TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Bauingenieurwesen Institut für Bauinformatik

Diplomarbeit

Das Multimodell unter Ausnutzung des Variationsprinzips am Beispiel der Brandsimulation mit dem Fire Dynamics Simulator

Dominik Holzem

Geboren am: 31. Januar 1985 in Bonn
Matrikelnummer: 3935401
Immatrikulationsjahr: 2013

zur Erlangung des akademischen Grades

Diplom-Ingenieur (Dipl.-Ing.)

Betreuer

Dipl.-Ing. Ngoc Trung Luu

Betreuender Hochschullehrer

Prof. Dr.-Ing. Raimar Scherer

Eingereicht am: 24. August 2018

TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Bauingenieurwesen Institut für Bauinformatik

Diplomarbeit

Das Multimodell unter Ausnutzung des Variationsprinzips am Beispiel der Brandsimulation mit dem Fire Dynamics Simulator

Dominik Holzem

Urschrift der „Aufgabenstellung zur Diplomarbeit“

TECHNISCHE UNIVERSITÄT DRESDEN

Fakultät Bauingenieurwesen Institut für Bauinformatik

Aufgabenstellung für die Diplomarbeit

Name	cand.-ing. Dominik Holzem
Vertiefung	Konstruktiver Ingenieurbau (KI)

Thema: **Das Multimodell unter Ausnutzung des Variationsprinzips am Beispiel der Brandsimulation mit dem Fire Dynamics Simulator**
(The multimodel under exploitation of the variation principle in example of the fire simulation with the Fire Dynamics simulator)

Zielsetzung:

Brandsimulationen im Vorbeugenden Brandschutz gewinnen immer öfter Anwendung in der Planung von Gebäuden. International weit verbreitet ist dabei der *Fire Dynamics Simulator (FDS)*, ein freies Programm zur Simulation eines Brandszenarios. Die Auswertungen solch einer Simulation führen zu neuen und essenziellen Informationen in der Bauwerksstruktur. Dieser gewonnene Informationsraum veranlasst die Betrachtung des beim *Building Information Modelling (BIM)* verwendeten Multimodell-Ansatzes, sodass insgesamt eine neue und einheitliche Informationsressource entsteht.

Es soll ein Multimodell erstellt werden, dessen Fachmodelle die Simulation von Brandszenarien mit dem *Fire Dynamics Simulator* in einem Gebäude ermöglichen. Die Bauwerksstruktur soll dabei durch die *Industry Foundation Classes (IFC)* in einem Gebäudemodell abgebildet werden. Alle für die Simulation notwendigen Parameter sind durch Datenobjekte in Fachmodellen bereitzustellen. Es sind Bedingungen festzulegen, welche die einzelnen Datenobjekte erfüllen müssen bzw. auf welche Art und Weise sich die Objekte darstellen sollen (= Datenannotation). Die Brandszenarien sollen sich an einem übergeordneten Grundmodell orientieren und durch Variation der dort vorgeschriebenen Parameter in untergeordneten Tochtermodellen darstellen (= Variationsprinzip).

Mit den objektorientierten Daten ist eine kompilierbare Eingabedatei für den *Fire Dynamics Simulator* zu generieren. Der Einsatz einer Anwendungsprogrammierschnittstelle (z.B. JSDAI) wird zum Lesen und Schreiben dieser Daten empfohlen, jedoch nicht zwingend vorausgesetzt. Es soll eine Transformation in das Quelltextformat des *Fire Dynamics Simulator* realisiert werden, wobei ein generisches Verhalten der Transformationsroutine anzustreben ist.

Das Kompilieren dieser Eingabedatei soll die Simulation mit dem *Fire Dynamics Simulator* ermöglichen. In einer theoretischen Betrachtung sind ferner die Grundlagen dieser Simulationsmethode zu verdeutlichen. Mittels einer Modellanalyse sind Zielkonflikte der Modellierung herauszuarbeiten. Simulationsergebnisse sollen hinsichtlich ihrer Auswertmöglichkeit eingeschätzt und durch geeignete Filtermethoden in weiteren Datenobjekten dem Gebäudemodell bereitgestellt werden. Es wird grundsätzlich empfohlen, die gewonnenen Simulationsergebnisse einem Benchmark zu unterziehen (= Validierung der Ansätze).

Es soll insgesamt untersucht werden, ob der Multimodell-Ansatz und das Variationsprinzip zukünftige Anwendungen im Bereich der Brandsimulationen ermöglichen können. Exemplarisch ist dies am Beispiel des *Fire Dynamics Simulator* zu hinterfragen. Ferner ist zu ergründen, inwieweit die *Industry Foundation Classes* als Standard für derartige Anwendungen im Bauwesen geeignet ist.

Die besonderen Hinweise des Instituts für die Anfertigung der Diplomarbeit sind zu beachten.

Wiss. Betreuer TU Dresden: Dipl.-Ing. Ngoc Trung Luu

ausgehändigt am:
einzureichen am:

Prof. Dr.-Ing. Raimar Scherer
Verantwortlicher Hochschullehrer

In dir muß brennen, was du in anderen entzünden willst.

Augustinus Aurelius (354 - 430)

Danke.

Inhaltsverzeichnis

Abbildungsverzeichnis **8**

Tabellenverzeichnis **10**

Quelltextverzeichnis **12**

Abkürzungsverzeichnis **13**

1. Einleitung **14**
1.1. Motivation 15
1.2. Ziele und Abgrenzung der Arbeit 16
1.3. Aufbau der Arbeit 17

2. Systematik und theoretische Grundlagen **18**
2.1. Methodische Vorgehensweise 19
2.2. Brandsimulationen mit dem Fire Dynamics Simulator 23
2.2.1. Einführung in die CFD-Methode 23
2.2.2. Räumliche Anordnung von Geometrien 26
2.2.3. Erste Schritte mit dem Simulationsprogramm 27
2.3. Das Multimodell mit dem Variationsprinzip 28
2.3.1. Begriffserklärung Multimodell 28
2.3.2. Begriffserklärung Variationsmodell 31

3. FDS-Minimalbeispiel **32**
3.1. Einführung in das Beispiel 33
3.2. Simulationsprofil 33
3.3. Bauwerksmodellierung 35
3.4. Modellierung einer Brandentstehung 36
3.5. Modellierung brennbarer Gegenstände 39
3.6. Auswertung und Simulation 40
3.7. Zusammenfassung der Erkenntnisse 41

4. Informationsraum der Industry Foundation Classes **42**
4.1. Gegenstand der Untersuchung 43
4.2. Analyse der IFC-Austauschdatei 44
4.2.1. Darstellung der Räume durch die IfcSpace 47
4.2.2. Höhenversetzte L-förmige Geometrie am Beispiel der IfcSlab 49

4.2.3. Öffnungen in Wänden am Beispiel der IfcWindow 51
4.2.4. Achsausrichtungen der lokalen Koordinatensysteme 53
4.2.5. Informationen aus der IfcPropertySet 55
4.3. Zusammenfassung der Erkenntnisse 56

5. Das Multimodell mit dem Variationsprinzip 57
5.1. Bildung der Fachmodelle 58
5.1.1. Brandschutztechnische Infrastruktur 59
5.1.2. Informationen zu der Entstehung von Bränden 61
5.1.3. Bereitstellung der Materialdaten 66
5.2. Bündelung der Informationsräume 69
5.2.1. Datenstruktur zur Verlinkung der Fachmodelle 69
5.2.2. Neutrales Dateiformat für ein Linkmodell 71
5.2.3. Informationsressource für den Fire Dynamics Simulator 72
5.3. Effizienz von Brandsimulationen 76
5.3.1. XML Schema Definition zum Variationsmodell 77
5.3.2. Erkennen der Variationsmöglichkeiten 79
5.3.3. Simulationsmodell für den Fire Dynamics Simulator 81
5.4. Zusammenfassung der Erkenntnisse 83

6. Programmentwicklung 85
6.1. Vorbemerkung 86
6.2. Beschreibung der Programmabläufe 86
6.3. Verifikation der vorgestellten Ansätze 89

7. Schlussbetrachtung 92
7.1. Zusammenfassung 93
7.2. Ergebnisse der Arbeit 94
7.3. Ausblick 95

Literatur- und Quellenverzeichnis 96
Literatur 96
Normen, Regelwerke und Richtlinien 97
Internet 97
Software 98

Erklärung zur selbständigen Abfassung der Diplomarbeit 100

A. Anhang 100
A.1. FDS-Minimalbeispiel 101
A.2. Informationsraum der Industry Foundation Classes 108
A.3. Das Multimodell mit dem Variationsprinzip 111

B. Compact Disk 120

Abbildungsverzeichnis

2. Systematik und theoretische Grundlagen 18
2.1. Methodische Vorgehensweise . . . 22
2.2. Beispiel für einen kartesischen Rechenraum . . . 24
2.3. Räumliche Anordnung der Geometrien . . . 26
2.4. Ablauf der FDS-Simulation für die Beispiel-Eingabedatei Couch . . . 27
2.5. Schematischer Aufbau von Multimodellen . . . 28
2.6. Das generische Variationsmodell in einem UML-Klassendiagramm . . . 31

3. FDS-Minimalbeispiel 32
3.1. Ausgewähltes Simulationsprofil . . . 33
3.2. Generierung der Bauwerksgeometrie . . . 35
3.3. Modellierung einer Brandentstehung . . . 36
3.4. Zeitlicher Verlauf der Wärmefreisetzungsrate . . . 38
3.5. Darstellung brennbarer Gegenstände . . . 39
3.6. Auswertung und Simulation für das FDS-Minimalbeispiel . . . 41

4. Informationsraum der Industry Foundation Classes 42
4.1. Auswahl an Gebäudeelementen zur Untersuchung der IFC . . . 43
4.2. Transformation der Informationsdaten von der IfcSlab in den FDS . . . 46
4.3. Vererbung der Entitäten der IfcBuildingElement . . . 47
4.4. Geometrische Angaben aus der IfcSpace . . . 48
4.5. Transformation der Informationsdaten von der IfcSpace in den FDS . . . 49
4.6. Geometrische Angaben aus der IfcSpace . . . 50
4.7. Geometrische Angaben zur Beschreibung einer Wandöffnung . . . 52
4.8. Beispiele für die Achsausrichtungen der lokalen Koordinatensysteme . . . 53
4.9. Grafische Darstellung der Rotationsvarianten in der xy-Ebene . . . 54

5. Das Multimodell mit dem Variationsprinzip 57
5.1. EXPRESS-Datenschema für die Vergabe eindeutiger Identifikatoren . . . 58
5.2. EXPRESS-Datenschema für eine brandschutztechnische Infrastruktur . . . 60
5.3. Schematisierter Brandverlauf für einen natürlichen Brand . . . 61
5.4. EXPRESS-Datenschema für die Modellierung einer Brandentstehung . . . 64
5.5. EXPRESS-Datenschema für die Bereitstellung der Materialdaten . . . 68
5.6. EXPRESS-Datenschema für die Umsetzung der Linkmodelle . . . 70
5.7. Grafische Darstellung der XML Schema Definition LinkModel-2.0.0.xsd . . . 71

5.8. Bewertung und Quantifizierung eines Relatums im LinkModel-2.0.0.xsd . . 72
5.9. Schematischer Aufbau von Grund- und Untermodellen 76
5.10. XML Schema Definition zum Variationsmodell 77
5.11. EXPRESS-Datenschema für das FDS-Simulationsmodell 82
5.12. Das Multimodell für Simulationen mit dem Fire Dynamics Simulator 83
5.13. Das Multimodell unter Ausnutzung des Variationsprinzips 84

6. Programmentwicklung **85**
6.1. Auswertung eines Linkmodells mit dem LinKCubE 87
6.2. Auswertung der Fachmodellinformationen mit dem LinKBuffeR 88
6.3. Datenübertrag aus dem Linkmodell in den LinKCubE 89
6.4. Ergebnis der FDS-Eingabedatei im Vergleich zu der Bauwerksmodellierung 89
6.5. Ergebnis der FDS-Eingabedatei 90
6.6. Geometrische Überschneidungen im FDS-Simulationsmodell 91

A. Anhang **101**
A.1. Auswertung und Simulation nach T = 4 Sekunden 104
A.2. Auswertung und Simulation nach T = 10 Sekunden 104
A.3. Auswertung und Simulation nach T = 30 Sekunden 105
A.4. Auswertung und Simulation nach T = 45 Sekunden 105
A.5. Brandentstehung nach T = 7 Sekunden 106
A.6. Brandentstehung nach T = 10 Sekunden 106
A.7. Brandentstehung nach T = 15 Sekunden 107
A.8. Brandentstehung nach T = 20 Sekunden 107

Tabellenverzeichnis

2. Systematik und theoretische Grundlagen **18**
2.1. Übersicht Software 22

3. FDS-Minimalbeispiel **32**
3.1. Abmessungen der Bauwerksgeometrie 35
3.2. Auflistung gewählter Materialparameter 40

4. Informationsraum der Industry Foundation Classes **42**
4.1. Brandschutztechnische Infrastruktur aus der IfcPropertySet 55

5. Das Multimodell mit dem Variationsprinzip **57**
5.1. Informationen für eine brandschutztechnische Infrastruktur 59
5.2. Übersicht der Linkkategorien 73
5.3. Übersicht der Variationsparameter in den Fachmodellen 79

