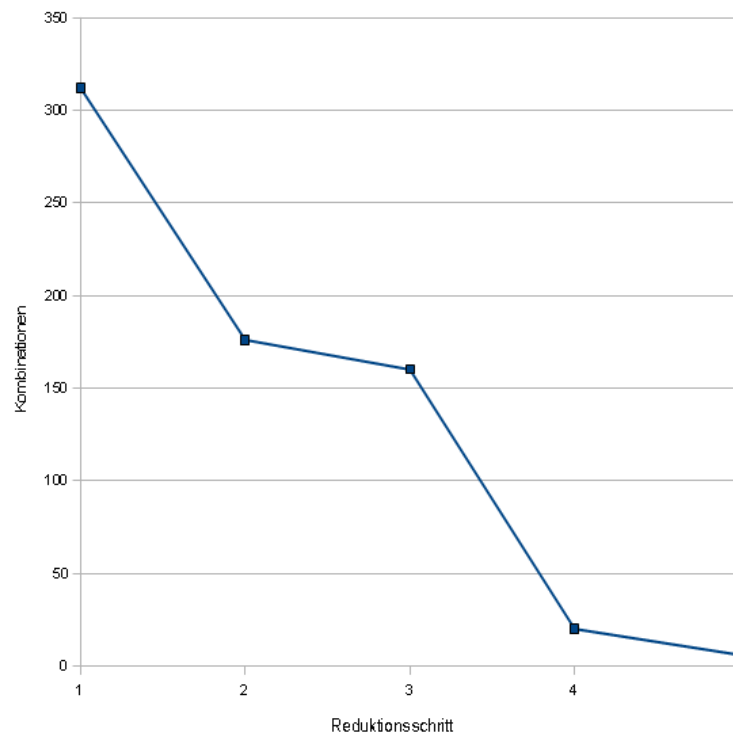


Abb. 6.3.: Reduktionsschritte der Schema-Analyse von IfcRelAssociatesMaterial

6. Schema-Analyse

Es müssen folgende fünf Parameterkombinationen definiert werden:

```
[IfcRelAssociatesMaterial,  
  IfcMaterial,  
  IfcColumn]
```

```
[IfcRelAssociatesMaterial,  
  IfcMaterial,  
  IfcBeam]
```

```
[IfcRelAssociatesMaterial,  
  IfcMaterial,  
  IfcMember]
```

```
[IfcRelAssociatesMaterial,  
  IfcMaterial,  
  IfcWallStandardCase]
```

```
[IfcRelAssociatesMaterial,  
  IfcMaterial,  
  IfcSlab]
```

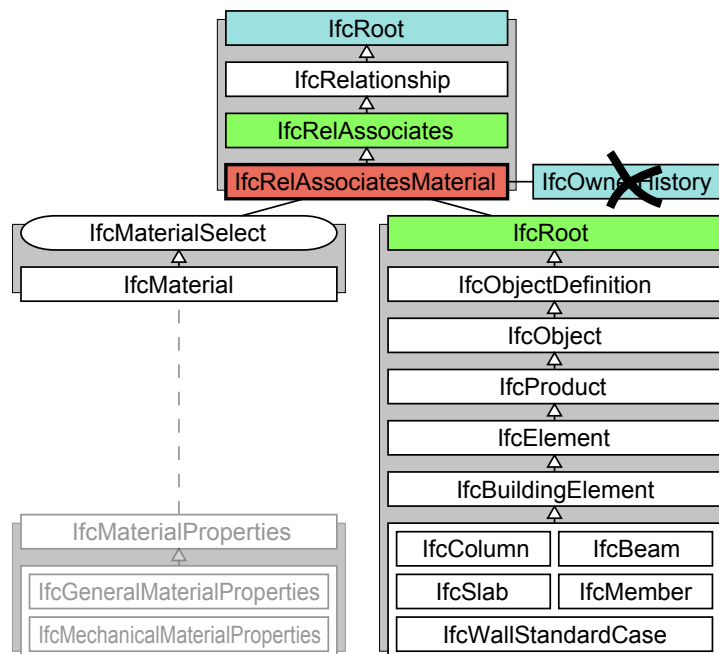


Abb. 6.4.: Objektdiagramm für Materialrelationen mit den möglichen Bauelementen. Der Vollständigkeit halber ist darin auch die indirekte Beziehung zu den zusätzlichen Materialeigenschaften enthalten, die kein Bestandteil der Schema-Analyse ist.

6.3.3. Strukturmodell

Für die das geometrische Modell ergänzenden Elemente aus der Structural Analysis Domain sollen hier nur die Ergebnisse dargestellt werden. Der Aufbau des Auflagerknotens und der Punktlast sind, aus der Perspektive der IFC-Struktur gesehen, ähnlich. Die Geometrie beschränkt sich aufgrund ihres Charakters auf einen Punkt im Koordinatensystem. Die Koordinaten beziehen sich relativ zu denen der Platzierung.

```
[IfcStructuralPointConnection,
  IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,
  IfcProductDefinitionShape, IfcTopologyRepresentation, IfcVertexPoint,
    IfcCircleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,
      IfcCartesianPoint,
    IfcBoundaryNodeCondition]
```

```
[IfcStructuralPointAction,
  IfcLocalPlacement, IfcAxis2Placement3D, IfcCartesianPoint,
  IfcProductDefinitionShape, IfcTopologyRepresentation, IfcVertexPoint,
    IfcCircleProfileDef, IfcAxis2Placement2D, IfcCartesianPoint,
      IfcCartesianPoint,
    IfcStructuralLoadSingleForce]
```

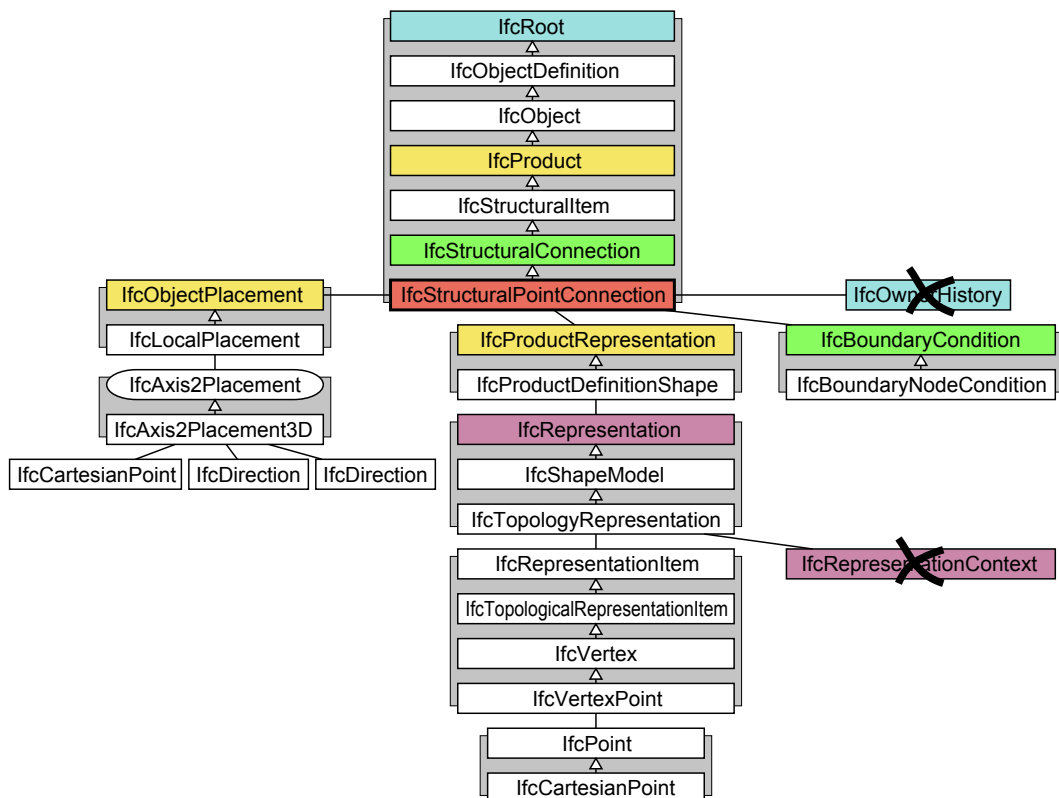


Abb. 6.5.: Schema von IfcStructuralPointConnection

6. Schema-Analyse

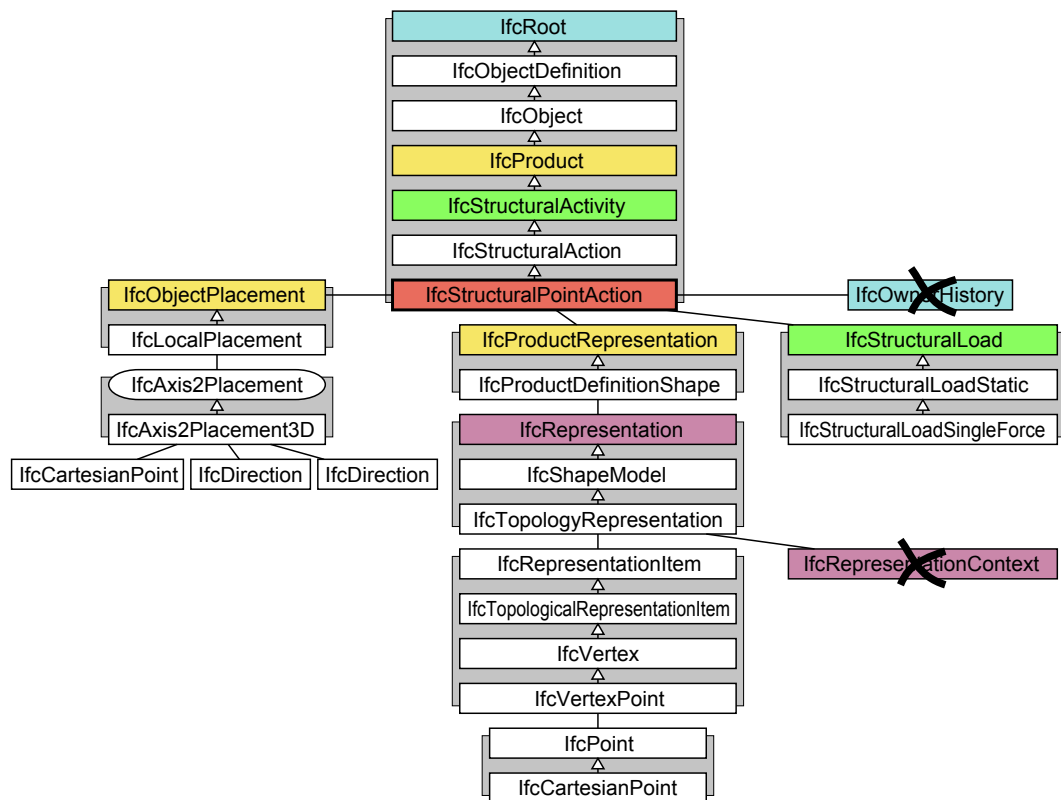


Abb. 6.6.: Schema von IfcStructuralPointAction

7. Schema-Mapping

7.1. Generelles

Der Konverter wurde, wie die anderen selbst erstellten Programm auch, in Java geschrieben. Er greift ebenfalls auf das Open IFC Toolkit zurück, um auf die IFC-Modelle zugreifen zu können. Die Konvertierung eines Modells besteht aus zwei Phasen. In der ersten Phase werden das Modell geladen und die gesuchten Objekte herausgefiltert, untersucht, validiert und gegebenenfalls sortiert. Dabei werden Containerobjekte erzeugt, die die nötigen Parameter in primitiver und direkter Form enthalten. In der zweiten Phase wird aus den Containerobjekten der APDL-Code erzeugt und in eine Datei geschrieben.

Phase A - Aufbau einer sequenziellen Datenstruktur

1. Es erfolgt ein rekursiver Durchlauf der Gebäudestruktur. Man kann sich die Struktur wie einen Baum vorstellen, an dessen Ästen die gesuchten Bauelemente sitzen. Abbildung 3.8 aus Kapitel 3.3.4 stellt einen solchen Baum beispielhaft dar. Jedes Bauelement wird auf seine Parameter untersucht und muss in deren Zusammensetzung einer der ermittelten Varianten aus der Schema-Analyse entsprechen. Ist dies nicht der Fall, wird das entsprechende Bauteil nicht konvertiert. Ein Containerelement wird erzeugt und in die dafür vorgesehene Liste eingefügt. Außerdem wird noch eine Liste der verwendeten Bauelementtypen (z.B. IfcBeam) angelegt.
2. Alle Materialeigenschaften definierende Klassen werden durchlaufen. Für jedes Material, das sie referenzieren, wird ein Containerobjekt angelegt, die relevanten Materialeigenschaften übergeben und das Containerobjekt in eine Liste übernommen.
3. Alle Auflager vom Typ IfcStructuralPointConnection werden durchlaufen, die Attribute in einem Containerobjekt gespeichert und das Objekt in eine Liste übernommen.
4. Alle Lasten vom Typ IfcStructuralPointAction werden durchlaufen, die Attribute in einem Containerobjekt gespeichert und das Objekt in eine Liste übernommen.
5. Die List der Containerobjekte für Bauelemente wird sortiert. Lineare Elemente wie Stützen, Balken und Zugstäbe werden nach vorne, Platten und Wände nach hinten

7. Schema-Mapping

geschoben. Die Reihenfolge spielt für die ordnungsgemäße Modellierung in Ansys eine Rolle.

Phase B - Generierung des APDL-Codes

6. Die Liste der Material-Containerobjekte wird abgearbeitet. Für jedes Material wird in Ansys ein Materialtyp erstellt.
7. Die Liste der Bauelementtypen wird abgearbeitet. Für jeden IFC-Bauelementtyp wird in Ansys ein Elementtyp erzeugt. Tabelle 7.1 zeigt die Zuordnung.
8. Die Liste der Bauelemente wird abgearbeitet. Für jedes Bauelement wird in Ansys temporär ein Koordinatensystem erzeugt und aktiviert, eine Extrusion für einen bestimmten Grundriß erzeugt, den zugehörigen Element- und Materialtyp übergeben und die Geometrie schließlich vernetzt. Die detaillierte Generierung hängt vom Bauteiltyp ab. Eine genauere Beschreibung dazu findet sich im nächsten Kapitel.
9. Die Liste der Containerobjekte für Auflager wird abgearbeitet. Ansys sucht den nächstliegenden Elementknoten und legt die Freiheitsgrade entsprechend der Parameter fest.
10. Die Liste der Containerobjekte für Lasten wird abgearbeitet. Ansys sucht den nächstliegenden Elementknoten und legt den Lastvektor entsprechend der Parameter fest.

Innerhalb der und zwischen den Schritten wurden in den APDL-Code noch entsprechende Steuerbefehle eingefügt, hauptsächlich um Nummerierungen zu ermitteln und zu setzen oder zwischen den Ansys-Bearbeitungsabschnitten zu wechseln.

IFC-Entity	Ansys-Element
IfcColumn	BEAM188
IfcBeam	BEAM188
IfcMember	LINK10
IfcWallStandardCase	SHELL181
IfcSlab	SHELL181

Tab. 7.1.: Mapping der Elementtypen

Die Vernetzung und damit einhergehende Erzeugung von FEM-Elementen in Ansys wird im IFC-Modell nicht beschrieben. Es ist ein Feature des Konverters, der über einen Parameter die Feinheit des Netzes bestimmen kann. Dieser bestimmt die Anzahl der Unterteilungen der einzelnen Bauteile in Elemente¹. Für Skelettbauten wie das Referenzobjekt, das in einer Rastergeometrie vorliegt, ist dies im Rahmen dieser Arbeit ausreichend.

¹Ausgenommen davon ist das Ansys-Element LINK10. Es ist als Zugstab konzipiert, gelenkig angeschlossen und sollte nicht unterteilt werden.

7. Schema-Mapping

Um die einzelne Bauteile miteinander zu verbinden, wurde im Skript ein Mechanismus implementiert, der die Position von End- bzw. Eckpunkten der Bauteile mit schon vorhandenen Punkten vergleicht. Ist die Differenz zweier Punkte kleiner als ein vorgegebener Wert ε , teilen sich die entsprechenden Bauteile den Punkt und gelten als miteinander verbunden. ε ist im Konverter fest mit 0.0001 vorgegeben, wird von ihm am Anfang des Skripts in Ansys definiert und soll vorrangig Ungenauigkeiten bei Koordinatentransformationen ausgleichen.

Das physikalische Modell mit angebrachten Bindungen und Lasten ist im Prinzip das Ergebnis der Konvertierung. Ein weiteres Feature des Konverters bietet allerdings noch die Option, um die Statikberechnung des Modells anzustoßen. Ist diese abgeschlossen, wird noch eine passende Ansicht mit den Verrückungen erzeugt und das APDL-Skript ist abgearbeitet.

7.2. Geometrie

Im Folgenden werden die Mapping-Schemata der Bauelemente mit ihren relevanten Parametern dargestellt. Dabei wird einer IFC-Klasse mit ihren Eigenschaften mehr oder weniger direkt ein APDL-Befehl zugeordnet.

Die Behandlung der Positionierung ist für alle Bauteile gleich. Jedes Bauteil besitzt ein lokales Koordinatensystem, das sich durch einen Bezugspunkt und zwei Achsen definiert. Da für die Systeme die Rechte-Hand-Regel gilt, ist die Ausrichtung klar definiert. Dabei ist zu beachten, dass sich in den IFC auf die Z- und X-Achsen bezogen wird, während die Koordinatensysteme in Ansys durch die X- und Y-Achsen definiert werden. Der Konverter erzeugt also den gesuchten Vektor der Y-Achse durch die Bildung des Vektorprodukts von Z- und X-Achse. Mit diesen Informationen wird in Ansys über drei generierte Bezugspunkte ein lokales Koordinatensystem erzeugt und aktiviert.

Das Mapping für Stützen und Balken ist identisch. Der Fußpunkt bildet in der lokalen XY-Ebene den Mittelpunkt des Querschnittsprofils. Für die drei Querschnittsvarianten Rechteck, Kreis und I-Profil bietet Ansys über eine Bibliothek eine parametrisierte Generierung an. Die Parameter werden jeweils mit den selben Befehl übergeben. Der Körper wird entlang der Linie vom Fußpunkt bis zum Endpunkt erzeugt. Abbildung 7.1 stellt exemplarisch das Schema für eine Stütze mit Rechteckprofil dar.

Dem Zugstab kann aufgrund seines Elementtyps LINK10 nur die rechnerische Größe der Querschnittsfläche übergeben werden. Der Konverter rechnet diese anhand der aus dem IFC-Modell gewonnenen Querschnittsparameter aus und speichert sie Ansys-seitig in einer dem Stab zugeordneten Realkonstante. Das Schema wird in Abbildung 7.2 gezeigt.

Wände unterscheiden sich zwischen IFC und Ansys anhand ihrer Extrusionsrichtung.