Quelltextverzeichnis

3. FDS-Minimalbeispiel **32**
3.1. Eingabebefehle für das Simulationsprofil 34
3.2. Eingabebefehle für die Bauwerksgeometrie 36
3.3. Eingabebefehle für die Modellierung einer Brandentstehung 37
3.4. Eingabebefehle hinsichtlich geometrischer Bedingungen 37
3.5. Eingabebefehle für die Modellierung brennbarer Gegenstände 40
3.6. Eingabebefehle für die Temperaturauswertung 41

4. Informationsraum der Industry Foundation Classes **42**
4.1. Darstellung eines Gebäudeelements durch die IFC 44
4.2. Örtliche Lage und geometrische Abmessungen der IfcSlab 45
4.3. Generierung der FDS-Eingabebefehle aus der IfcSlab 45
4.4. Generierung der FDS-Eingabebefehle aus der IfcSpace 48
4.5. Geometrische Angaben aus der IfcSlab 50
4.6. Angaben zu der Höhenlage in der IfcSlab. 51
4.7. Generierung einer &HOLE-Anweisung aus der IfcWindow 52
4.8. Beispiel für eine Achsausrichtung im lokalen Koordinatensystem 54
4.9. Beispiel für eine Bauteilklassifizierung durch die IfcPropertySet 55

5. Das Multimodell mit dem Variationsprinzip **57**
5.1. Darstellung der Bauteilanforderungen im FDS durch RGB-Anweisungen . 60
5.2. Informationsraum für die Modellierung einer Brandentstehung 65
5.3. Typische Materialparameter im FDS in einer MATL-Anweisung 66
5.4. Erweiterte Materialparameter im FDS in einer MATL-Anweisung 67
5.5. Materialparameter im FDS für komplexe Pyrolyse-Modelle 67
5.6. Informationen für die Bereistellung der Materialdaten 69
5.7. Generierung eines Linkmodells gemäß XML Schema Definition 74
5.8. Informationsgehalt aus dem Link 4 mit FDS-Anweisungen 75
5.9. Generierung eines Variationsmodells gemäß XML Schema Definition . . . 78

A. Anhang **101**
A.1. Vollständiger Quelltext der Eingabedatei 101
A.2. Darstellung der Gebäudeelemente durch die IFC 108
A.3. EXPRESS Schema für das Fachmodell fire protection concept 111
A.4. EXPRESS Schema für das Fachmodell initial fire 113
A.5. EXPRESS Schema für das Fachmodell material 114

A.6. EXPRESS Schema für ein Linkmodell 116
A.7. EXPRESS Schema für das FDS-Simulationsmodell 117

Abkürzungsverzeichnis

BIM	Building Information Modelling
CAD	Computer-Aided Design
CFD	Computational Fluid Dynamics
CSV	Comma-Separated Values
FEM	Finite-Elemente-Methode
FDM	Finite-Differenzen-Methode
FDS	Fire Dynamics Simulator
FVM	Finite-Volumen-Methode
HRR	Heat Release Rate
HRRPUA	Heat Release Rate Per Unit Area
IDE	Integrated Development Environment
IFC	Industry Foundation Classes
LK	Linkkategorie
MBO	Musterbauordnung
NIST	National Institute of Standards and Technology
REI90	Résitance Étanchéité Isolation 90 (Minuten)
SDK	Software Development Kit
STEP	Standard for the Exchange of Product model data
UML	Unified Modeling Language
XML	Extensible Markup Language
XSD	XML Schema Definition

1. Einleitung

Und jedem Anfang wohnt ein Zauber inne.

(Hermann Hesse)

1.1. Motivation

Die Grundsätze des Brandschutzes sind leicht und verständlich, jedoch gewinnen sie bei konkreten Bauvorhaben oft an Komplexität: Gebäude sind so zu errichten, „[...] dass der Entstehung eines Brandes und der Ausbreitung von Feuer und Rauch (Brandausbreitung) vorgebeugt wird und bei einem Brand die Rettung von Menschen und Tieren sowie wirksame Löscharbeiten möglich sind“ [ARG02, S. 14]. Zur Erreichung dieser Schutzziele, in Deutschland besser bekannt als die vier allgemeinen Schutzziele, werden eine Vielzahl an materiellen Anforderungen in den jeweiligen Landesbauordnungen benannt. Nicht selten sind dabei Optionen zu finden, um konkrete Anforderungen auf alternativen Wegen schutzzielgerecht zu erfüllen. In diesem Zusammenhang können Brandsimulationen die Möglichkeit bieten, innovative und maßgeschneiderte Brandschutzlösungen aufzuzeigen.

Die vorliegende Arbeit widmet sich der Thematik und zeigt, wie durch eine modellbasierte Arbeitsweise neue Informationsräume entstehen und effizient für Brandsimulationen genutzt werden können. Es ist sinnvoll, sich der Bedeutung des Begriffs Informationsraum von Beginn an klarzuwerden. Harald H. Zimmermann, Universitätsprofessor für Informationswissenschaft (in Ruhestand) definiert Information als „[...] Transfer von Wissen [...]“ [See95, S. 352]. Ein Raum verkörpert im Allgemeinen eine Abgeschlossenheit, in dem durch einzelne Bauteile geometrische Grenzen gebildet werden. Insgesamt kann hieraus geschlussfolgert werden, dass Informationsräume einen für sich eingeschränkten Wissensstand repräsentieren. Die Kombination solcher Räume führt zur Erweiterung eines bisherigen Wissensstands, was unmittelbar neue Anwendungen ermöglicht.

Der *Fire Dynamics Simulator* (FDS), eine gemeinfreie Software zur Simulation von Bränden, stellt für die Arbeit einen solchen Anwendungsfall dar. Datenmodelle erschließen dabei Informationsräume, in dem sie einzelne Daten zu einheitlichen Informationen bündeln. Durch eine Vernetzung der Modelle wächst eine Informationsressource zusammen, aus der ein Simulationsmodell generiert werden kann. Das hieraus resultierende Brandszenario kann durch geschickte Variation von Parametern in vielzählige Brandszenarien gewandelt werden, womit die Simulation effizient gestaltet wird.

Damit wird eine Herausforderung aus dem Bereich der Bauinformatik angenommen, die im Rahmen der Diplomarbeit mit wissenschaftlichen Methoden bewerkstelligt wird. Der Kandidat zeigt darin, dass er „[...] die Zusammenhänge seines Faches überblickt, die Fähigkeit besitzt, wissenschaftliche Methoden und Erkenntnisse anzuwenden und die für den Übergang in die Berufspraxis notwendigen gründlichen Fachkenntnisse erworben hat“ [Mül15, S. 13]. Der Diplomand stellt mit dem Werk auch seine persönlichen Weichen, um zukünftig den Brandschutz aktiv mitzugestalten.

1.2. Ziele und Abgrenzung der Arbeit

Das Hauptziel dieser Arbeit ist es herauszufinden, welche Datenmodelle erstellt werden müssen, um mit diesen ein Multimodell zur Simulation mit dem *Fire Dynamics Simulator* (FDS) darzustellen. Die Simulation soll sich an einem übergeordneten Grundmodell orientieren, sodass mithilfe des Variationsprinzips eine Vielzahl an Untermodellen generiert werden kann. Auf diese Weise können mehrere Szenarien zu einem Brandereignis aus einer zur Verfügung stehenden Informationsressource analysiert werden.

Ein Standardwerk zum Thema Multimodelle ist das Buch von Raimar J. Scherer und Sven-Eric Schapke, in dem die Autoren zeigen, wie mit Multimodellen neue Informationsressourcen entstehen [SS14]. Mit dem Variationsprinzip haben sich Tobias Mansperger, Ngoc Trung Luu, Al-Hakam Hamdan, Michael Polter und Raimer J. Scherer befasst. In ihrer 2018 erschienenen wissenschaftlichen Publikation zeigen sie, wie das Variationsprinzip für Eingabemodelle im Bereich von Massensimulationen eingesetzt werden kann [Sch+18]. Zu einem Meilenstein der Ingenieurmethoden im Brandschutz zählt sicherlich der im Jahr 2013 überarbeitete Leitfaden von Dietmar Hosser, in dem sehr gute Anhaltspunkte für die Erstellung einzelner Datenmodelle gegeben werden [Hos13]. Damit ist ein Überblick über die wichtigste Literatur gegeben, die zur Erreichung der oben genannten Zielsetzung erforderlich wird.

Mit der Arbeit soll die Umsetzung der theoretischen Überlegungen gezeigt werden, was die Erstellung eines eigenen Programms beinhaltet. Damit können entwickelte Ansätze verifiziert und Potenziale erkannt werden. Weiterhin ist zu hinterfragen, wie sich die *Industry Foundation Classes* (IFC) als Datenstandard für Brandsimulationen mit dem *Fire Dynamics Simulator* eignen.

Für die Simulation von Brandereignissen stellt der FDS eine Vielzahl an Befehlen bereit, die bei vollständiger Berücksichtigung zu Zielkonflikten in der Bearbeitungszeit führen. Es wird auf die Grundfunktionalität des Simulationsprogramms Bezug genommen. Der FDS stellt ein Feldmodell dar und dient der Berechnung physikalischer Feldgrößen, wie beispielsweise einer Temperatur. Eine tiefgründige Betrachtung der theoretischen Grundlagen sprengt den Umfang dieser Arbeit und leistet keinen ersichtlichen Mehrwert[1]. Eine *Preprozessor-Software*, welche eine grafische Benutzeroberfläche für den FDS ermöglicht, findet keine Anwendung. Der Einsatz stellt die Zielsetzung in den Hintergrund, da es sich dabei um eine eigenständige Software handelt.

Die vorliegende Arbeit verfolgt allgemeingültige und erweiterbare Lösungsansätze.

[1] Der *Fire Dynamics Simulator* ist eine Software zur Berechnung von thermisch angetriebenen Strömungen. Die rudimentäre Betrachtung der theoretischen Grundlagen ist für das ingenieurmäßige Grundverständnis anzustreben. Eine Einführung erfolgt im Abschnitt 2.2.1.

1.3. Aufbau der Arbeit

Der Hauptteil dieser Arbeit gliedert sich in fünf Kapitel, mit denen insgesamt ein Konzept für die effiziente Durchführung von Brandsimulationen beschrieben wird. Im zweiten Kapitel werden hierfür das systematische Vorgehen sowie die wesentlichen theoretischen Grundlagen erläutert. Der *Fire Dynamics Simulator* ist eine Software zur Berechnung von thermisch angetriebenen Strömungen. Um einen unkomplizierten Einstieg in den Themenbereich zu ermöglichen, erfolgt eine Einführung in die CFD-Methode. Hieraus ergeben sich wichtige Erkenntnisse im Hinblick auf die Anordnung einzelner Geometrien im Simulationsprogramm. Darüber hinaus werden erste Schritte im Umgang mit dem *Fire Dynamics Simulator* erläutert. Für die Konzepterstellung ist es erforderlich, die beiden Begrifflichkeiten Multimodell und Variationsprinzip zu klären. In dem Zusammenhang werden die wichtigsten Begriffe definiert. Damit wird ein grundlegendes Verständnis für den weiteren Verlauf der Arbeit vermittelt.

Im dritten Kapitel wird ein FDS-Minimalbeispiel eingeführt, mit dem die wesentlichen Programmfunktionalitäten erkannt werden sollen. Damit wird das Ziel verfolgt, eine kompilierbare Eingabedatei für die Simulation eines Zimmerbrandes zu generieren. Das Kapitel 4 untersucht den Informationsraum der *Industry Foundation Classes* (IFC) und zeigt, wie mithilfe einzelner IFC-Daten konkrete FDS-Anweisungen für die Modellierung der Gebäudeelemente erzeugt werden. Da mit der Untersuchung Angaben zu Abmessungen und Platzierungen einzelner Gebäudeelemente erwartet werden, beschäftigt sich Kapitel 5 mit weiteren Informationsräumen. Dies schließt die Erstellung eigener Fachmodelle, die ein für sich eigenständiges Anwendungsgebiet repräsentieren, mit ein. Die Arbeit zeigt, wie mithilfe von Linkmodellen Beziehungen einzelner Datenobjekte aus unabhängigen Fachmodellen hergestellt werden. Damit wird der Ansatz eines Multimodells verfolgt und die Möglichkeit, Parameter einzelner Klassen hinsichtlich der Variationsmöglichkeiten zu analysieren, eröffnet. In den Kapiteln 3, 4 und 5 werden textliche Ausführungen durch einzelne Quelltexte sowie Abbildungen untermauert.

Insgesamt wird ein Multimodell beschrieben, dass durch das Variationsprinzip erweitert werden kann und Simulationen mit dem *Fire Dynamics Simulator* ermöglicht. Im sechsten Kapitel erfolgt die Verifizierung der vorgestellten Ansätze durch eine eigene Programmentwicklung. Damit endet der Hauptteil der Arbeit und widmet sich in einer Schlussbetrachtung den wesentlichen Erkenntnissen.

2. Systematik und theoretische Grundlagen

Phantasie ist wichtiger als Wissen, denn Wissen ist begrenzt.