7. Schema-Mapping

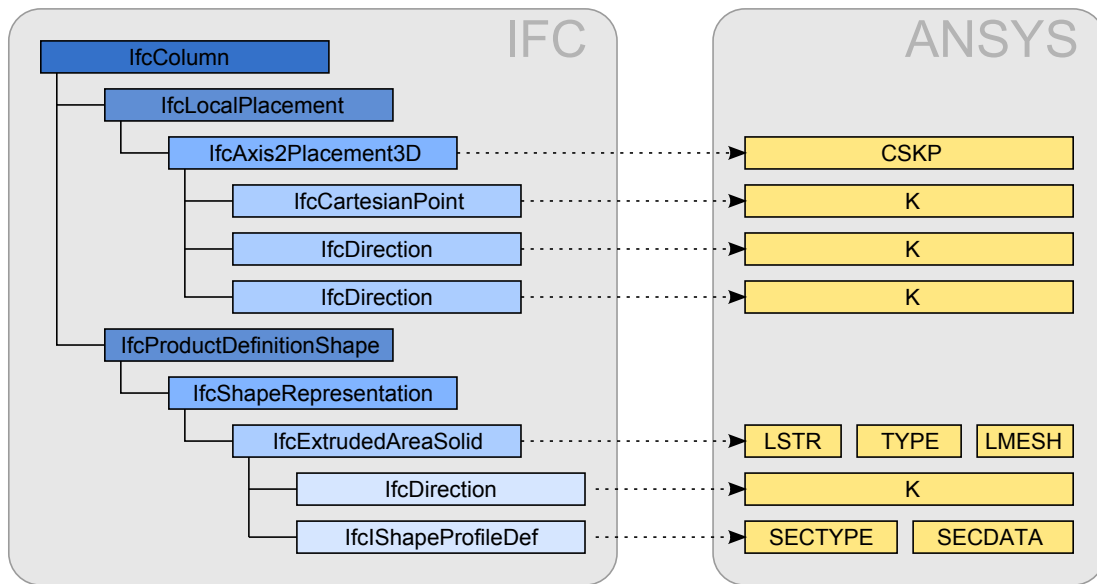


Abb. 7.1.: Mapping von IfcColumn mit I-Profil

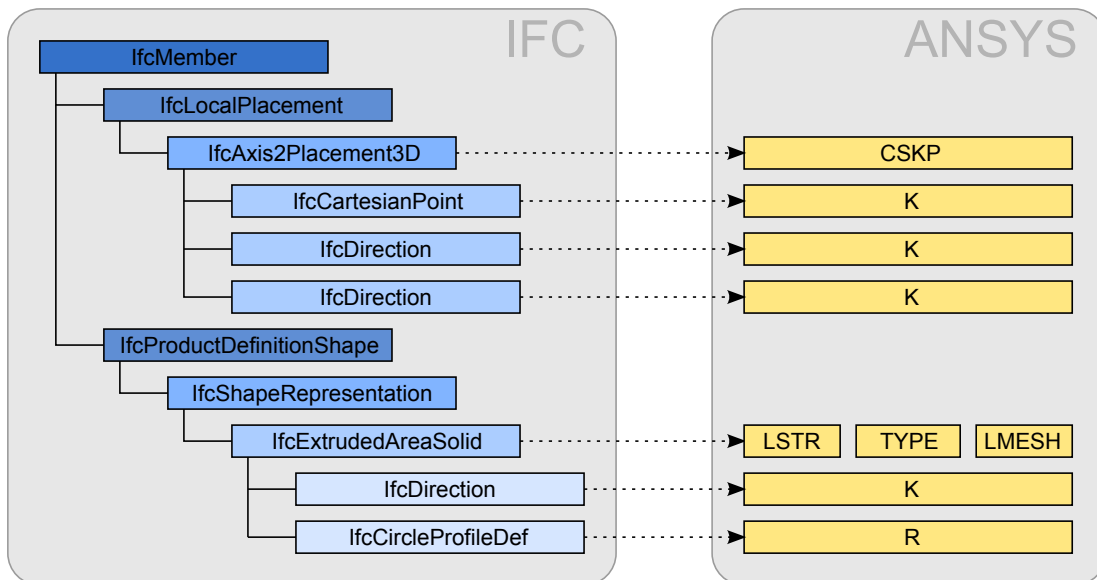


Abb. 7.2.: Mapping von IfcMember mit Kreisprofil

7. Schema-Mapping

Während nach IFC-Definition die Querschnittsfläche im Grundriß erzeugt und der Körper dann über die Wandstärke extrudiert werden soll, wird in Ansys das für das zugeordnete SHELL181-Element eine Fläche aus der Mittellinie und Wandhöhe erzeugt und über eine zugeordnete Realkonstante die Wandstärke angegeben. Für diesen Fall ist das Mapping für den Extrusionskörper nicht aufteilbar und muss als Ganzes abgehandelt werden. Abbildung 7.3 zeigt das entsprechende Schema.

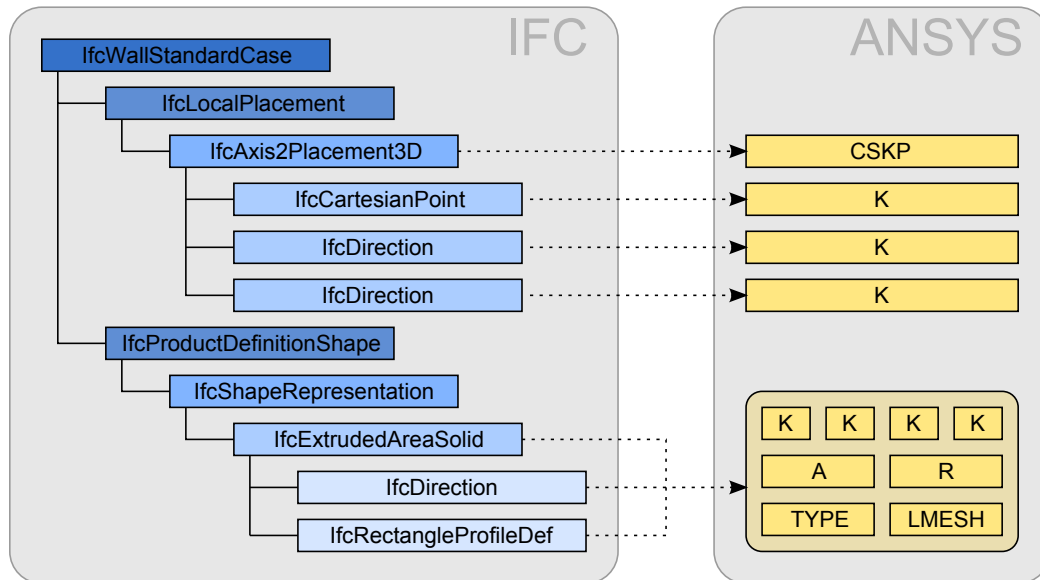


Abb. 7.3.: Mapping von IfcWallStandardCase

Die Platte entspricht in ihrem Ansys-Aufbau weitgehend dem der IFC. So wird die Grundfläche aus den Eckpunkten gebildet und die Plattendicke über eine Realkonstante definiert. Das Schema dazu ist in Abbildung 7.4 zu sehen.

7.3. Material

Da die Materialeigenschaften seitens der IFC für Geometriemodelle nur einseitig verknüpfbar sind, besitzt das Schema - zu sehen in Abbildung 7.5 - keine monolithische Struktur. Der Geometrie wird in Ansys die Referenznummer eines Materials gegeben, das vorher mit Materialeigenschaften versehen wurde. Elastizitätsmodul und Querdehnzahl stammen aus IfcMechanicalMaterialProperties, die Materialdichte aus IfcGeneralMaterialProperties.

7. Schema-Mapping

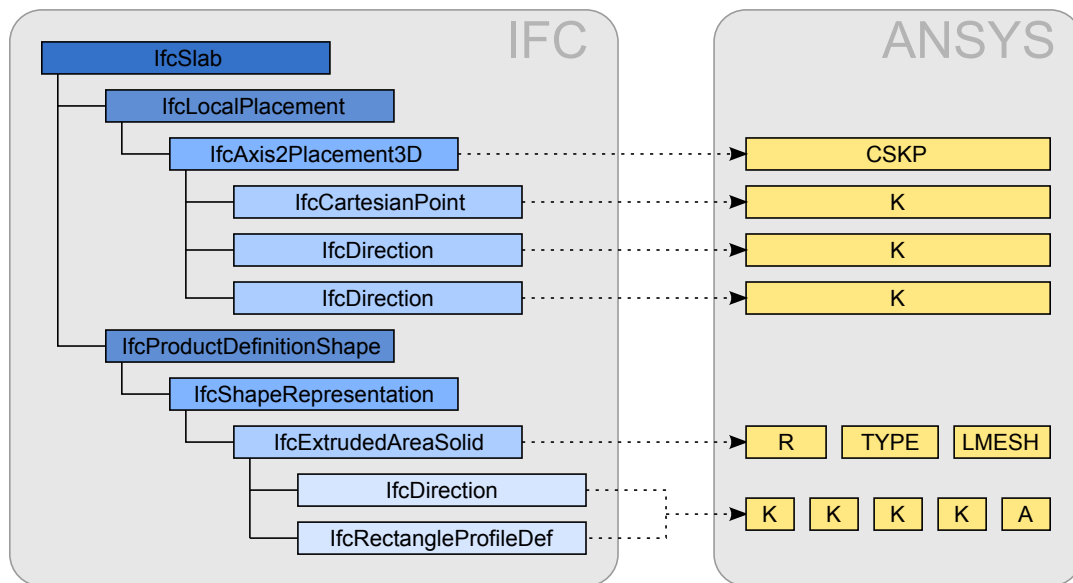


Abb. 7.4.: Mapping von IfcSlab

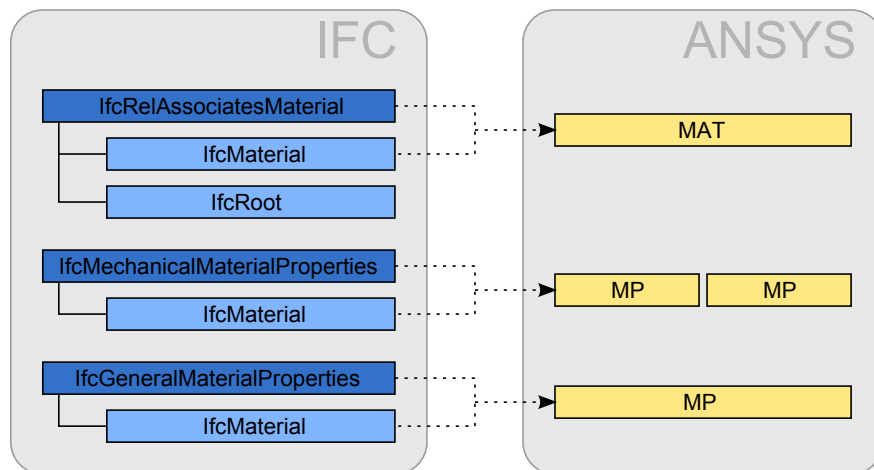


Abb. 7.5.: Mapping von Material

7.4. Strukturmodell

Die Komponenten der Structural Analysis Domain aus den IFC lassen sich mit den erforderlichen Parametern teilweise nur indirekt mappen. Die Positionierung im IFC-Schema erfolgt über den geometrischen Kontext in Form eines Punktes. In Ansys wird stattdessen am vernetzten Modell der nächstliegende Knoten zu den entsprechenden Koordinaten gesucht.