(Albert Einstein)

2.1. Methodische Vorgehensweise

Die vorliegende Arbeit basiert auf einem anwendungsorientierten Vorgehen, durch das ein Konzept für die effiziente Durchführung von Brandsimulationen beschrieben wird. Ziel ist es, mithilfe eines Multimodells sowie dem Variationsprinzip eine Simulation mit dem *Fire Dynamics Simulator* (FDS) zu ermöglichen. Das Konzept wird durch ein Programm umgesetzt und besteht im Kern aus der gezielten Bündelung von Informationen.

Definition:

Konzepte beschreiben Wege, nehmen Bezug auf Situationen und verfolgen erstrebenswerte Zustände. Letzteres bedeutet für diese Arbeit die Möglichkeit, Brandsimulationen im Bauwesen effizient und anwendungsorientiert gestalten zu können. Dabei definiert Effizienz eine möglichst hohe Aussagefähigkeit eines Brandereignisses und dessen Auswirkung bei einer fest zur Verfügung stehenden Informationsressource.

Konkret: Die gezielte Veränderung von Parametern in einem Simulationsmodell führt zu einer Vielzahl an Brandszenarien, ohne zusätzliche Informationen zu benötigen. Das Vorgehen beschreibt den Leitgedanken des Variationsprinzips, das auf einem übergeordneten Grundmodell basiert. Das Modell, nachfolgend als Simulationsmodell bezeichnet, ermöglicht die Durchführung von Brandsimulationen mit dem FDS und wird aus programmspezifischen Simulationsbefehlen zusammengesetzt.

Die Beschreibung einer Ausgangssituation erfolgt anhand eines FDS-Minimalbeispiels, durch das die Brandentwicklung in einem Zimmer simuliert wird. Mit dem Beispiel wird eine Bauwerksgeometrie dargestellt, eine mögliche Zündquelle abgebildet und eine Brandausbreitung durch brennbare Materialien verdeutlicht. Auf diese Weise wird die einfache Handhabung komplexer Zusammenhänge ermöglicht, was die Erstellung eines allgemeingültigen Konzepts erleichtern soll. Auf den Einsatz einer *Preprozessor-Software* wird verzichtet, sodass Simulationsbefehle nur durch eine Eingabedatei ausgeführt werden können[1]. Das Erlernen einer programmspezifischen Eingabesyntax wird bereits für das Beispiel vorausgesetzt und verdeutlicht wesentliche Zusammenhänge über die Struktur des Simulationsprogramms.

Mit dieser Arbeit soll untersucht werden, inwieweit die *Industry Foundation Classes* (IFC) für Brandsimulationen im Bauwesen geeignet sind. Dies veranlasst den Einsatz einer CAD/BIM Software, mit der die Geometrie von Gebäudeelementen in einem Bauwerksmodell abgebildet werden kann. Die Anwendung erlaubt den Datenaustausch in die IFC, mit denen Bauwerksinformationen in ein maschinenlesbares Format übertragen werden. Der Datenstandard wird in den Versionen IFC2x3 und IFC4 unterstützt, wobei die Arbeit auf dem neueren Schema IFC4 aufbaut. Der eigentliche Datenaustausch erfolgt über die Generierung einer Austauschdatei, mit der das Datenmodell von weiteren

[1] Damit wird der Grundgedanke einer freien Softwareanwendung verfolgt, siehe auch 1.2

Softwareanwendungen eingelesen und genutzt werden kann. Die Inhalte der Datei sind mithilfe der Dokumentationen interpretierbar, sodass die Informationsübertragung nachvollzogen werden kann. Auf diese Weise werden Simulationsparameter ermittelt, die durch die Verwendung der IFC zur Verfügung gestellt werden können. Hierbei wird die Bauwerksgeometrie fokussiert, um durch einen gezielten Einsatz von Programmroutinen die Geometrie in das Simulationsmodell zu übertragen. Des Weiteren wird die prinzipielle Darstellung von Brandlasten untersucht, was sich insbesondere auf deren Abmessungen und Lage bezieht. Fehlen Angaben zu einzelnen Simulationsparametern innerhalb der IFC entsteht eine Informationslücke, was die weitere Vorgehensweise begründet und zum Kern dieser Arbeit führt – *dem Multimodellansatz*.

Ein Multimodell bietet die Möglichkeit, fehlende Simulationsparameter in Fachmodellen bereitzustellen und zueinander in Beziehung zu setzen. Derartige Fachmodelle begrenzen einzelne Anwendungsgebiete und stellen Datenmodelle dar, durch die Informationen repräsentiert werden. Informationen entstehen aus Datenobjekten, deren spezifischen Attribute sowie der konkreten Zuweisung von Werten. Objekte orientieren sich an übergeordneten Strukturen, auch Klassen genannt und stellen anwendungsorientierte Daten zur Verfügung. Für die Erstellung der Fachmodelle und deren Datenstrukturen sind demnach im Vorfeld Daten zu erschließen, die das jeweilige Fachmodell bereitstellen soll. Hierfür werden in einem ersten Schritt alle Simulationsparameter betrachtet, die durch die IFC unberücksichtigt bleiben. Deren Ursprung wird untersucht, was u.a. eine Brücke zu den Ingenieurmethoden des Brandschutzes schlägt und essenzielle Zusammenhänge offen legt. Ingenieurmäßige Überlegungen führen zu weiteren Informationen mit denen zusätzliche Daten erschlossen werden können.

Die Bildung von Fachmodellen und deren Datenstruktur wird mit dieser Datenbasis ermöglicht. In diesem Zusammenhang stellt die Programmierschnittstelle *JSDAI* [LKS] ein Werkzeug dar, um das Datenaustauschformat *STEP Physical File* in der Modellierungssprache *EXPRESS* anzuwenden. Hiermit werden innerhalb von Entwicklungsumgebungen Datenstrukturen einzelner Fachmodelle erzeugt, grafisch abgebildet sowie die Funktionsweise überprüft. Auf die Erstellung eines einzigen Fachmodells wird verzichtet, um die universelle Einsatzmöglichkeit der Modelle zu gewährleisten. Auch werden somit überschaubare Datenstrukturen erzeugt, die für ihren Anwendungsbereich unkompliziert zu erweitern bzw. modifizierbar sind.

Die wichtigste Grundvoraussetzung ist die eindeutige Identifikationsmöglichkeit einzelner Datenobjekte, die nachfolgend als Elemente bezeichnet werden. Ein Identifikator (ID) ermöglicht es, ein Element eines Fachmodells gezielt mit anderen Elementen weiterer Fachmodelle in Beziehung zu setzen. Die Modelle bleiben dabei unabhängig voneinander, jedoch wird der Informationsgehalt eines jeden Elements eindeutig zugänglich. Auf diese Weise können Informationen aus den jeweiligen Modellen gebündelt werden, um somit an erforderliche Parameter für die Generierung von Simulationsbefehlen zu gelangen.

Beispielsweise stehen durch die Verknüpfung von Materialparametern mit einem Wandbauteil alle Daten für die Simulation eines individuellen Brandverhaltens zur Verfügung. Für die Erstellung dieser Verknüpfungen, den sogenannte Links, stützt sich diese Arbeit auf das von dem *buildingSMART e.V.* bereitgestellte Linkmodell [buib]. Damit wird auf eine bereits erstellte Datenstruktur zugegriffen, was sich insbesondere auf die Umsetzung des angedachten Programms auswirkt. Aufgrund der eingesetzten Programmierschnittstelle werden die Fachmodelle ausschließlich durch das Datenaustauschformat *STEP Physical File* dargestellt. Die Verwendung anderer Formate sowie deren Modellierungssprachen ist möglich, führt im Rahmen der Arbeit jedoch zu keinem ersichtlichen Mehrwert. Insgesamt wird damit die Grundidee eines Multimodells umgesetzt, womit eine einheitliche Informationsressource entsteht und gleichzeitig das Simulationsmodell generiert werden kann.

Das Simulationsmodell spiegelt sich in der Eingabedatei für den FDS wider, basiert auf der entstandenen Informationsressource und simuliert genau ein einziges Brandszenario. Realitätsnahe Schwankungen können dabei zu einer Parameterunschärfe in einzelnen Simulationsbefehlen führen, sodass ein dargestelltes Brandereignis an Aussagekraft verliert. Mithilfe des in der Arbeit angewandten Variationsprinzips sind Varianten des ursprünglichen Brandszenarios möglich, die diesen Informationsverlust kompensieren. Die eigentliche Informationsressource bleibt unverändert, lediglich einzelne Attribute der Fachmodelle werden in ihrer Wertigkeit variiert. Nach der Erstellung von Fachmodellen können somit Elementattribute analysiert werden, die durch eine Werteänderung die Aussagefähigkeit eines Brandereignisses erhöhen. Zum Beispiel unterliegt die Menge einer Brandlast immer einer Momentaufnahme und deren Beschränkung auf einen festen Wert ist praxisfern. Diese Vorgehensweise ermöglicht das Erzeugen mehrerer Eingabedateien, die sich im Gesamten an einem übergeordneten Simulationsmodell orientieren. Aus einer verfügbaren Informationsressource entsteht somit eine Vielzahl an Brandszenarien, die insgesamt die Aussagekraft für ein Brandereignis erhöhen[2].

Die Konzeptumsetzung erfolgt durch ein Programm, welches das Multimodell anwendet und kompilierbare FDS-Eingabedateien generiert. Im Fokus stehen der zielgerichtete Einsatz von Filterroutinen und die Transformation einzelner Informationsbündel in konkrete FDS-Simulationsbefehle. Als Programmiersprache wird *Java* gewählt, um damit eine plattformunabhängige und objektorientierte Programmierweise zu verfolgen. Die Programmierschnittstelle *JSDAI* [LKS] erlaubt den Zugriff auf Daten der einzelnen Modelle und wird somit angewendet. Funktionsweisen der Programmroutinen werden an den Gebäudeelementen aus Kapitel 4 erprobt.

Die Abbildung 2.1 fasst die beschriebene Vorgehensweise zusammen. Ausgehend von dem FDS-Minimalbeispiel veranschaulicht sie den Weg für die Erstellung des Multimodells, das eine effiziente Simulation von Brandereignissen mit dem *Fire Dynamics Simulator* ermöglicht. In der Tabelle 2.1 wird ein Überblick über die relevante Software gegeben.

[2] Die damit verbundene Erhöhung an Rechenkapazitäten ist kein Bestandteil der Arbeit.

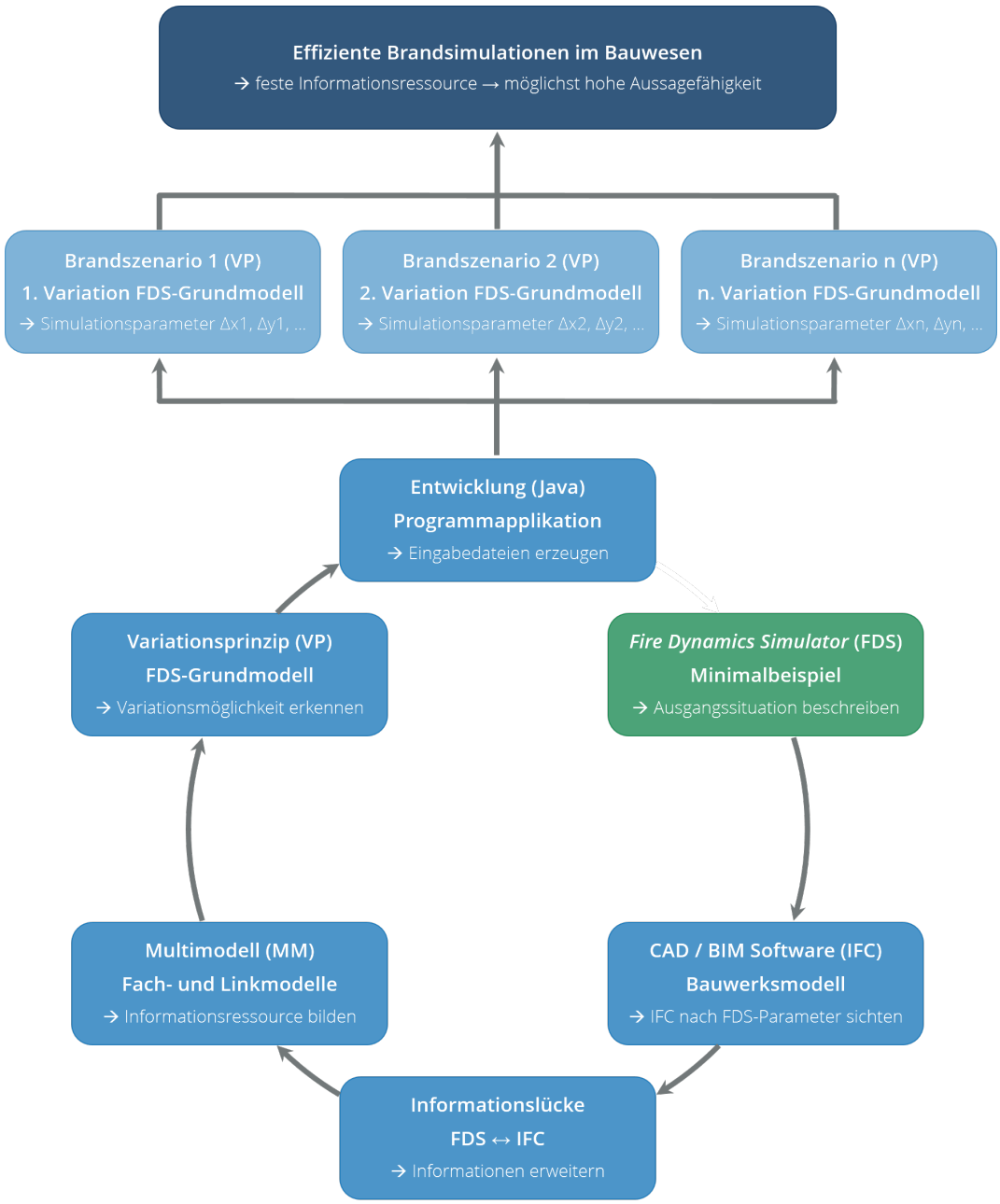

Abb. 2.1.: Methodische Vorgehensweise.