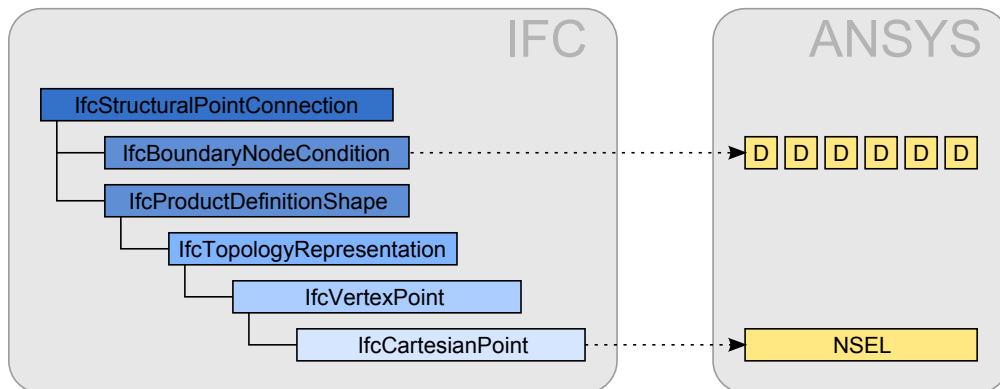


Abb. 7.6.: Mapping eines Auflagerknotens

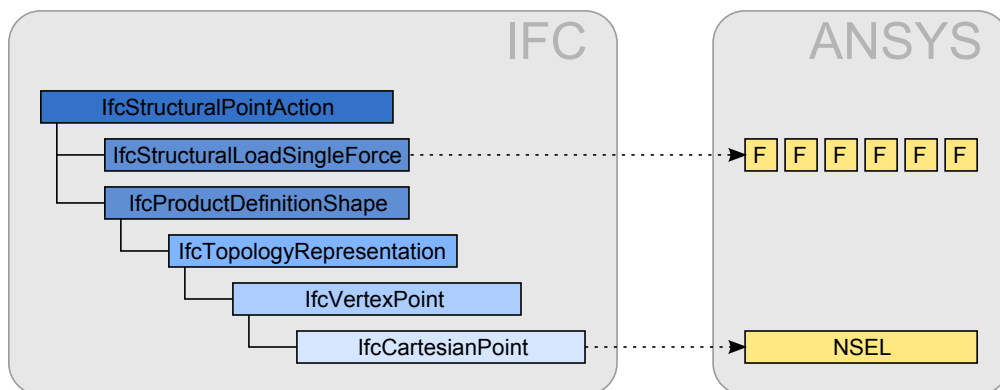


Abb. 7.7.: Mapping einer Punktlast

Soll ein Auflager modelliert werden, werden die sechs Freiheitsgrade am ermittelten Knoten mit den in `IfcBoundaryNodeCondition` enthaltenen Parametern definiert. Siehe dazu Abbildung 7.6. Während die Freiheitsgrade auf IFC-Seite durch Steifigkeiten ausgedrückt werden, passiert dies in Ansys durch Verschiebungswerte. Im Rahmen der Arbeit wurde sich auf komplett unverschiebliche und komplett verschiebliche Zustände beschränkt. Ein Freiheitsgrad mit quasi unendlicher Steifigkeit wird in den IFC mit dem Wert -1 ausgedrückt. Dies entspricht in Ansys einer Verschiebung von 0. Im unbeschränkten Fall ist durch die IFC der Wert gleich 0 zu setzen. In Ansys braucht dann der Einfachheit halber gar kein Wert gesetzt werden, da die Knoten dort standardmäßig

7. Schema-Mapping

verschieblich sind.

Handelt es sich um eine Punktlast, wird in Ansys das entsprechende Kommando zur Erzeugung einer Punktlast am Knoten mit den ermittelten Koordinaten benutzt. Das Schema wird in Abbildung 7.7 dargestellt.

7.5. Resultate

In Abbildung 7.8 werden die beiden Modellvarianten im IFC-Betrachter dargestellt.

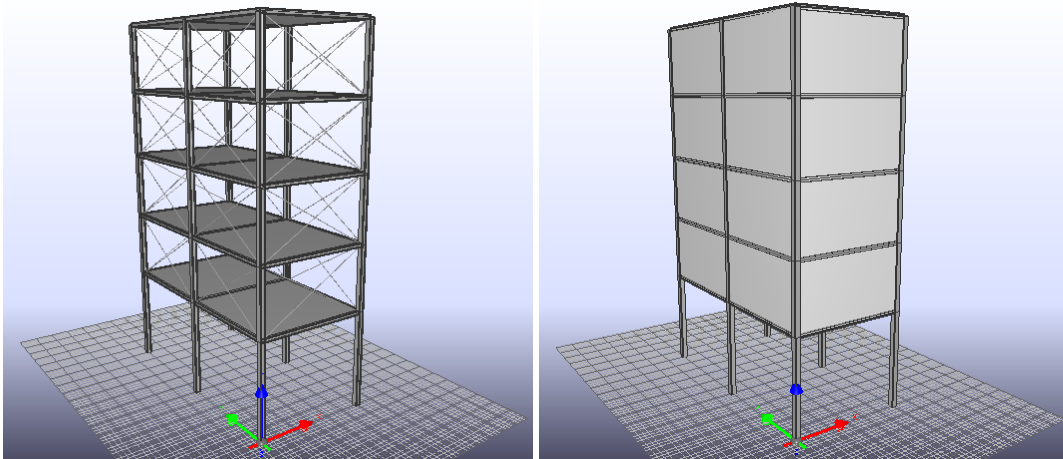


Abb. 7.8.: Ausgangsmodelle im IFC-Schema

Nach dem Mapping stehen die APDL-Dateien für die jeweiligen Modelle zur Verfügung. Abbildung 7.9 zeigt die Darstellung der Modelle, nachdem der APDL-Code in Ansys ausgeführt worden ist. Die Vernetzung wurde im Konverter auf fünf Elemente pro Seitenlänge eingestellt.

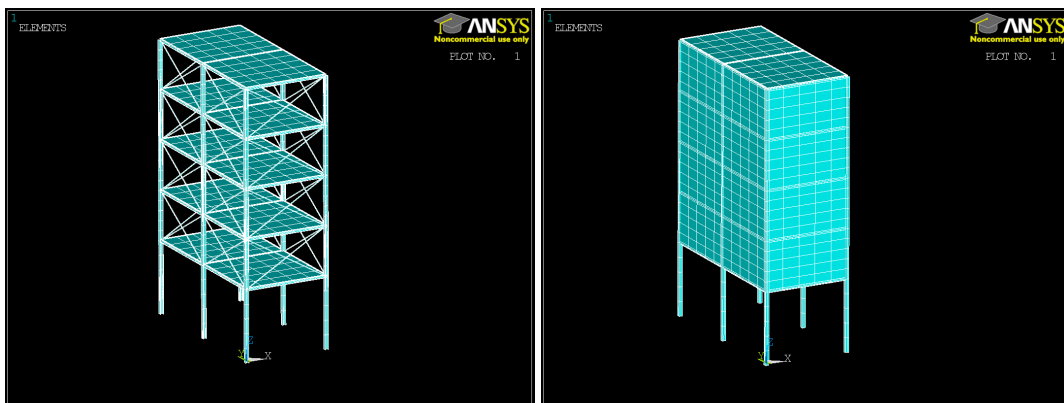


Abb. 7.9.: Modelle nach Mapping in Ansys

Die Darstellung der Verformungen aufgrund einer exemplarischen Punktlast in X-

7. Schema-Mapping

Richtung ist auch vom Konverter aus aktivierbar. In Abbildung 7.10 ist die Modelldarstellung mit aktivierter Verformungsanzeige zu sehen.

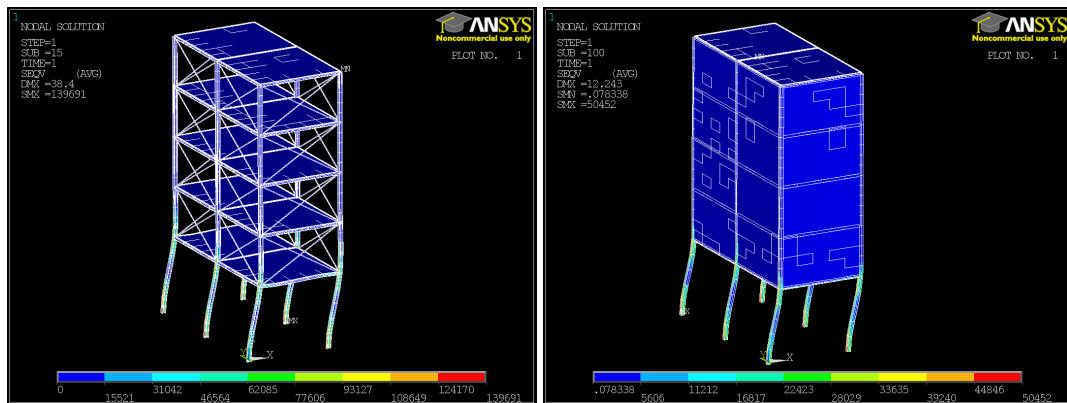


Abb. 7.10.: Modelle in Verformungsdarstellung in Ansys

8. Qualitative Bewertung

8.1. Bewertungsvarianten

Die Bewertung der Modelle wird nach dem Prinzip vorgenommen, wie es in Kapitel 1.4 beschrieben wurde. Das heißt, das vorhandene Schema-Mapping wird um Qualitätswerte zu einem bewerteten Schema-Mapping ergänzt. Jede Instanz des Modells erhält einen Wert für ihre Mapping-Qualität. Diese ermittelt sich wiederum aus dem Mittelwert der Übertragungsqualitäten der Attribute. Die Summe der Instanzen-Qualitätswerte geteilt durch die Anzahl der Instanzen ergibt dann den Wert für die Gesamtqualität.

Die Übertragungsqualität der Attribute lässt sich nach unterschiedlichen Kriterien einteilen. In dieser Arbeit wurden die Modelle durch zwei Varianten der Parameterbewertung untersucht:

- *Variante 1:* Bewertung basierend auf der Anzahl der übertragbaren Parameter
- *Variante 2:* Bewertung aufgrund eines Näherungsfehlers im Querschnitt

Variante 2 ist natürlich nur auf die Instanzen der Bauelemente anwendbar. Die Bewertung des Mappings der Materialien erfolgt in beiden Varianten parameterbasiert. Eine Bewertung des Mappings der strukturellen Komponenten findet in diesem Kontext nicht statt. Die Problematik der Bewertung des Strukturmodells wird in Kapitel 8.4 diskutiert.

8.2. Parameterbasierte Bewertung

Die Entities der IFC, von denen Instanzen gebildet werden, besitzen als Attribut häufig weitere Instanzen mit eigenen Attributen. So bildet sich eine Baumstruktur. Attribute, die in der IFC als Typen vorliegen, enthalten die eigentlichen, gesuchten Zahlenwerte. Die Instanzen werden per nach Ansys in einen entsprechenden Befehl oder eine Sequenz von Befehlen gemappt.

Betrachtet man sich beispielsweise eine Stütze mit I-Profil, wie sie in Abbildung 8.1 zu sehen ist, wird dort das Attribut `IfcIShapeProfileDef` in einen Sequenz zweier Ansys-Befehle gemappt. Dieses Attribut verkörpert ein parametrisiertes I-Profil und besitzt fünf Attribute vom Typ Dezimalzahl, die das Profil definieren. Das Attribut für den Radius

8. Qualitative Bewertung

der Ausrundung zwischen Flansch und Steg wird in Ansys aber nicht verwendet. So können nur vier von fünf Attributen übertragen werden, was einem Qualitätswert von 0.8 entspricht. Dass in Ansys dafür zwei weitere Parameter definiert werden können, spielt für das Mapping nach Ansys keine Rolle¹. Von den Qualitätswerten wird an der Verzweigung der Mittelwert gebildet, wobei zu beachten ist, dass der Qualitätswert eines möglichen Mappings der Verzweigung selbst mit in die Rechnung eingeht. An der Wurzel wird der endgültige Wert für die Übertragungsqualität dieser Instanz errechnet.