Tabelle 2.1.: Übersicht Software.

Bereich	Kurzbeschreibung	Quelle
Simulation	Fire Dynamics Simulator (FDS) and Smokeview (SMV)	[SC]
CAD/BIM	Graphisoft Archicad 21	[Gra]
IFC	Industry Foundation Classes Release 4 (IFC4)	[buia]
Fachmodell	JSDAI for Eclipse	[LKS]
Linkmodell	LinkModel-2.0.0.xsd	[buib]
Variationsmodell	VariationModelV2.xsd	[Sch+]
Programmierung	Eclipse Oxygen.3a (4.7.3a)	[Fou]

2.2. Brandsimulationen mit dem Fire Dynamics Simulator

Der *Fire Dynamics Simulator* (FDS) ist nach McGrattan et al. eine Software zur Berechnung von thermisch angetriebenen Strömungen. Der Schwerpunkt liegt auf dem Rauch- und Wärmetransport von Bränden, wobei sich das Simulationsprogramm der numerischen Strömungsmechanik (CFD) bedient. Damit wird eine Form der *Navier-Stokes-Gleichungen* gelöst (2017, S. 3).

Die vorliegende Arbeit gibt mit den beiden nachfolgenden Abschnitten eine Einführung in die CFD-Methode und verdeutlicht dabei die hieraus resultierende Geometrieform im *Fire Dynamics Simulator*. Im Abschnitt 2.2.3 wird rudimentär gezeigt, wie Simulationen erstellt und durchgeführt werden können. Insgesamt werden die theoretischen Grundlagen, mit denen die Einführung eines FDS-Minimalbeispiels ermöglicht wird, wiedergegeben.

2.2.1. Einführung in die CFD-Methode

„Die numerische Strömungsmechanik, englisch *Computational Fluid Dynamics* (CFD), ist eine etablierte Methode der Strömungsmechanik. Sie hat das Ziel, strömungsmechanische Probleme approximativ mit numerischen Methoden zu lösen“ [W18]. „Die Entwicklung der numerischen Methoden in der Strömungsmechanik ging von den Finite-Differenzen-Methoden (FDM) über die Finite-Volumen-Methoden (FVM) bis hin zu den adaptiven Finite-Elemente-Methoden (FEM) für instationäre dreidimensionale Strömungsprobleme“ [OL03, S. 2]. „Die Verfahren, welche auf räumlichen Netzen basieren, lassen sich bezüglich Genauigkeit und Flexibilität [. . .] einordnen [OL03, S. 131f]. „Bei Finite-Differenzen-Methoden muss das Rechengebiet, in einen kartesischen Rechenraum transformiert werden, in dem dann sehr genau approximiert werden kann. Diese Transformation schränkt die Flexibilität ein“ [OL03, S. 132].

„Weil im vorbeugenden baulichen Brandschutz vorwiegend Gebäude untersucht werden, deren Wände und Bauteile parallel zu den Koordinatenebenen verlaufen (die nicht schräg oder gekrümmt sind), ist die Anwendung der FDM hier naheliegend. Das FDM-Verfahren wird auch in der frei verfügbaren Software FDS des NIST verwendet“ [GG12, S. 51]. „Für die FDM ist ein sog. kartesisches Netz erforderlich [. . .], also ein Netz, in dem alle Seitenflächen der Zellen parallel zu den Koordinatenebenen sind“ [GG12, S. 52].

Simulationen mit dem FDS sind demnach auf strömungsmechanische Berechnungen zurückzuführen, deren Lösungen mithilfe von Finite-Differenzen-Methoden numerisch angenähert werden. Dieser Einsatz erlaubt eine hohe Rechengenauigkeit. Aufgrund des kartesischen Netzes sind jedoch Geometrien nur über Würfel bzw. Quader darstellbar.

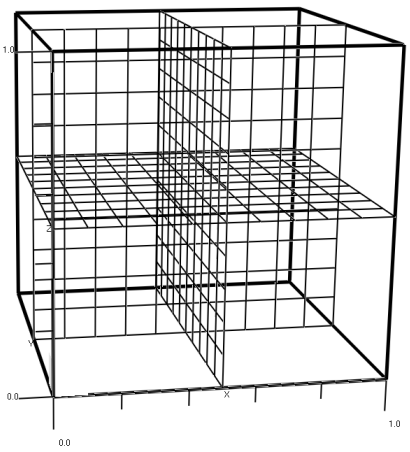

Abb. 2.2.: Beispiel für einen kartesischen Rechenraum.

Abbildung 2.2 verdeutlicht ein derartiges Netz im FDS. Im vorliegenden Fall werden Würfel mit einer Seitenlänge von zehn Zentimetern generiert, in dem die ein Meter langen Seitenflächen jeweils in zehn Elemente unterteilt werden. Für Computersimulationen der Brand- und Rauchausbreitung geben die beiden Autoren Grewolls eine Netzauflösung von 5-25 cm als Stand der Technik an (2012, S. 54).

In dem Zusammenhang erläutern sie die physikalischen Grundlagen für die Berechnung von Feldgrößen, die sich aus den Erhaltungsgleichungen für die Masse, den Impuls und die Energie ergeben. „Alle Variablen, welche in den Erhaltungsgleichungen auftreten, wie z.B. die Temperatur, der Druck, die Geschwindigkeitsvektoren, sind örtlich und zeitlich veränderlich. Sie können an jedem Punkt des Raumes unterschiedlich sein und sich im Zeitverlauf des Brandereignisses verändern“ [GG12, S. 50]. Als Folge nennen die Autoren das Aufstellen gekoppelter Systeme von partiellen nichtlinearen Differentialgleichungen, was unmittelbar zur Beschreibung von Anfangs- und Randbedingungen führt. Analytische Lösungen auf händischem Wege sind demnach nicht mehr möglich, was den eigentlichen Einsatz der Finite-Differenzen-Methoden als numerische Lösungsmethode rechtfertigt (2012, S. 49ff).

Oertel und Laurien widmen sich in ihrem Buch den Grundgleichungen der Strömungsmechanik (2003, S. 31). „Für die Berechnung von Strömungen gelten die kontinuumsmechanischen *Erhaltungssätze* für *Masse*, *Impuls* und *Energie*. Es werden die drei Geschwindigkeitskomponenten u, v und w des Geschwindigkeitsvektors ν, der Dichte ρ, der Druck p und die Temperatur T in Abhängigkeit der drei kartesischen Koordinaten und der Zeit ermittelt“ [OL03, S. 31]. Sie führen diesbezüglich allgemeine Formulierungen für die Erhaltungssätze an einem differentiellen Volumenelement ein, auf denen die Herleitungen einzelner Differentialgleichungen folgen[3] (2003, S. 31-47).

[3] Auf eine weitere Ausführung wird verzichtet, da die verbale Formulierung der Erhaltungssätze einen ausreichenden Einblick in die theoretischen Grundlagen gewährt.

Formulierung für den Erhalt einer Masse:

„Die zeitliche Änderung der Masse im Volumenelement =
$\sum$ der einströmenden Massenströme in das Volumenelement -
$\sum$ der ausströmenden Massenströme aus dem Volumenelement.“ [OL03, S. 31]

Formulierung für den Erhalt eines Impulses:

„Die zeitliche Änderung des Impulses im Volumenelement =
$\sum$ der eintretenden Impulsströme in das Volumenelement -
$\sum$ der austretenden Impulsströme aus dem Volumenelement +
$\sum$ der auf das Volumenelement wirkenden Scher- und Normalspannungen +
$\sum$ der auf die Masse des Volumenelements wirkenden Kräfte.“ [OL03, S. 33]

Formulierung für den Erhalt von Energie:

„Die zeitliche Änderung der Gesamtenergie im Volumenelement =
$\sum$ der durch die Strömung ein- und ausfließenden Energieströme +
$\sum$ der durch Wärmeleitung ein- und ausfließenden Energieströme +
$\sum$ der durch die Druck-, Normalspannungs- und Schubspannungskräfte am Volumenelement geleisteten Arbeiten pro Zeit + der Energiezufuhr von außen + Arbeit pro Zeit, die durch das Wirken der Volumenkräfte verursacht wird.“ [OL03, S. 43]

„Die Grundgleichungen der Strömungsmechanik sind partielle *Differentialgleichungen* zweiter Ordnung. Sie enthalten also partielle *Differentialoperatoren*, in denen die Änderungen der Variablen *kontinuierlich* in den räumlichen Richtungen und in der Zeit ausgedrückt werden“ [OL03, S. 126]. „Die Diskretisierung der Strömungsgrößen und ihrer Ableitungen bezüglich der Koordinaten x, y und z zu einem konstanten Zeitpunkt t bezeichnet man als *räumliche Diskretisierung*. Voraussetzung ist die Definition eines zwei- oder dreidimensionalen Netzes. “[OL03, S. 130]. Oertel und Laurien beschreiben dabei eine wesentliche Eigenschaft von Finite-Differenzen-Methoden: „Die Differentialquotienten werden durch *Differenzenquotienten* ersetzt, welche auf den an Gitterpunkten definierten Variablen basieren. An jedem dieser Gitterpunkte werden die Grundgleichungen näherungsweise erfüllt“ [OL03, S. 131]. In diesem Zusammenhang verdeutlichen sie den geringen Aufwand für die Herleitung von sehr genauer Differenzverfahren bei Verwendung einfacher Geometrien. Als wesentliche Nachteile nennen sie den erhöhten Berechnungs- sowie damit verbundenen Programmieraufwand bei komplexen Transformationen zwischen den Rechen- und physikalischen Räumen (2003, S. 133ff).

Insgesamt wird damit die rechteckförmige Darstellungsweise von geometrischen Körpern im FDS begründet sowie ein grundlegender Einblick in dessen Berechnungsmethoden gegeben. Die vorliegende Arbeit grenzt sich an dieser Stelle von einer weiterführenden Beschreibung der Thematik ab (vgl. Abschnitt 1.2).

2.2.2. Räumliche Anordnung von Geometrien

Simulationen mit dem FDS setzen eine Anordnung von Geometrien in einem kartesischen Koordinatensystem voraus (vgl. Abschnitt 2.2.1). Die Abbildung 2.3 zeigt hierzu die grundlegende Darstellungsweise eines geometrischen Körpers sowie die damit erforderlichen Koordinatenpunkte X_1, X_2, Y_1, Y_2, Z_1 und Z_2. Alle Abmessungen werden in der Einheit Meter angegeben und orientieren sich an den Achsen (x, y, z) des Koordinatensystems. Die räumliche Lage einer Geometrie wird durch einen Bezugspunkt mit den Koordinaten X_1, Y_1 und Z_1 beschrieben. Bezogen auf diesen Punkt spannen die Koordinaten X_2, Y_2 und Z_2 einen durch sechs Rechtecke begrenzten Körper auf. Dadurch werden Geometrien in der Form eines Quaders bzw. Würfels dargestellt. Auf eine explizite Differenzierung zwischen einem globalen und einem lokalen Koordinatensystem wird verzichtet. Damit wird eine unkomplizierte räumliche Anordnung einzelner Geometrien verfolgt, deren Größenordnungen in der jeweiligen Richtung wie folgt bestimmt werden.

$$\Delta x = |\, X_2 - \; X_1| \tag{2.1}$$

$$\Delta y = |\, Y_2 - \; Y_1| \tag{2.2}$$

$$\Delta z = |\, Z_2 - \; Z_1| \tag{2.3}$$

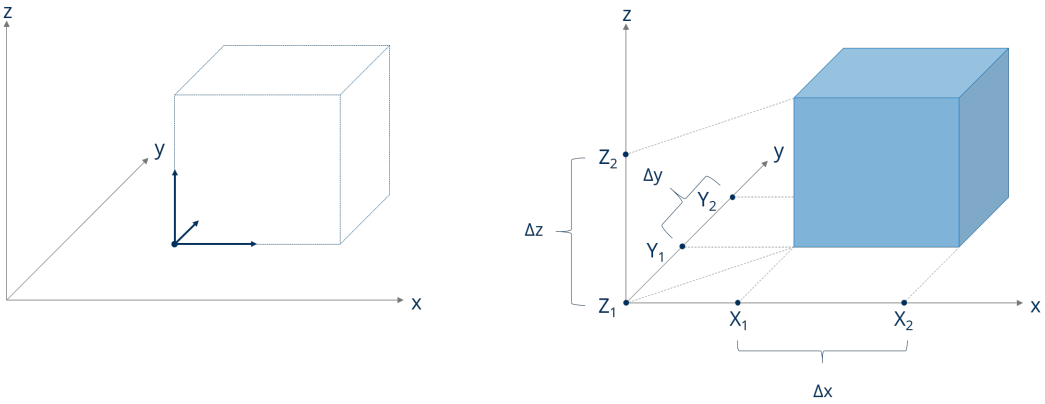

Abb. 2.3.: Räumliche Anordnung der Geometrien.

Es wird festgestellt, dass die Darstellung gekrümmter Flächen und schräger Geometrieverläufe im Grundsatz ausgeschlossen ist. In diesem Zusammenhang weisen die beiden Autoren Grewolls auf die Anordnung von treppenförmigen Geometrien hin (2012, S. 52). Es ist jedoch kritisch zu hinterfragen, inwieweit eine geometrische Approximation die Eigenschaften der tatsächlichen Geometrie beeinflusst. Das Erzeugen treppenförmiger Geometrien führt zu einer Vergrößerung der geometrischen Verhältnisse und beeinflusst damit das Brandereignis.

2.2.3. Erste Schritte mit dem Simulationsprogramm

Nach McGrattan et al. basiert jede FDS-Simulation auf einer textbasierten Eingabedatei (.fds), die mithilfe von Texteditoren oder grafischen Benutzeroberflächen erstellt werden kann[4]. Beispiel-Eingabedateien sind Bestandteil der Standardinstallation und eignen sich für die Erstbenutzung des Simulationsprogramms. Eine FDS-Eingabedatei wird über die Eingabeaufforderung des Betriebsprogramms mit dem Befehl `fds Dateinname.fds` kompiliert (2017, S. 13). In der Folge werden Anweisungen zu einem Simulationsmodell umgesetzt und weitere Dateien im Installationspfad hinterlegt. Standardmäßig wird eine Datei (.smv) generiert, die mit dem Kommando `smokeview dateiname.smv` ausführbar ist. In dem Zusammenhang nennen McGrattan et al. das Visualisierungsprogramm *Smokeview* (SMV), das die Ergebnisse der FDS-Simulation veranschaulicht (2017, S. 3).

Die Abbildung 2.4 verdeutlicht exemplarisch den Ablauf für die Beispiel-Eingabedatei Couch. In den beiden Eingabeaufforderungen werden der Kompiliervorgang (l.o.) und die Ausführung des Visualisierungsprogramms (l.u.) dargestellt. Die Ergebnisse zu dem Simulationsmodell zeigen den Sofabrand bei einer Simulationszeit von 90 Sekunden.

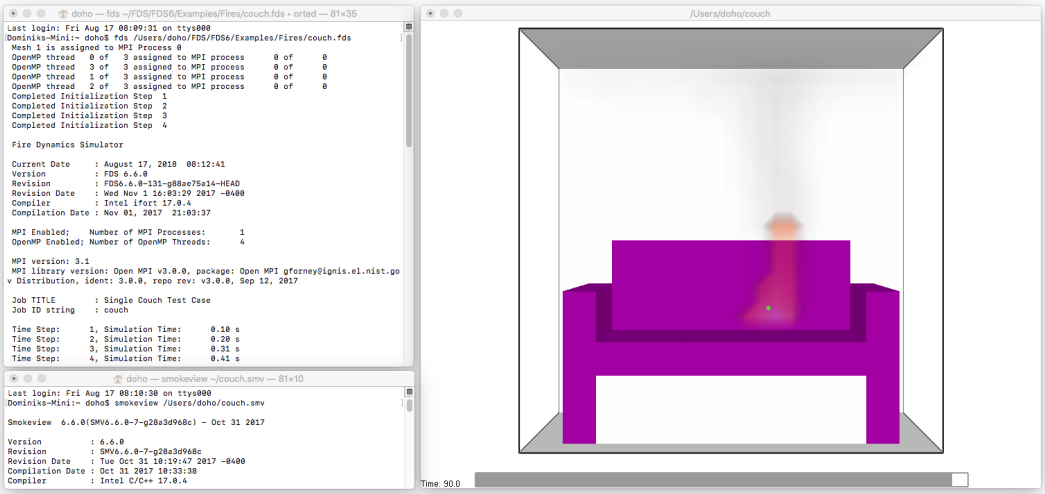

Abb. 2.4.: Ablauf der FDS-Simulation für die Beispiel-Eingabedatei Couch.

Die vorgestellten Grundlagen ermöglichen zusammen mit dem *User's Guide* [McG+17] die Einführung von einem Minimalbeispiel für den *Fire Dynamics Simulator*. Ziel des Beispiels ist es herauszufinden, welche FDS-Anweisungen zur Simulation eines Zimmerbrandes erforderlich sind. Damit werden zum einen einzelne Eingabebefehle analysiert und zum anderen deren Wirkung zueinander berücksichtigt. Auf diese Weise soll die Generierung von Simulationsmodellen aus der Informationsressource eines Multimodells ermöglicht werden.

[4] Auf den Einsatz einer *Preprozessor-Software* wird verzichtet, um grundlegende Zusammenhänge für die Informationen zu einzelnen FDS-Anweisungen zu erhalten.

2.3. Das Multimodell mit dem Variationsprinzip

„BIM ist die ganzheitliche Arbeitsweise mit allen Daten des gesamten Baulebenszyklus in digitaler Form“ [SS14, S. V]. Die darin enthaltene Abkürzung BIM definieren Scherer und Schapke als *Building Information Modelling* und stellen damit den Multimodellansatz vor (2014, S. Vf). Mit dem Ansatz beschreiben sie eine Methode für die Ermöglichung einer prozessorientierten und vollständigen digitalen Arbeitsweise. „Grundidee des Multimodells ist es, ausgewählte Fachmodelle aus der Planung und aus dem Projektmanagement in einer einzigen Informationsressource zu kombinieren und ihre Abhängigkeiten durch ergänzende explizite Linkmodelle abzubilden“ [SS14, S. 14].

Das Institut für Bauinformatik der Technischen Universität Dresden veröffentlicht in einer wissenschaftlichen Publikation zum Thema der Überwachung von Brückenbauwerken, die Methode des Variationsprinzips. „*Using this method results in the automatized creation of numerous input models for mass simulation*“ [Sch+18, S. 1]. Damit zielt die Methode auf die automatische Generierung von zahlreichen Eingabemodellen für Massensimulationen ab, sodass eine Schnittmenge für die effiziente Durchführung von Brandsimulationen abgeleitet werden kann (vgl. Abschnitt 2.1). Das Prinzip sieht die gezielte Variation von Variablen vor und nimmt Bezug auf Datenressourcen, die auf einem BIM-Modell beruhen.

Die vorliegende Arbeit nutzt die beiden Ansätze und beschreibt damit ein Konzept für die Durchführung von Simulationen mit dem FDS. Ein Multimodell kombiniert dabei einzelne Fachmodelle zu einer Informationsressource, aus der ein Simulationsmodell generiert werden kann. Realitätsnahe Schwankungen können durch den Einsatz des Variationsprinzips abgebildet werden, was zu einer Vielzahl möglicher Szenarien des Brandereignisses führt. Die nächsten Abschnitte widmen sich den Begriffserklärungen der beiden Ansätze, was insgesamt ein besseres Grundverständnis für die Arbeit bereitstellt.

2.3.1. Begriffserklärung Multimodell

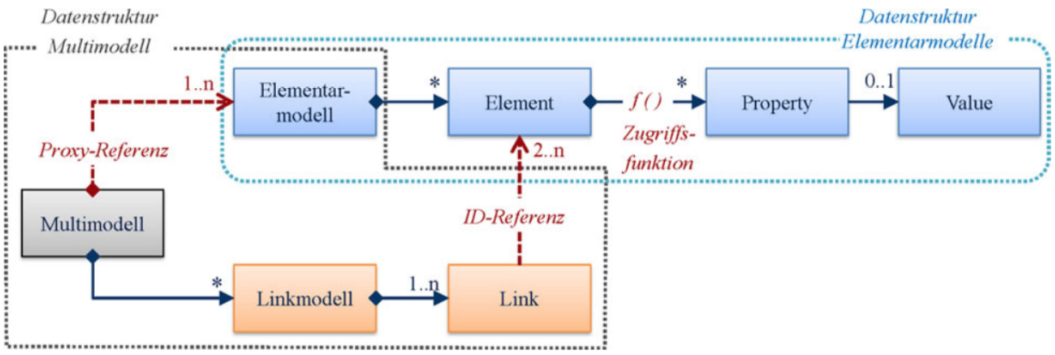

Abb. 2.5.: „Schematischer Aufbau von Multimodellen“ [SS14, S. 375].

Um einen unkomplizierten Einstieg in den Themenbereich zu ermöglichen, wird der prinzipielle Aufbau von Multimodellen anhand der Abbildung 2.5 gezeigt. Ein Multimodell setzt sich demnach aus mindestens einem (1) oder einer Vielzahl (n) von referenzierten Elementarmodellen (EM) zusammen. Die Einbindung von Linkmodellen ist erlaubt, jedoch besteht keine explizite Forderung hinsichtlich ihrer Existenz (*). Wird innerhalb eines Multimodells ein Linkmodell vorgesehen, stellt sich dies durch mindestens einen (1) bzw. einer unbegrenzten Anzahl (n) von Links dar. Ein Link zeigt dabei auf einzelne Elemente aus bestehenden EM, wobei mindestens zwei (2) referenzierte Elemente aus unterschiedlichen EM vorgeschrieben sind.

Insgesamt wird damit die Datenstruktur von Multimodellen vereinfacht beschrieben und die Anordnung der Daten innerhalb des Modells gezeigt. Die Struktur ist auf das „[...] generische Multimodell [...]“ [SS14, S. 40] zurückzuführen, welches die Einführung einer Definition von Multimodellen und der damit verbundenen Fachbegriffsdefinitionen ermöglicht. „Das generische Multimodell bildet die Systematik für allgemeine Multimodelle ohne Beschränkung auf ein Anwendungsgebiet“ [SS14, S. 44].

Definition:

„Ein Multimodell *MM* ist eine serialisierbare, übertragbare Zusammensetzung aus einer nicht leeren Menge von Elementarmodellen *EM* und einer leeren oder nicht leeren Menge von Linkmodellen mit Elementen von *EM* als Subjekt.“ [SS14, S. 41]

Definition:

„Ein Elementarmodell *EM* ist eine übertragbare Instanz eines Datenmodells, das einen abgegrenzten Informationsraum mittels einer vereinbarten Semantik beschreibt, der nicht weiter unterteilt wird“. [SS14, S. 40]

Vereinfacht ausgedrückt stellen EM Informationsquellen innerhalb von Multimodellen dar, in denen Daten zu Informationen gebündelt werden und dabei abgeschlossene Bereiche beschreiben. Abbildung 2.5 verdeutlicht die Datenstruktur von EM, die sich so „[...] genügend abstrakt und den Erfordernissen des generischen Multimodells [...]“ [SS14, S. 375] darstellt. Ein EM kann (*) dabei aus einer Vielzahl an Elementen bestehen, die durch eine uneingeschränkte Anzahl von Eigenschaften repräsentiert werden können (*). Die Art und Weise wie sich ein sogenanntes *Property* darstellt, wird über eine Zugriffsfunktion im jeweiligen Modell geregelt. Es ist möglich (0...1), jede Eigenschaft mit einem konkreten Wert zu versehen, sodass aus einzelnen Daten individuelle Informationen entstehen können. Weiterhin setzen Scherer und Schapke die eindeutige Identifikation der Elemente als wesentliches Kriterium voraus (2014, S. 375). Alle Informationen eines Elements können somit in einem EM eindeutig identifiziert sowie modellübergreifend verlinkt werden. „Weiterhin wird davon ausgegangen, dass die verlinkbaren Elemente

eine innerhalb des Elementarmodells eindeutige ID besitzen“ [SS14, S. 375]. Eine ID ist somit der Zugang zu der, in einem EM vorhandenen, Information und Voraussetzung für Verknüpfungen in Linkmodellen.

Definition:

„Ein Linkmodell *LM* ist eine serialisierbare, übertragbare Instanz eines Datenmodells mit zugehörigem Schema, das Referenzen zwischen Elementen verschiedener Elementarmodelle, sogenannte Links, speichert.“ [SS14, S. 41]

Definition:

„Ein Link *L* ist eine Menge von Identifikatoren von Elementen aus verschiedenen Elementarmodellen, die miteinander verknüpft sind.“ [SS14, S. 41]

Mit der beschriebenen Datenstruktur von EM kann eine „[...] möglichst große Anzahl von Fachmodellen [...]“ [SS14, S. 375] repräsentiert werden. Diesen Begriff führen Scherer und Schapke im Zusammenhang mit der Spezialisierung von Multimodellen ein. Sie empfehlen, abgeschlossene Informationsräume durch eine eindeutige Datenübertragung abzugrenzen und näher zu beschreiben (2014, S. 44-48).

Definition:

„Ein Fachmodell *FM* ist eine übertragbare Instanz eines Datenmodells, dessen Informationsraum ein Fachgebiet oder eine Fachdomäne darstellt. Es kann ein Elementarmodell als auch ein Multimodell sein.“ [SS14, S. 46]

$$FM \in \{MM, EM\} \tag{2.4}$$

Die vorliegende Arbeit orientiert sich an dem Grundgedanken des generischen Multimodells und beschreibt somit die Bündelung von Informationsquellen. Bei den hierbei entstandenen Modellen handelt es sich damit streng genommen um Elementarmodelle, die jedoch nachfolgend als Fachmodelle bezeichnet werden. Dies ist nach 2.4 prinzipiell zulässig und hebt den jeweiligen Anwendungsbereich des Modells hervor. In dieser Arbeit wird von einer Auslegung der Fachmodelle als Multimodelle, aufgrund der damit verbundenen Komplexität, vorerst abgesehen.