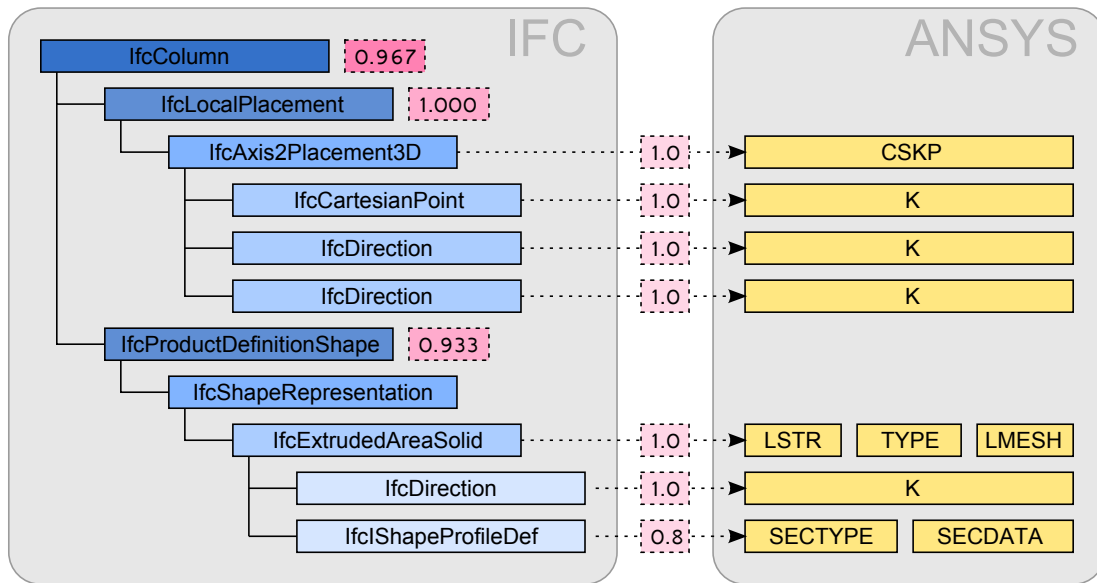


Abb. 8.1.: Parameterbasiertes bewertetes Schema-Mapping einer Stütze mit I-Profil

Eine Besonderheit tritt beim Mapping von Zugstäben auf. Das Element LINK10, das in Ansys als Elementtyp verwendet wird, kann nicht über die Querschnittsgeometrie modelliert werden, sondern akzeptiert nur den Flächeninhalt als nackten Dezimalwert. Das Problem wurde so gelöst, dass für dieses Element der Flächeninhalt in Ansys als universeller, immer passender Parameter angesehen wird. Ein Druckstab mit Kreisquerschnitt würde beim Mappen vorher und hinterher mit einem Parameter beschrieben werden, daher wäre die Qualität also trotzdem 1. Ein Rechteckquerschnitt benötigt schon zwei Parameter, so dass die Mapping-Qualität 0.5 beträgt. Das Mappen eines I-Profiles mit fünf Parametern ergäbe eine Qualität von 0.2. Das Mapping lässt sich so interpretieren, dass die Komplexität des Querschnitts beim Mappen abnimmt und dies durch einen Dezimalwert ausgedrückt wird. Abbildung 8.2 zeigt das bewertete Mapping eines rechteckigen Zugstabs.

Da aufgrund der Randbedingungen Wand und Platte nur einen Rechteckquerschnitt besitzen dürfen, ist das Mapping ohne Qualitätsverlust durchführbar.

¹Beim Mapping in die andere Richtung dagegen schon.

8. Qualitative Bewertung

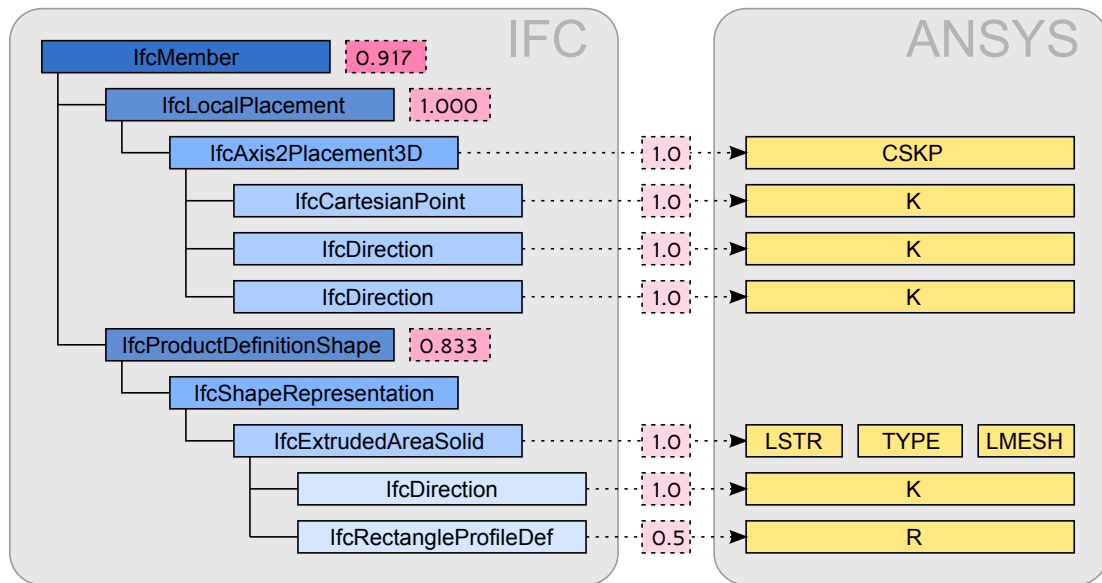


Abb. 8.2.: Parameterbasiertes bewertetes Schema-Mapping eines Zugstabes mit Rechteckprofil

Aufgrund der dezentralen Struktur der Materialverknüpfungen in den IFC gibt es für diese Bewertung nicht eine einzige Instanz, sondern drei, die zusammengerechnet werden. Auch hier geht das Mapping mit einer Qualität von 1.0 vonstatten (Abb. 8.3).

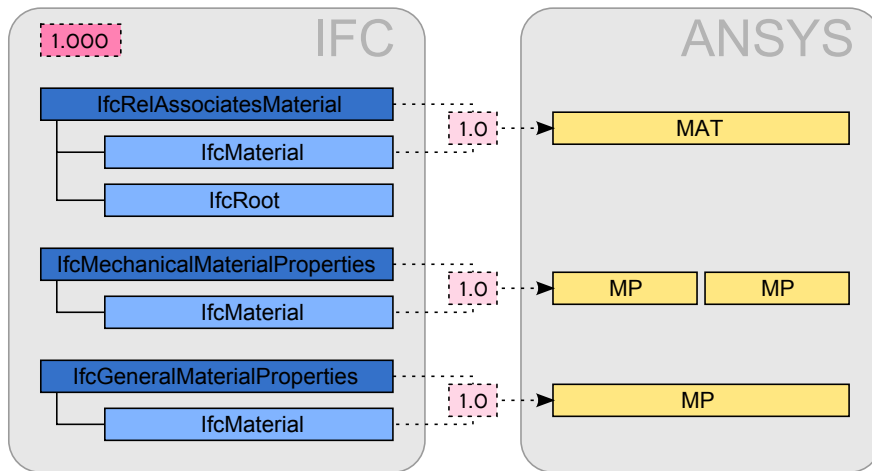


Abb. 8.3.: Bewertetes Schema-Mapping eines Materials

Wurden alle Mappings der Instanzen bewertet, müssen sie nur noch zusammengerechnet und gemittelt werden. Dabei ist zu beachten, dass bei unterschiedlicher Struktur der Entities (z.B. Querschnittsprofil von Bauelementen) unterschiedliche Werte für deren Qualität ermittelt werden.

Für die in Kapitel 5.3 vorgestellten Modelle A und B findet sich die Berechnung der Gesamtqualitäten Q_A und Q_B in den Tabellen 8.1 und 8.2.

8. Qualitative Bewertung

Bauelement	Instanz	Anzahl	Qualität
Stütze, I-Profil	IfcColumn	30	0.967
Balken, I-Profil	IfcBeam	35	0.967
Zugstab, Rundstab	IfcMember	48	1.000
Platte	IfcSlab	10	1.000
Material	IfcMaterial	2	1.000
Σ		125	122.855
Q_A			0.983

Tab. 8.1.: Berechnung der parameterbasierten Modellqualität Q_A

Bauelement	Instanz	Anzahl	Qualität
Stütze, rechteckig	IfcColumn	30	1.000
Balken, I-Profil	IfcBeam	35	0.967
Wand	IfcWallStandardCase	24	1.000
Platte	IfcSlab	10	1.000
Material	IfcMaterial	2	1.000
Σ		101	98.855
Q_B			0.979

Tab. 8.2.: Berechnung der parameterbasierten Modellqualität Q_B

8.3. Querschnittsbasierte Bewertung

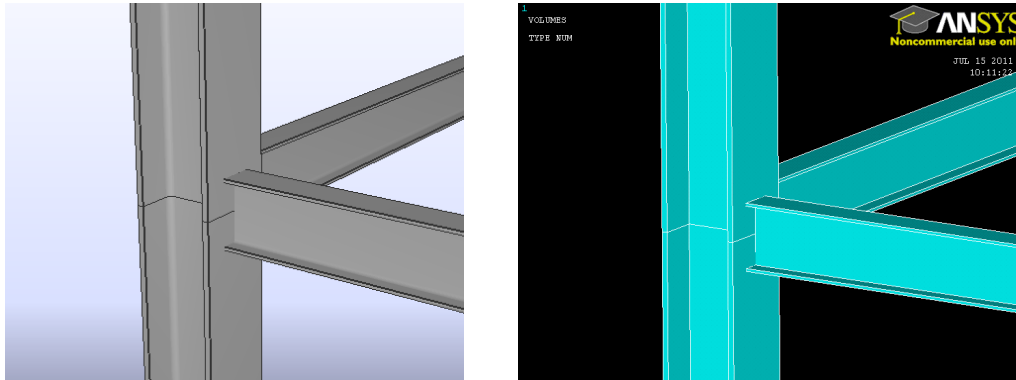


Abb. 8.4.: Vergleich der I-Profil-Modellierung zwischen IFC (links) und Ansys (rechts)

Eine andere Möglichkeit ist die Qualitätsbewertung nach dem Ansatz der Berechnung der Querschnittsungenauigkeit. Als Beispiel wird der Querschnitt eines I-Profiles herangezogen. Wie im letzten Kapitel schon festgestellt, bietet Ansys im Gegensatz zu IFC-Modellen nicht die Möglichkeit der Modellierung von Ausrundungen zwischen Flansch und Steg (Abb 8.4). Daher besitzen beide Profile eine unterschiedliche Querschnittsflä-

8. Qualitative Bewertung

che. Der Fehler lässt sich anhand des Radius r berechnen:

$$\delta A = (4 - \pi) \cdot r^2$$

Um den Fehler nicht jedes Mal ausrechnen zu müssen, kann man ihn auch kategorisieren. Abbildung 8.5 zeigt die Verteilung und den Mittelwert des Näherungsfehler über die IPB-Profilreihe. Der Mittelwert $\overline{\delta_A}$ beträgt 3.13%.