Abschließend ist festzuhalten, dass Multimodelle isolierte Informationsräume durch eine Verlinkung der Elemente miteinander verbinden. Die Gesamtheit der Informationsräume repräsentiert den möglichen Simulationsgrad für den *Fire Dynamics Simulator*.

2.3.2. Begriffserklärung Variationsmodell

Zur effizienten Durchführung von Massensimulationen stellen Scherer et al. ein Variationsmodell für den Austausch und die Speicherung von variablen Daten vor. Sie beschreiben das Modell als Hilfsmodell, da es Verknüpfungen mit Datenquellen bzw. Parametern vorsieht (2018, S. 3f). Scherer et al. kategorisieren das Modell in die vier Hauptteile *Resource*, *Variable*, *Variation* sowie *Target* und zeigen an einem UML-Klassendiagramm deren Zusammenhänge auf (vgl. Abbildung 2.6). Über *Resource* wird ein Bezug zu Datenmodellen hergestellt, die z.B. Fachmodelle in einem Multimodell sein können. Variablen stellen externe Datentypen für einen Parameter der Instanz in einer *Resource* dar (2018, S. 4f).

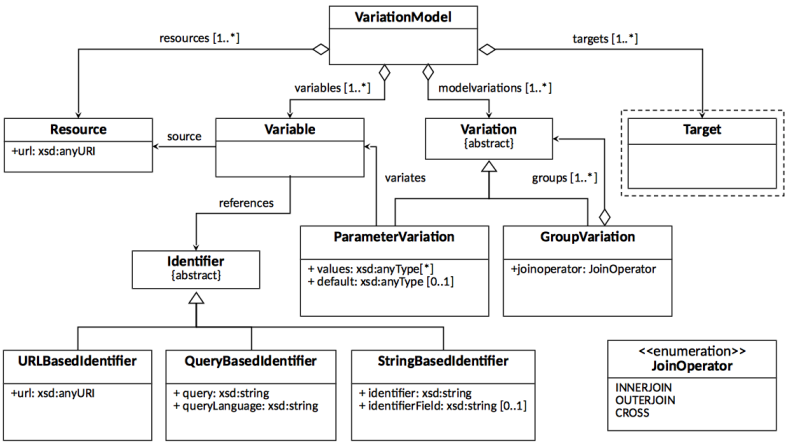

Abb. 2.6.: Das generische Variationsmodell in einem UML-Klassendiagramm [Sch+18, S. 4].

Seitens der Arbeit ist festzuhalten, dass eine Variable lediglich als Zeiger fungiert und damit eine einseitige Beziehung zur Datenquelle herstellt. Konkret: Das Variationsmodell besitzt Kenntnis über die Fachmodelle sowie deren Elemente und Attribute. Fachmodelle hingegen sind unabhängig, d.h. sie besitzen keine Kenntnisse über das Variationsmodell. Der Abbildung 2.6 ist zu entnehmen, dass Referenzierungen durch mehrere Arten von *Identifier* erlaubt werden. Die vorliegende Arbeit sieht in einem *StringBasedIdentifier* die Möglichkeit, problemlos die Beziehungen zu den Datenquellen in einem Multimodell herzustellen. Das Attribut *identifier* kann dabei für die ID einer Instanz, ein *identifierField* für die Kennung eines spezifischen Parameters verwendet werden. Nach Scherer et al. werden bei der Parametervariation den einzelnen Variablen, Werte in Form einer Liste vorgegeben. Der Variable werden damit nacheinander Listenwerte zugeschrieben, sodass aus jeder Angabe eine individuelle Modellvariante entsteht. Mehrere Parametervariationen können zu Gruppenvariationen kombiniert und über ein *JoinOperator* gesteuert werden (2018, S. 5). Sowohl die *GroupVariation* als auch Möglichkeiten für den Austausch und die Definition von Simulationszielen (Target) spielen für die Konzeptentwicklung eine untergeordnete Rolle. Die Arbeit fokussiert sich auf die Variation einzelner Parameter und verfolgt damit einen unkomplizierten Ansatz, um das Multimodell mit dem Variationsprinzip für Simulationen im *Fire Dynamics Simulator* zu vereinen.

3. FDS-Minimalbeispiel

Im Kleinen ist man nicht allein.

(Johann Wolfgang von Goethe)

3.1. Einführung in das Beispiel

Im Abschnitt 2.1 wird ein Minimalbeispiel für den *Fire Dynamics Simulator* vorgesehen, mit dem ein Zimmerbrand simuliert und Programmfunktionalitäten erkannt werden sollen. Alle notwendigen Eingabebefehle werden dem *User's Guide* [McG+17] entnommen und in einer kompilierbaren Textdatei hinterlegt. Neben der Beschreibung des gewählten Simulationsprofils folgen detaillierte Angaben zur Bauwerksmodellierung sowie zu den Darstellungen der Brandlasten und Zündquellen. Dies erfolgt unter Bezugnahme auf einzelne Quelltexte und bildlichen Veranschaulichungen. Um den Rechenaufwand gering zu halten, wird eine Simulationszeit von einer Minute vorgegeben. Materialparameter werden so gewählt, dass sich Stoffe bereits nach wenigen Sekunden Brandeinwirkung entzünden können. Aufgrund der Anpassungen ist darauf hinzuweisen, dass die am Ende des Kapitels gezeigten Simulationsergebnisse an Aussagekraft verlieren[1]. Die Grundidee des Minimalbeispiels ist es, Materialien unter einer definierten Brandeinwirkung selbstständig entzünden zu lassen sowie die daraus resultierende Ausbreitung von Feuer und Rauch zu simulieren. Ein elektrisches Gerät dient dabei als Zündquelle, die beispielsweise durch einen technischen Defekt hervorgerufen wird. Ein darüber liegendes Regal soll sich, aufgrund der freigesetzten Wärme, selbstständig entzünden und das Brandereignis im Zimmer ausbreiten. Als weiteres Kriterium wird vorausgesetzt, dass das brennende Regal weitere Möbelstücke in Brand setzt und damit an der Brandentwicklung beteiligt wird. Auf diese Weise soll die Brandausbreitung in einem Raum simuliert werden. Die Geometrie wird frei gewählt, d.h. auf eine explizite Plandarstellung wird verzichtet.

3.2. Simulationsprofil

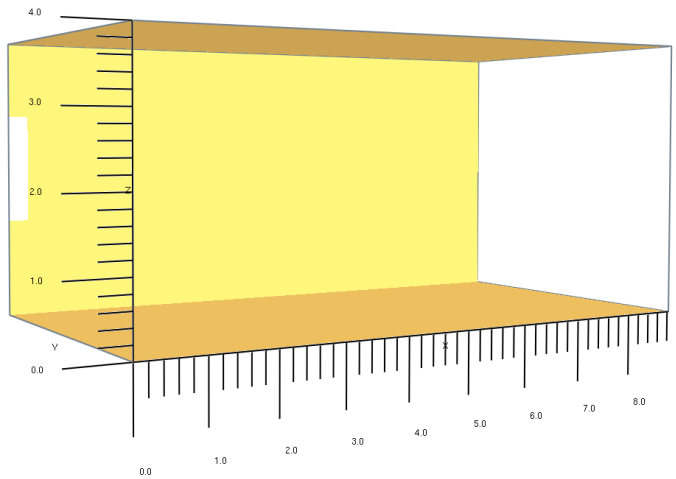

Abb. 3.1.: Ausgewähltes Simulationsprofil.

[1] Die Arbeit verfolgt die Zielsetzung, Informationsräume für die Simulation mit dem *Fire Dynamics Simulator* zu erschließen und in Beziehung zu setzen. Die Aussagekraft des simulierten Brandereignisses ist vorerst von untergeordneter Bedeutung.

Eine Simulation mit dem FDS setzt die Eingabe grundlegender Befehle voraus, deren Gesamtheit nachfolgend als Simulationsprofil bezeichnet wird. Es wird an den Anfang der Eingabedatei geschrieben und enthält Angaben zur Dauer, räumlichen Begrenzung und Genauigkeit der Simulation. Für das Minimalbeispiel wird ein Profil nach Quelltext 3.1 angelegt, mit dem der Simulationsumfang hinreichend genau ermöglicht wird.

Quelltext 3.1: Eingabebefehle für das Simulationsprofil

```
2  &HEAD FYI                  = 'Neuen Auftrag anlegen',
3        CHID                 = 'FDS-Minimalbeispiel_V1',
4        TITLE                = 'Zimmerbrand Variante 1'/
5  &TIME FYI                  = 'Simulationszeit in Sekunden',
6        T_End                =  60.0000/
7  &MESH FYI                  = 'Simulationsraum, 60 Maschen in x,y,z',
8        ID                   = 'Netz',
9        IJK                  =  60,60,60,
10       XB                   =  0.000,8.830,0.000,4.465,0.000,4.000,
11       RGB                  =  230,230,250/
12 &VENT FYI                  = 'Simulationsraum-Xmax oeffnen',
13       MB                   = 'XMAX',
14       SURF_ID              = 'OPEN'/
15 &VENT FYI                  = 'Simulationsraum-Ymax Oeffnung',
16       XB                   =  0.000,0.350,4.465,4.465,1.400,2.900,
17       SURF_ID              = 'OPEN'/
```

Über den Befehl &HEAD ist ein neuer Auftrag mit einer Bezeichnung (TITLE) sowie einem Dateinamen (CHID) anzulegen. Alle durch die Simulation generierten Dateien beziehen sich auf den genannten Dateinamen, was eine eindeutige Zuweisung im Installationspfad ermöglicht. Die Simulationszeit wird mit &TIME auf 60 Sekunden (T_END) eingestellt. Ein wichtiger Bestandteil des Simulationsprofils wird mit dem Befehl &MESH realisiert: Eine Netzgenerierung sowie die räumliche Begrenzung des Simulationsraums. Bezogen auf den Koordinatenursprung (0, 0, 0) wird ein 8.830 m langer (x), 4.465 m breiter (y) und 4.000 m hoher (z) Simulationsraum (XB) erzeugt, der sich in jeder Richtung durch 60 Maschen (IJK) unterteilt. Für das Programm entstehen dreidimensionale Berechnungsfelder, deren Anzahl ein Indikator für die zu erwartende Rechenzeit ist. Es können mehrere Simulationsräume vorgesehen werden und durch einzelne Identifikatoren (ID) identifiziert werden. Dies ermöglicht, Simulationen mit unterschiedlichen Detaillierungsgraden innerhalb eines Objekts durchzuführen, wobei deren Vernetzung zueinander prinzipiell erlaubt ist. In dem Zusammenhang zeigen McGrattan et al. Regelungen für die Netzkopplung von Simulationsräumen (2017, S. 37f), was im Rahmen der Arbeit unberücksichtigt bleibt. Damit sind grundlegende Simulationsbefehle beschrieben, die bereits das Kompilieren der Eingabedatei ermöglichen. Für das Beispiel werden darüber hinaus zwei optionale &VENT-Befehle verwendet, die es erlauben, die Begrenzungen des Simulationsraums zu öffnen. Die Wirkung einer unüberwindbaren Barriere geht verloren, was direkt die Brandausbreitung beeinflusst. Demnach wird die Raumbegrenzung vollständig (MB) an der größten x-Koordinate geöffnet (SURF_ID) und eine Öffnung im Bereich (XB) vorgesehen. Nach dem Kompilieren der Eingabedatei stellt sich ein Ergebnis nach Abbildung 3.1 ein.

3.3. Bauwerksmodellierung

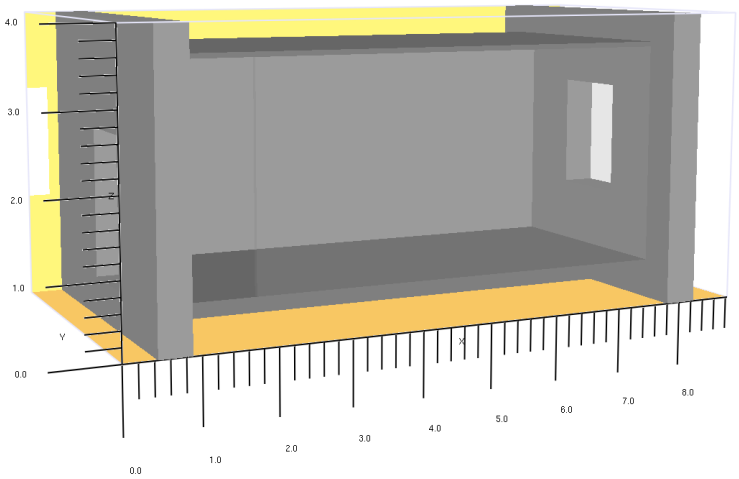

Abb. 3.2.: Generierung der Bauwerksgeometrie.

Mit einem Zimmerbrand wird eine unkomplizierte Bauwerksgeometrie gewählt, die sich aus Decken- und Wandbauteilen zusammensetzt. Fenster und Türen werden in einem ersten Schritt durch Öffnungen dargestellt, was die Abgeschlossenheit des Raums unterbrechen soll. Der Bauwerksgeometrie werden keine Materialparameter zugewiesen, um weitere Rechenkapazitäten zu ermöglichen. Die geometrischen Abmessungen werden frei gewählt und in Tabelle 3.1 aufgeführt.

Tabelle 3.1.: Abmessungen der Bauwerksgeometrie [m].

Bezeichnung	X1	X2	Y1	Y2	Z1	Z2
Kellerdecke	0.865	7.865	0.500	3.965	0.300	0.500
Geschossdecke	0.865	7.865	0.500	3.965	3.500	3.600
Außenwand	7.865	8.230	0.000	4.465	0.000	4.000
Trennwand	0.865	7.865	3.600	3.965	0.500	3.500
Treppenraumwand	0.500	0.865	0.000	4.465	0.000	4.000
Eingangstür	0.000	0.865	1.595	2.605	0.500	2.501
Zimmertür	1.000	1.885	0.500	0.600	0.500	2.501
Fenster	7.865	8.230	1.400	2.600	1.400	2.900

Der Quelltext 3.2 veranschaulicht Eingabebefehle, mit denen die Bauwerksgeometrie im Simulationsraum dargestellt wird. Dabei erzeugt die &OBST-Anweisung, bezogen auf den Koordinatenursprung, einzelne Quader mit den spezifischen Abmessungen (XB). Jeder generierten Geometrie werden eine Kennzeichnung (ID) sowie ein RGB-Farbraum (RGB) zugewiesen, wobei transparente Darstellungsweisen (TRANSPARENY) möglich sind. Mit dem &HOLE Befehl werden Öffnungen in den Quadern vorgesehen[2].

[2] Eine Öffnung ist nur im Bereich vorhandener Gegenstände sichtbar.

Quelltext 3.2: Eingabebefehle für die Bauwerksgeometrie

```
19 &OBST FYI                    = 'Modellierung Deckenbauteile',
20       ID                     = 'Kellerdecke',
21       XB                     =  0.865,7.865,0.500,3.965,0.300,0.500,
22       RGB                    =  105,105,105/
23 &OBST ID                     = 'Geschossdecke',
24       XB                     =  0.865,7.865,0.500,3.965,3.500,3.600,
25       RGB                    =  105,105,105/
26 /----------------------------------------------------------------------
27 &OBST FYI                    = 'Modellierung Wandbauteile',
28       ID                     = 'Aussenwand',
29       XB                     =  7.865,8.230,0.000,4.465,0.000,4.000,
30       RGB                    =  128,128,128/
31 &OBST ID                     = 'Trennwand',
32       XB                     =  0.865,7.865,3.600,3.965,0.500,3.500,
33       RGB                    =  128,128,128/
34 &OBST ID                     = 'Treppenraumwand',
35       XB                     =  0.500,0.865,0.000,4.465,0.000,4.000,
36       RGB                    =  128,128,128/
37 /----------------------------------------------------------------------
38 &HOLE FYI                    = 'Modellierung Oeffnungen',
39       ID                     = 'Eingangstuer',
40       XB                     =  0.000,0.865,1.595,2.605,0.500,2.501/
41 &HOLE ID                     = 'Zimmertuer',
42       XB                     =  1.000,1.885,0.500,0.600,0.500,2.501/
43 &HOLE ID                     = 'Fenster',
44       XB                     =  7.865,8.230,1.400,2.600,1.400,2.900/
```

3.4. Modellierung einer Brandentstehung

Für das Minimalbeispiel wird folgendes Szenario angenommen: Ein technischer Defekt in einem Gerät führt zu einer Initialzündung und leitet einen Entstehungsbrand ein. „In der Brandentstehungsphase findet innerhalb eines bestimmten Bereichs [...] eine Flammenausbreitung mit ansteigender Wärmeentwicklung statt. Die Wärmezunahme wird jedoch begrenzt durch Art und Menge der am Entstehungsbrand beteiligten Stoffe“ [MB18, S. 7 (4.1)]. Im FDS können Flächen mit zeitabhängigen Wärmefreisetzungsraten versehen und damit Entstehungsbrände modelliert werden.

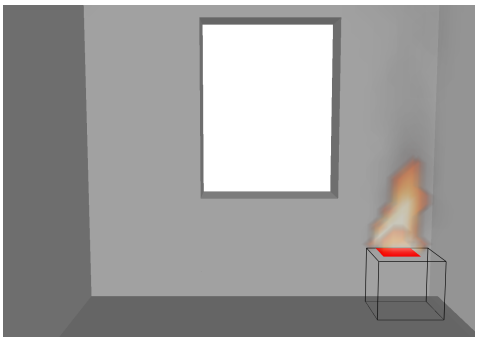

Abb. 3.3.: Modellierung einer Brandentstehung.

Die Abbildung 3.3 zeigt die Brandentstehung nach zehn Sekunden Simulationszeit. Durch die &OBST-Anweisung wird eine eigenständige Geometrie erzeugt, die lediglich durch die Außenkanten (OUTLINE) abgebildet ist. Auf diesem Quader wird eine 0.3m x 0.3m große Fläche gelegt, deren Eigenschaften über den Befehl &SURF definierbar sind[3]. Aufgrund der hierfür notwendigen Referenzierung ist eine eindeutige Kennzeichnung (ID) erforderlich. In Abbildung 3.3 wird diese Fläche rot dargestellt (RGB) und mit einer zeitabhängigen (RAMP_Q) Wärmefreisetzungsrate (HRRPUA) versehen. Der angedachte Funktionsverlauf wird durch einzelne &RAMP-Anweisungen beschrieben, die jeweils zu einer Simulationszeit (T) einen prozentualen Funktionswert (F) wiedergeben. Eine eindeutige Zuweisung (ID) wird auch hier zwingend vorausgesetzt. Der Funktionswert von 1.0 bezieht sich dabei auf die angegebene Wärmefreisetzungsrate (HRRPUA), wobei das Ergebnis des vorgegebenen Verlaufs der Abbildung 3.4 entnommen werden kann.

Quelltext 3.3: Eingabebefehle für die Modellierung einer Brandentstehung

```
48 &OBST FYI                  = 'Modellierung Entstehungsbrand',
49       ID                   = 'E-Geraet',
50       XB                   =  7.200,7.700,0.700,1.200,0.500,1.000,
51       RGB                  =  0,0,0,
52       OUTLINE              = .TRUE./
53 &VENT FYI                  = 'Bereich auf Oberflaeche',
54       XB                   =  7.300,7.600,0.800,1.100,1.000,1.000,
55       SURF_ID              = 'Defekt'/
56 &SURF FYI                  = 'Waermefreisetzungsrate',
57       ID                   = 'Defekt',
58       RGB                  =  255,0,0,
59       HRRPUA               =  1000.000,
60       RAMP_Q               = 'Funktionsverlauf'/
61 &RAMP ID                   = 'Funktionsverlauf', T= 5.000, F=0.000/
62 &RAMP ID                   = 'Funktionsverlauf', T=10.000, F=1.000/
63 &RAMP ID                   = 'Funktionsverlauf', T=30.000, F=1.000/
64 &RAMP ID                   = 'Funktionsverlauf', T=31.000, F=0.000/
65 &REAC FYI                  = 'Verbrennungsreaktion',
66       FUEL                 = 'PROPANE',
67       SOOT_YIELD           =  0.010/
```

Quelltext 3.4: Eingabebefehle hinsichtlich geometrischer Bedingungen

```
   &OBST XB                   = 7.200,7.700,0.700,1.200,0.500,1.000,
   &VENT XB                   = 7.300,7.600,0.800,1.100,1.000,1.000,
```

[3] Geometrische Bedingungen sind zu beachten, vgl. Quelltext 3.4.

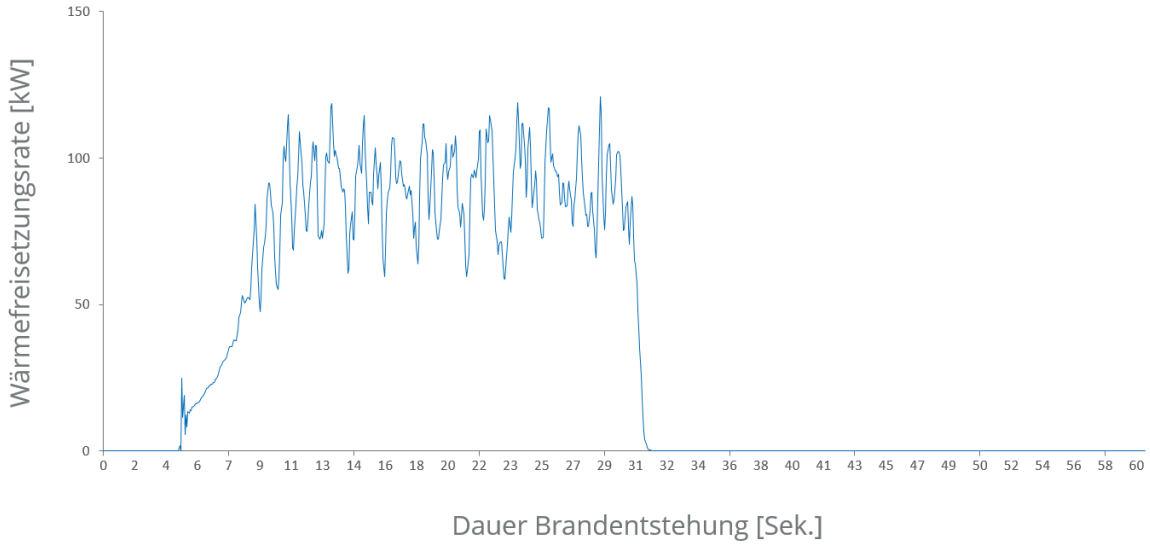

Abb. 3.4.: Zeitlicher Verlauf der Wärmefreisetzungsrate.

Die Grafik kann mithilfe einer, durch die Simulation generierte, CSV-Datei erstellt werden. Das Dateiformat erlaubt die Speicherung und den Austausch von Daten und stellt im vorliegenden Fall die Funktionswerte zur Wärmefreisetzungsrate bereit. Damit kann eine Plausibilitätskontrolle vorgenommen werden, was nachfolgend Formel 3.1 verdeutlicht.

$$HRR = HRRPUA \cdot A = 1000\ kW/m^2 \cdot (0.300 \cdot 0.300)\ m^2 = 90\ kW \tag{3.1}$$

Aus der Grafik geht der zeitliche Verlauf der gewählten Wärmefreisetzungsrate für den Entstehungsbrand hervor. Auf der Abszisse ist die Simulationszeit bis zu 60 Sekunden angeben, die y-Achse zeigt die zugehörige Wärmefreisetzung in Kilowatt. Fünf Sekunden nach Beginn der Simulation erhöht sich die Wärmefreisetzung, bis nach etwa zehn Sekunden ein konstanter Verlauf festzustellen ist. Der dabei erreichte Wert entspricht dem in 3.1 ermittelten Maximalwert, wobei die Streuung auf die programmspezifischen Berechnungsalgorithmen zurückgeführt werden kann. Bei 31 Sekunden fällt die Kurve, sodass nach etwa zwei Sekunden keine Wärmefreisetzung bis zum Ende der Simulation aufgezeichnet wird. Insgesamt entspricht dies den eingestellten Funktionsparametern im Quelltext 3.3, womit diese vom Simulationsprogramm umgesetzt und plausibel werden.

Abschließend ist darauf hinzuweisen, dass für die Modellierung einer Brandentstehung eine Verbrennungsreaktion definiert werden muss. Hierfür wird im Quelltext 3.3 der Befehl &REAC verwendet und mit einem Brennstoff (FUEL) sowie einer Rußausbeute (SOOT_Yield) spezifiziert. Für das Minimalbeispiel werden die Werte frei gewählt, um in der Simulationszeit von 60 Sekunden ein aussagekräftiges Brandereignis darzustellen.

3.5. Modellierung brennbarer Gegenstände

In diesem Abschnitt wird auf die Grundidee des eingeführten Minimalbeispiels Bezug genommen: Die selbstständige Entzündung brennbarer Materialien. Hierzu werden im Simulationsraum einzelne Gegenstände vorgesehen, die durch definierte Materialeigenschaften das Brandverhalten einer Möblierung aufweisen[4]. Aufgrund der im Abschnitt 3.4 gezeigten Brandentstehung sollen sich die Gegenstände selbstständig entzünden und an der Brandausbreitung beteiligen.

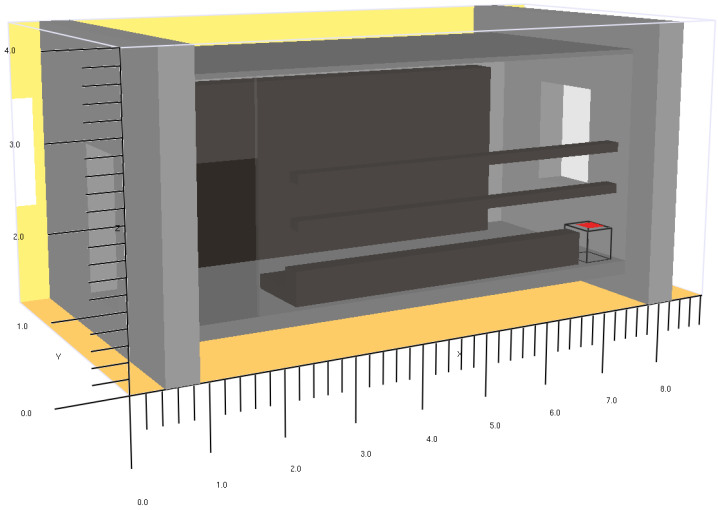

Abb. 3.5.: Darstellung brennbarer Gegenstände.