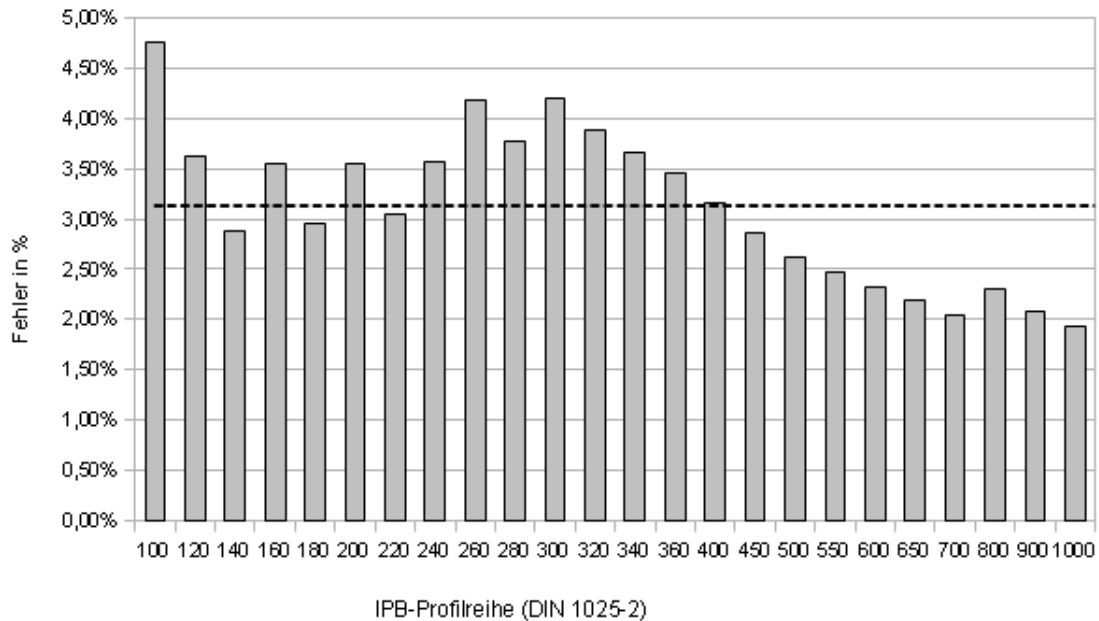


Abb. 8.5.: Näherungsfehler für IPB(HEB)-Profile in Ansys

Dies entspricht einer Qualität von 0.969 für das Mapping des Querschnittes. Setzt man diesen Wert in das bewertete Schema-Mapping für eine Stütze ein, ergibt sich die Mapping-Qualität der Stütze aus Abbildung 8.6. Anhand des Schemas lässt sich eine Gleichung zur Ermittlung der Qualität des Mappings der Stütze ausgehend von der Qualität des Querschnitts formulieren:

$$Q_{Col} = \frac{\left(\frac{Q_{Prof}+2}{3} + 1\right)}{2} = \frac{Q_{Prof} + 5}{6}$$

Da in Modell A nur drei und in Modell B nur zwei verschiedene I-Profile benutzt werden, sollen für die Berechnung der Gesamtqualität die genauen Fehlerwerte anhand dieser Formel eingehen. Die Querschnitte der anderen Bauteile lassen sich fehlerfrei mappen, so dass deren Mapping-Qualität bei 1.0 liegt. Der Druckstab wird bei dieser Form der Qualitätsberechnung ebenfalls mit einem Wert von 1.0 gemappt, da der Flächeninhalt

8. Qualitative Bewertung

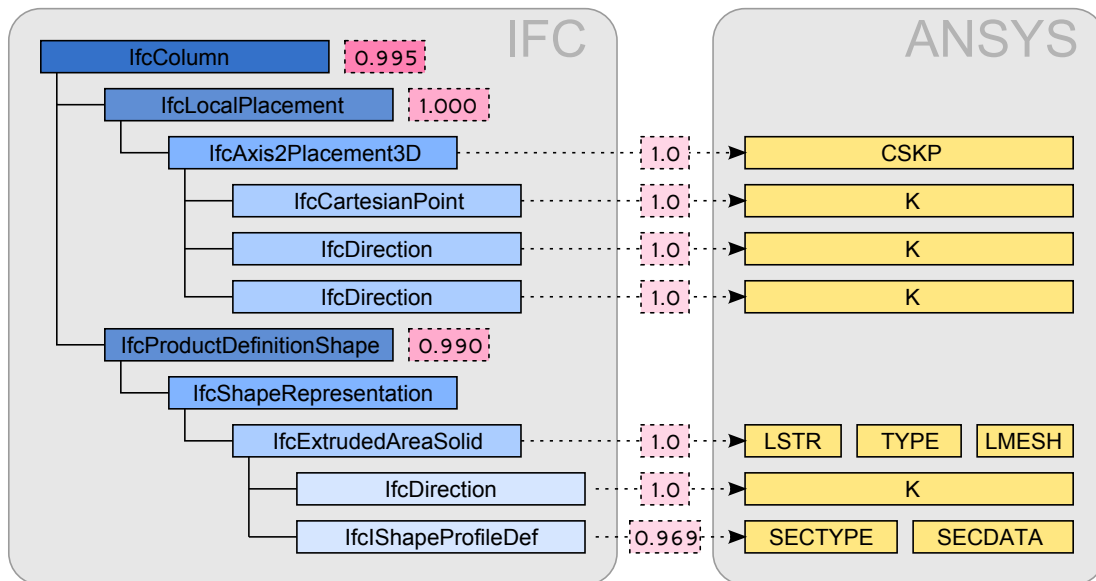


Abb. 8.6.: Querschnittsbasiertes bewertetes Schema-Mapping einer Stütze mit IPB-Profil

auf Ansys-Seite exakt dem des IFC-Bauteils entspricht.

In den Tabellen 8.3 und 8.4 werden die Gesamtqualitäten Q_A des Modells A und Q_B des Modells B ermittelt.

Bauelement	Instanz	Anzahl	δ_A	Q_{Prof}	Q_{Inst}
Stütze IPB240	IfcColumn	30	3.57%	0.964	0.994
Balken IPE200	IfcBeam	30	4.34%	0.957	0.993
Balken IPB200	IfcBeam	5	3.56%	0.964	0.994
Zugstab Ø50	IfcMember	48	0.0%	1.0	1.0
Platte	IfcSlab	10	0.0%	1.0	1.0
Material	IfcMaterial	2	0.0%	1.0	1.0
Σ		125			124.580
Q_A					0.997

Tab. 8.3.: Berechnung der querschnittsbasierten Modellqualität Q_A

8.4. Betrachtungen zur Bewertung des Strukturmodells

Die Modellierung von Auflagerknoten und Punktlasten wurde in die Bewertungen nicht mit einbezogen. Dies hat den Grund, dass das Modell grundlegend geometrisch umgesetzt wurde. Knoten und Punktlasten sind ein Bonus und gehören vom Standpunkt der IFC gesehen gar nicht zum geometrischen Modell. Würde die Struktur komplett auf Basis der Structural Analysis Domain (vgl. Kapitel 3.3.8) abgebildet werden, hätte man das Modell in einem einheitlichen Schema, das sich dann sinnvoll koppeln ließe.

8. Qualitative Bewertung

Bauelement	Instanz	Anzahl	δ_A	Q_{Prof}	Q_{Inst}
Stütze 240x240	IfcColumn	30	0.0%	1.0	1.0
Balken IPE200	IfcBeam	30	4.34%	0.957	0.993
Balken IPB200	IfcBeam	5	3.56%	0.964	0.994
Wand 10cm	IfcWallStandardCase	24	0.0%	1.0	1.0
Platte 10cm	IfcSlab	10	0.0%	1.0	1.0
Material	IfcMaterial	2	0.0%	1.0	1.0
Σ		101			100.760
Q_B					0.998

Tab. 8.4.: Berechnung der querschnittsbasierten Modellqualität Q_B

Würde man die Qualität der umgesetzten Kopplung trotzdem bewerten wollen, wäre ein Ansatz die Bewertung über die Übertragungsgenauigkeit der Koordinaten der jeweiligen Position von Knoten und Lasten. Die Bestimmung der Freiheitsgrade sowie des Kraftvektors wird in beiden Schemata analog umgesetzt und erhielte eine Qualität von 1. Die Koordinaten hingen dagegen davon ab, wie fein das generierte Netz in Ansys strukturiert ist. Der Konverter bestimmt über einen Parameter, wie fein das Netz in Ansys werden soll. Er müsste allerdings auch ermitteln, wie weit der Abstand zum nächsten Knoten wäre. Und das erforderte einen Algorithmus, der über alle vorhandenen Geometrien lief und die Abstände bestimme. Angenommen, es würde für eine Last L ein auf die Geometriedimensionen bezogener Koordinatenfehler $\delta_{K,L}$ und daraus resultierend eine Qualität $Q_{K,L}$ ermittelt werden, ergäbe sich nach Abbildung 8.7 eine Kopplungsqualität Q_L für eine Instanz einer Last mit der Formel $Q_L = 0.5 \cdot (Q_{K,L} + 1)$. Mit diesem Ansatz werden die Betrachtungen zur Bewertung des Mappings von Strukturmodellkomponenten für die untersuchte Kopplung abgeschlossen.

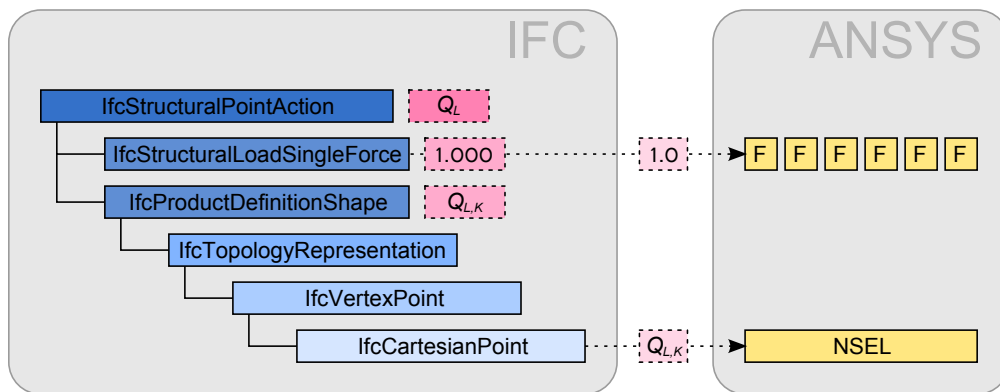


Abb. 8.7.: Bewertetes Mapping einer Punktlast

9. Zusammenfassung und Ausblick

9.1. Zusammenfassung

Kopplungen von Anwendungen - gerade im Bereich des Ingenieurwesens - sollen einen möglichst fehlerfreien Austausch von Informationen realisieren. Um dies zu gewährleisten, gibt es verschiedene Bewertungsverfahren. Apostiori-Verfahren vergleichen die Modelle des Ausgangs- und des Zielschemas. Apriori-Verfahren untersuchen die Kopplung direkt. Daher sind sie vorteilhafter, da sie anstatt der Modelle die Schemata selbst betrachten. Das bewertete Schema-Mapping ist ein Apriori-Verfahren, das für die Untersuchungen in dieser Arbeit eingesetzt wurde.

Die Arbeit hat sich damit beschäftigt, zwei Modelle, die im IFC-Schema vorliegen, per Kopplung in jeweils ein Ansys-Modell zu konvertieren. Dabei wurden die Beziehungen der Datenstrukturen im Rahmen einer Schema-Analyse untersucht und reduziert. Durch ein Schema-Mapping konnten die IFC-Objekte in ADPL-Code zur Erzeugung von Ansys-Objekten umgeformt werden. Dieses Schema-Mapping wurde durch zwei Varianten bewertet. Einmal erfolgte die Qualitätsbewertung allein auf Basis der Parametermengen, das andere Verfahren bewertete den Querschnitt von Geometrien. Als Ergebnis der Bewertungen ergab sich für jedes Modell ein Wert für eine Gesamtqualität. Die Werte liegen für beide Modelle knapp bei 1, was nur einen minimalen Qualitätsverlust bedeutet. Einen Vergleich durch Sichtprüfung der Modelle in den jeweiligen Anwendungen (vgl. Abbildungen im Kapitel 7.5) lässt den gleichen Schluss zu.

Für die Untersuchungen wurden die Werkzeuge mit Java selbst programmiert. Es wurden Klassen zur Erschaffung von IFC-Modellen programmiert, weiterhin ein Analyzer für die Untersuchung des IFC-Schemas zur Unterstützung der Schema-Analyse. Der Konverter wurde programmiert, um IFC-Modelle in Ansys-ausführbaren APDL-Code zu konvertieren.

9.2. Ausblick

Der Umfang der IFC-Modelle umfasst nur einen Teilbereich des Spektrums der kompletten IFC. Die Bauteile lassen sich in Bezug auf Typenvielfalt wie auch in den Möglichkeiten der geometrischen Modellierung deutlich erweitern. Die Schema-Analyse könnte

9. Zusammenfassung und Ausblick

nicht nur die Verknüpfungen über Attribute von Entities, sondern auch die Verknüpfungen durch IFC-Beziehungsklassen (z.B. Verknüpfung eines Bauteils mit einem Material) erfassen.

Es wäre sicherlich eine spannende Aufgabe, ein Modell komplett in der Structural-Analysis-Domäne der IFC abzubilden und darauf ein Schema-Mapping zu Ansys zu konstruieren. Dafür wäre allerdings mehr Information seitens der Entwickler des IFC-Standards zu dieser Domäne hilfreich.

Würde ein strukturelles Modell existieren, könnte man das Mapping auch auf struktureller Ebene bewerten. Man könnte genauer auf die Eingangswerte, zum Beispiel die Ermittlung von Trägheitsmomenten, seitens der statischen Berechnungen eingehen und deren Mapping unter diesem Aspekt bewerten.

Eine Einführung von Wichten könnte die Aussagekraft der Qualitätsbewertung steigern. Qualitäten von Objekten eines jeden Typs könnten mit einem Faktor versehen werden, der die entsprechende Wichte des Objekttypes ausdrückt.

Nicht zuletzt wäre es interessant zu untersuchen, ob und wie ein Mapping samt Bewertung in die Gegenrichtung, also von Ansys zu den IFC, zu realisieren ist. Voraussetzung wäre in diesem Fall der APDL-Quellcode, der zu einem Modell der IFC konvertiert werden müsste.

Literaturverzeichnis

- [ANS09] ANSYS, Inc.: *Release 12.1 Documentation for ANSYS*, 2009
- [Bui11] *buildingSMART*. Website, 2011. – <http://www.buildingsmart.de/index.htm> (Abgerufen am 20.09.2011)
- [DOO07] DIVINGRI, Kaan ; OZCAN, Can ; OZEN, Metin: Coupling the AnyBody and ANSYS Software Suites for Biomedical Applications / Ozen Engineering, Inc. 2007. – Forschungsbericht
- [FFK11] FRÖBEL, Toni ; FIRMENICH, Berthold ; KOCH, Christian: Quality assessment of coupled civil engineering applications. In: *Advanced Engineering Informatics* 25 (2011)
- [Fli03] FLIEGNER, Christian: *Umsetzung der IFC mit Java und XML am Beispiel der Tragwerksplanung*, Bauhaus-Universität Weimar, Diplomarbeit, 2003
- [Gei01] GEIGER, Andreas: Produktdatenmodelle im Bauwesen - IFC im Praxistest / Forschungszentrum Karlsruhe, Institut für Angewandte Informatik. 2001. – Forschungsbericht
- [Gro10] GROLLA, Sebastian: *Synchron bidirektionale Kopplung verteilter Anwendungen im Bauingenieurwesen am Beispiel CAD - FEM*, Bauhaus-Universität Weimar, Studienarbeit, 2010
- [Lie09] LIEBICH, Thomas ; BUILDINGSMART INTERNATIONAL MODELING SUPPORT GROUP (Hrsg.): *IFC 2x3 Model Implementation Guide*. buildingSMART International Modeling Support Group, 2009. – Version 2.0
- [MG07] MÜLLER, Günther ; GROTH, Clemens: *FEM für Praktiker - Band 1: Grundlagen*. 8., neu bearbeitete Auflage. expert verlag, 2007
- [MHCA06] MA, Homan ; HA, Kwan Mei E. ; CHUNG, Chun Kit J. ; AMOR, Robert: Testing Semantic Interoperability. In: *Proceedings of Joint International Conference on Computing and Decision Making in Civil and Building Engineering (ICCCBE)* (2006), S. 1216–1225

Literaturverzeichnis

- [MSG07] MODEL SUPPORT GROUP, IAI: *IFC2x3 Final Documentation*. Website, 2007. – <http://buildingsmart-tech.org/ifc/IFC2x3/TC1/html/index.htm> (Abgerufen am 27.07.2011)
- [OR05] OSTERRIEDER, Prof. Dr.-Ing. P. ; RICHTER, Dipl.-Ing. S. ; LEHRSTUHL STATIK UND DYNAMIK BRANDENBURGISCHE TECHNISCHE UNIVERSITÄT COTTBUS (Hrsg.): *IAI-Projekt ST-5 Structural Timber Model Teil III Implementierungshilfe*. Lehrstuhl Statik und Dynamik Brandenburgische Technische Universität Cottbus, 2005. – Version: 1.ß2
- [Reu11] REUTER, Markus C.: *Multicriterial Evaluation Method for the Prognosis Quality of Complex Engineering Models*, Bauhaus-Universität Weimar, Diss., 2011
- [SF211] *FEM kurz und bündig*. Website, 2011. – <http://www.smart-fem.de/fem.html> (Abgerufen am 03.10.2011)
- [SNY07] STOLARSKI, Tadeusz ; NAKASONE, Y. ; YOSHIMOTO, S.: *Engineering Analysis with ANSYS Software*. Butterworth Heinemann, 2007
- [Sto07] STOIBER, Franz: *Analyse der Verwendbarkeit von IFC 2x3 als produktneutrales Austauschformat der österreichischen Hochbaurichtlinien*, TU Wien, Diplomarbeit, 2007
- [Wix00a] WIX, Jeffrey: *Data Modelling Using EXPRESS-G for IFC Development*. 2000. – Forschungsbericht
- [Wix00b] WIX, Jeffrey: *The EXPRESS Definition Language for IFC Development*. 2000. – Forschungsbericht
- [WKLS03] WEISE, Matthias ; KATRANUSCHKOV, Peter ; LIEBICH, Thomas ; SCHERER, Raimar J.: *Structural analysis extension of the IFC modelling framework / Institute of Applied Computer Science in Civil Engineering, University of Technology Dresden*. 2003. – Forschungsbericht

Abbildungsverzeichnis

1.1. Dateibasierter Datenaustausch zwischen Applikationen	3
1.2. Schema-Matching	5
1.3. Beispielszenario mit Kopplungen	6
1.4. Schemata c' und s	6
1.5. Mapping Patterns beim Schema-Mapping	7
1.6. Schema-Mapping von c' nach s	7
1.7. Übertragung der Stützen von Modeller A zu Modeller B	8
1.8. Matrizen der Schema-Mappings $\tilde{m}_{C',S}$ und $\tilde{m}_{S,C'}$	10
1.9. Mapping mit mehreren Instanzen	11
2.1. Referenzobjekt	13
3.1. Layerstruktur der IFC	22
3.2. Struktur des IfcGeometricRepresentationContext	23
3.3. Exemplarische Unterteilung räumlicher Strukturen	24
3.4. Struktur von IfcObjectPlacement	25
3.5. Definition von geometrischen Repräsentationen	26
3.6. Varianten von IfcRepresentationItem	27
3.7. Vererbungsstruktur von IfcBuildingElement	28
3.8. Eingliederung von Bauelementen in räumliche Strukturen	29
3.9. Extrusionsdarstellung einer Stütze mit einem I-Profil	30
3.10. Darstellung einer Standardwand in den zwei erforderlichen Kontexten . .	31
3.11. Extrusionsdarstellung einer Platte	32
3.12. Relationen mit IfcStructuralAnalysisModel	33
3.13. Aufbau des Linienelementes IfcStructuralCurveMember	34
3.14. Funktionsweise der Materialzuweisung	35
3.15. Materialverknüpfung eines Strukturelements	36
4.1. Ablauf einer Finite-Elemente-Analyse	37
4.2. Verknüpfung zweier Modellierungskonzepte in Ansys	39
4.3. Element SHELL181 in Ansys	40

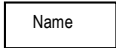
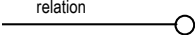
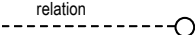
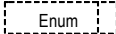
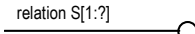
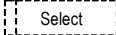
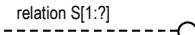
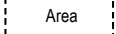
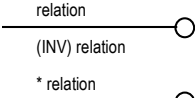
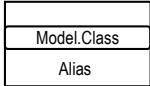
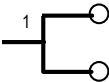
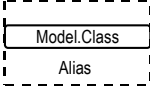
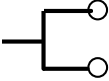
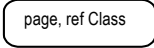
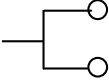
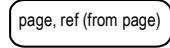
Abbildungsverzeichnis

5.1. Darstellung des Referenzobjektes mit I-Profil-Stützen und Zugstäben im IFC-Viewer	43
5.2. Darstellung des Referenzobjektes mit Rechteckstützen und Wänden im IFC-Viewer	43
6.1. Reduktionsschritte der Schema-Analyse von IfcColumn	50
6.2. Schema von IfcColumn mit einem Rechteckprofil	51
6.3. Reduktionsschritte der Schema-Analyse von IfcRelAssociatesMaterial . .	54
6.4. Objektdiagramm für Materialrelationen mit den möglichen Bauelementen	55
6.5. Schema von IfcStructuralPointConnection	56
6.6. Schema von IfcStructuralPointAction	57
7.1. Mapping von IfcColumn mit I-Profil	61
7.2. Mapping von IfcMember mit Kreisprofil	61
7.3. Mapping von IfcWallStandardCase	62
7.4. Mapping von IfcSlab	63
7.5. Mapping von Material	63
7.6. Mapping eines Auflagerknotens	64
7.7. Mapping einer Punktlast	64
7.8. Ausgangsmodelle im IFC-Schema	65
7.9. Modelle nach Mapping in Ansys	65
7.10. Modelle in Verformungsdarstellung in Ansys	66
8.1. Parameterbasiertes bewertetes Schema-Mapping einer Stütze mit I-Profil .	68
8.2. Parameterbasiertes bewertetes Schema-Mapping eines Zugstabes mit Rechteckprofil	69
8.3. Bewertetes Schema-Mapping eines Materials	69
8.4. Vergleich der I-Profil-Modellierung zwischen IFC und Ansys	70
8.5. Näherungsfehler für IPB(HEB)-Profile in Ansys	71
8.6. Querschnittsbasiertes bewertetes Schema-Mapping einer Stütze mit IPB-Profil	72
8.7. Bewertetes Mapping einer Punktlast	73

Anhang

A. Kurzübersicht über EXPRESS-G

List of Symbols

	- class		- Mandatory relation (exactly 1)
	- simple data type		- Optional relation zero or one
	- enumeration data type		- Set relation (one or many)
	- select data type		- Set relation (zero, one or many)
	- defined data type		- INVERSE relation
	- USE FROM interface		- Exclusive supertype/subtype relation
	- REFERENCE FROM interface		- Inclusive supertype/subtype relation
	- ONTO ANOTHER PAGE connector		- Select relation
	- ONTO THIS PAGE		

Quelle: [Wix00a]

B. Schema des EXPRESS-Parsers in Backus-Naur-Form

```
DOCUMENT START
TOKENS
<DEFAULT> SKIP : {
"\r"
| "\t"
| "\n"
| " "
}

<DEFAULT> TOKEN : {
<ABSTRACT: "ABSTRACT">
| <AGGR_ARR: "ARRAY">
| <AGGR_BAG: "BAG">
| <AGGR_LST: "LIST">
| <AGGR_SET: "SET">
| <DERIVE: "DERIVE">
| <END_ENTITY: "END_ENTITY">
| <END_FUNCTION: "END_FUNCTION">
| <END_RULE: "END_RULE">
| <END_SCHEMA: "END_SCHEMA">
| <END_TYPE: "END_TYPE">
| <ENTITY: "ENTITY">
| <ENUMERATION: "ENUMERATION">
| <FIXED: "FIXED">
| <FOR: "FOR">
| <FUNCTION: "FUNCTION">
| <GENERIC: "GENERIC" | "Generic">
| <INVERSE: "INVERSE">
| <OF: "OF" | "Of">
| <ONEOF: "ONEOF">
| <OPTIONAL: "OPTIONAL">
| <RULE: "RULE">
| <SCHEMA: "SCHEMA">
| <SELECT: "SELECT">
| <SELF: "SELF">
| <SUBTYPE: "SUBTYPE">
| <SUPERTYPE: "SUPERTYPE">
| <TYPE: "TYPE">
| <UNIQUE: "UNIQUE">
| <UNKNOWN: "?">
| <WHERE: "WHERE">
}
```

B. Schema des EXPRESS-Parsers in Backus-Naur-Form

```

<DEFAULT> TOKEN : {
<EOC: (";")+>
| <OP_DEF: ":">
| <OP_SET: "=">
| <OP_COM: ",">
| <OP_BRK: "(">
| <OP_BRK_END: ")">
| <OP_ARR: "[">
| <OP_ARR_END: "]">
| <OP_BLC: "{">
| <OP_BLC_END: "}">
| <OP_DOT: ".">
| <OP_BCK: "\\>
| <OP_CMT: "("*>
| <OP_CMT_END: "*)">
}

<DEFAULT> TOKEN : {
<NUMBER_LIT: ["0"-"9"] (["0"-"9"])*>
| <ID_LIT: ["_", "A"-"Z", "a"-"z"] (["_", "A"-"Z", "a"-"z", "0"-"9"])*>
| <STRING_LIT: (["A"-"Z", "a"-"z", "0"-"9", "$", "_", "*", ".", "-", "|", "<_", ">", "\u00a7",
"#", "^", "\"", "+", "\\", "/", "?", "!", "%", "&", "@", "\u00b4", "\u00b4", "\u00b4", "\u00e4",
"\u00f6", "\u00fc", "\u00c4", "\u00d4", "\u00dc", "\u00df"])+>
}

NON-TERMINALS

    start := ( section_intro )? section_header ( section_type | section_entity |
        section_function | section_rule | <EOF> )* <END_SCHEMA>
    section_intro := <OP_CMT> get_the_rest <OP_CMT_END>
    section_header := <SCHEMA> <ID_LIT> <EOC>

//--- TYPES ---
    section_type := <TYPE> <ID_LIT> <OP_SET> data_type <EOC>
        ( <WHERE> ( domain_rule <EOC> )+ )? <END_TYPE> <EOC>

//--- ENTITIES ---
    section_entity := <ENTITY> <ID_LIT> ( ent_supertype )? ( ent_subtype )? <EOC>
        ( ent_attrib <EOC> )* ( <UNIQUE> ( ent_unique <EOC> )+ )?
        ( <DERIVE> ( ent_derive <EOC> )+ )? ( <INVERSE>
        ( ent_inverse <EOC> )+ )? ( <WHERE> ( domain_rule <EOC> )+ )?
        <END_ENTITY> <EOC>
    ent_supertype := ( <ABSTRACT> )? <SUPERTYPE> <OF> <OP_BRK> <ONEOF> <OP_BRK> <ID_LIT>
        ( <OP_COM> <ID_LIT> )* <OP_BRK_END> <OP_BRK_END>
    ent_subtype := <SUBTYPE> <OF> <OP_BRK> <ID_LIT> <OP_BRK_END>
    ent_attrib := <ID_LIT> <OP_DEF> ( <OPTIONAL> )? data_type
    ent_unique := <ID_LIT> <OP_DEF> <ID_LIT> ( <OP_COM> <ID_LIT> )*
    ent_inverse := <ID_LIT> <OP_DEF> data_type <FOR> <ID_LIT>
    ent_derive := ( <ID_LIT> | <STRING_LIT> ) <OP_DEF> data_type
        <OP_DEF> <OP_SET> get_the_rest

//--- FUNCTIONS ---
    section_function := <FUNCTION> <ID_LIT> <OP_BRK> func_arglist <OP_BRK_END> <OP_DEF>

```

B. Schema des EXPRESS-Parsers in Backus-Naur-Form

```
data_type <EOC> ( get_the_rest <EOC> )* <END_FUNCTION> <EOC>
func_arglist := func_argset ( <EOC> func_argset )*
func_argset := <ID_LIT> ( <OP_COM> <ID_LIT> )* <OP_DEF> data_type

//--- RULES ---
section_rule := <RULE> <ID_LIT> <FOR> <OP_BRK> <ID_LIT> <OP_BRK_END> <EOC>
               ( get_the_rest <EOC> )* ( <WHERE> ( domain_rule <EOC> )+ )?
               <END_RULE> <EOC>

//--- SUBFUNS ---
data_type := ( <UNIQUE> )? ( data_simple | data_generic | data_enum_sel | data_aggr )
data_simple := <ID_LIT> ( ( <OP_BRK> <NUMBER_LIT> <OP_BRK_END> )? ( <FIXED> )? )
data_generic := <GENERIC> ( <OP_DEF> <ID_LIT> )?
data_enum_sel := ( ( <ENUMERATION> <OF> ) | ( <SELECT> ) ) <OP_BRK> <ID_LIT>
                ( <OP_COM> <ID_LIT> )* <OP_BRK_END>
data_aggr := ( <AGGR_ARR> | <AGGR_BAG> | <AGGR_LST> | <AGGR_SET> )
              ( <OP_ARR> ( <NUMBER_LIT> | <UNKNOWN> | <ID_LIT> | <STRING_LIT> )
                <OP_DEF> ( <NUMBER_LIT> | <UNKNOWN> | <ID_LIT> | <STRING_LIT> )
                <OP_ARR_END> )? <OF> data_type
domain_rule := <ID_LIT> <OP_DEF> get_the_rest
get_the_rest := ( ( <NUMBER_LIT> | <ID_LIT> | <STRING_LIT> | <UNKNOWN> | <OP_SET> |
                  <OP_DEF> | <OP_COM> | <OP_DOT> | <OP_BCK> | <OP_BRK> | <OP_BRK_END> |
                  <OP_ARR> | <OP_ARR_END> | <OP_BLC> | <OP_BLC_END> | <AGGR_ARR> |
                  <AGGR_BAG> | <AGGR_LST> | <AGGR_SET> | <OF> | <GENERIC> | <FOR> |
                  <ONEOF> | <SELF> ) )+

DOCUMENT END
```

Selbstständigkeitserklärung

Ich erkläre, dass ich die vorliegende Arbeit selbstständig und nur unter Verwendung der angegebenen Hilfsmittel angefertigt habe.

Weimar, Oktober 2011