Abbildung 3.5 zeigt die räumliche Anordnung der Möblierungen, die sich aus einer Schrankwand, einem Sofa sowie Regalen zusammensetzen. Durch den Befehl &OBST werden Quader (XB) erzeugt und deren Oberflächen konkrete Eigenschaften durch Identifikatoren (SURF_ID) zugewiesen. An dieser Stelle wird auf die Möglichkeit hingewiesen, die sechs Oberflächen eines Quaders einzeln definieren zu können. Hierauf wird im Rahmen dieses Beispiels jedoch verzichtet, sodass die dargestellten Körperflächen umlaufend die gleichen Eigenschaften aufweisen. Neben einer farblichen Darstellung (RGB) werden den Oberflächen des Quaders mithilfe der &SURF-Anweisung eine Dicke (THICKNESS) sowie konkrete Materialeigenschaften (MATL_ID) vorgegeben. Brennen Gegenstände vollständig nieder, werden diese im Detaillierungsgrad des eingestellten Simulationsprofils (&MESH) ausgeblendet (BURN_AWAY). Materialeigenschaften werden mit dem Befehl &MATL erzeugt, wobei die für das Beispiel angewandten Parameter der Tabelle 3.2 zu entnehmen sind. Die beschriebenen Simulationsbefehle werden im Quelltext 3.5 exemplarisch für die beiden Regale verdeutlicht. Die vollständige Auflistung aller modellierten brennbaren Gegenstände sind dem Anlagenteil A.1 zu entnehmen.

[4] Materialparameter werden frei gewählt.

Quelltext 3.5: Eingabebefehle für die Modellierung brennbarer Gegenstände

```
73 &OBST ID                    = 'Regal2',
74       XB                    =  2.500,7.700,0.600,0.800,2.100,2.250,
75       SURF_ID               = 'Regaleigenschaften'/
76 &SURF FYI                   = 'Eigenschaften',
77       ID                    = 'Regaleigenschaften',
78       RGB                   =  41,36,33,
79       THICKNESS             =  0.050,
80       MATL_ID               = 'Holz',
81       BURN_AWAY             = .TRUE./
82 &MATL FYI                   = 'Material',
83       ID                    = 'Holz',
84       SPEC_ID               = 'PROPANE',
85       DENSITY               =    40.000,
86       SPECIFIC_HEAT         =     6.500,
87       CONDUCTIVITY          =     0.050,
88       HEAT_OF_REACTION      =    10.000,
89       HEAT_OF_COMBUSTION    = 30000.000,
90       REFERENCE_TEMPERATURE =   120.000,
91       N_REACTIONS           =     1.000,
92       NU_SPEC               =     1.000/
```

Tabelle 3.2.: Auflistung gewählter Materialparameter.

Parameter	Kurzbeschreibung	Einheit
SPEC_ID	Zuordnung Reaktionsmechanismus	$'PROPANE'$
DENSITY	Thermische Eigenschaft Dichte	kg/m^3
SPECIFIC_HEAT	Thermische Eigenschaft spezifische Wärme	$kJ/(kg \cdot K$
CONDUCTIVITY	Thermische Eigenschaft Leitfähigkeit	$W/(m \cdot K$
HEAT_OF_REACTION	Materialeigenschaft Hitzereaktion	kJ/kg
HEAT_OF_COMBUSTION	Materialeigenschaft Verbrennungswärme	kJ/kg
REFERENCE_TEMPERATURE	Temperatur bei Abnahme der Masse	$°C$
N_REACTIONS	Gesamtanzahl der Reaktionsprodukte	$-$
NU_SPEC	Relative Mengenangabe Reaktionsausbeute	kg/kg

3.6. Auswertung und Simulation

Das Minimalbeispiel sieht eine Temperaturauswertung im Brandraum vor. Mit dem *Fire Dynamics Simulator* können hierfür Ebenen im Simulationsraum angelegt werden, was exemplarisch im Quelltext 3.6 gezeigt ist. Die &SELF-Anweisung enthält dabei den Parameter QUANTITY, der die Temperaturauswertung vorschreibt (TEMPERATURE). Über PBX wird an der Stelle x=7.500m eine Ebene in Richtung der y- und z-Achse aufgestellt. Die Ebene unterteilt sich, aufgrund des im Abschnitt 3.2 eingestellten Simulationsprofils, in 3600 Segmente[5].

[5] Die Segmente entstehen durch die Netzeinteilung (Parameter IJK = 60,60,60).

Quelltext 3.6: Eingabebefehle für die Temperaturauswertung

```
153  &SLCF FYI                    = 'Temperaturscheibe x-Ebene',
154        QUANTITY               = 'TEMPERATURE',
155        PBX                    =  7.500/
```

Nach dem Kompilieren der Eingabedatei sind die FDS-Berechnungen mit dem Programm *Smokeview* visualisierbar (vgl. Abschnitt 2.2.3). Die Abbildung 3.6 zeigt einen Bildausschnitt nach einer Simulationszeit von 40 Sekunden, mit dem das Zusammenspiel aller FDS-Anweisungen veranschaulicht wird[6]. Die Brandentstehung ist abgeschlossen, sodass sich der Brand im Raum ausschließlich über brennbare Gegenstände ausbreitet. Die Grundidee des Minimalbeispiels wird umgesetzt.

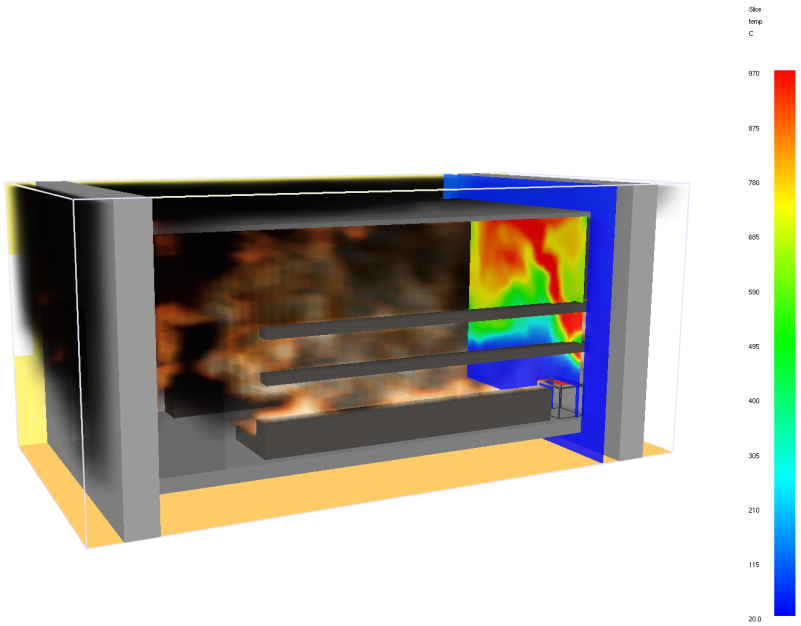

Abb. 3.6.: Auswertung und Simulation für das FDS-Minimalbeispiel.

3.7. Zusammenfassung der Erkenntnisse

Mit der Einführung des Beispiels werden essenzielle Informationsdaten für FDS-Simulationen erschlossen. Einzelne Anweisungen und deren Zusammenhänge werden berücksichtigt, was für den weiteren Verlauf der Arbeit einen ganzheitlichen Blick auf die Brandsimulation bietet. Durch die Vernetzung einzelner FDS-Eingabebefehle kann auf ein Datenschema geschlussfolgert werden, dass erst mit dem erfolgreichen Kompiliervorgang bestätigt wird. Damit liegt die Verantwortung für die richtige Zusammensetzung einzelner Anweisungen beim jeweiligen Anwender. Für die vorliegende Arbeit heißt das: Aus welchen Fachbereichen können Informationen gewonnen werden, die gezielt verknüpft, Simulationen für den *Fire Dynamics Simulator* ermöglichen?

[6] Die Simulationszeit ist darstellungsbedingt ausgeblendet; für weitere Bildausschnitte siehe Anlagenteil.

4. Informationsraum der Industry Foundation Classes

Gute Informationen sind schwer zu bekommen. Noch schwerer ist es, mit ihnen etwas anzufangen.

(Sir Arthur Conan Doyle)

4.1. Gegenstand der Untersuchung

Das Ziel der Untersuchung ist es herauszufinden, welche FDS-Eingabebefehle aus dem Informationsraum der *Industry Foundation Classes* generiert werden können. Die Untersuchung beschränkt sich auf Gebäudeelemente, die Bauwerke, brennbare Gegenstände und Ursachen für Entstehungsbrände abbilden. Schwerpunktmäßig werden deren Abmessungen und Platzierungen analysiert, um mithilfe von Programmroutinen ein Abbild der Elemente im Simulationsprogramm zu ermöglichen. In diesem Zusammenhang wird im Abschnitt 2.2.2 auf die Geometriedarstellung im *Fire Dynamics Simulator* eingegangen. Schräge Geometrieverläufe können demzufolge lediglich approximiert werden, was aufgrund des damit verbundenen Mehraufwands ausgeschlossen wird. Weiterhin werden Möglichkeiten für die Eingabe von spezifischen Eigenschaften der Gebäudeelemente analysiert, die als Simulationsparameter dienen können.

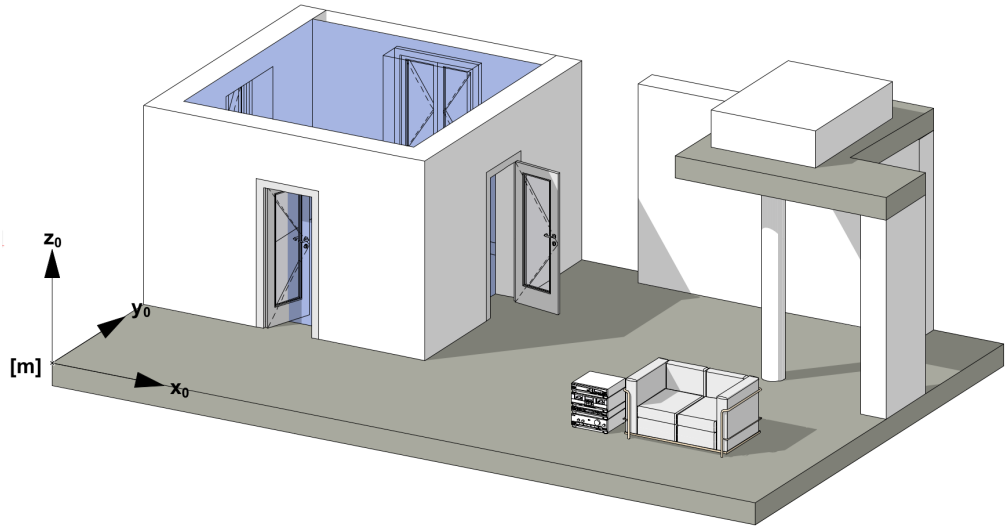

Abb. 4.1.: Auswahl an Gebäudeelementen zur Untersuchung der IFC.

Die Abbildung 4.1 stellt die für diese Arbeit untersuchten Elemente auf insgesamt zwei Höhenniveaus dar. Die Auswahl orientiert sich an Bauteilen, wie zum Beispiel Decken, Wände und Einrichtungsgegenstände. Eine quaderförmige Standardform dient der Repräsentation eines Gegenstands, dessen Eigenschaften die Modellierung von Brandlasten ermöglichen soll. Ein Verbund von Wänden erlaubt die Darstellung einzelner Räume, was in der Grafik bläulich hervorgehoben wird und ein Bestandteil der Untersuchung ist.

Die einzelnen Gebäudeelemente werden mit einer CAD/BIM Software [Gra] modelliert, wobei die Elemente durch Bezeichnungen und Eigenschaften klassifiziert werden. Im Anschluss erfolgt die Generierung einer IFC-Austauschdatei, deren Inhalte mit der von dem *buildingSMART e.V.* bereitgestellten Dokumentation [B118] analysiert werden können.

Damit wird ein anwendungsorientiertes Vorgehen gewählt, das eine unmittelbare Übertragung der Informationsdaten in konkrete FDS-Eingabebefehle ermöglicht. Im Gegensatz zu einer rein theoretischen Betrachtung der IFC-Datenstruktur, soll die Entwicklung von Filter- und Transformationsroutinen vereinfacht werden.

4.2. Analyse der IFC-Austauschdatei

Nach der Modellierung der in Abbildung 4.1 dargestellten Elemente ermöglicht die CAD/BIM Software die Generierung einer IFC-Austauschdatei. Durch einen programminternen IFC-Übersetzer wird das Datenschema IFC4 eingestellt, sodass die Austauschdatei auf der neusten Version der *Industry Foundation Classes* basiert[1]. Neben der eigentlichen Funktion, diese Datei durch weitere CAD/BIM Softwareanwendungen einzulesen und zu bearbeiten, kann diese mit herkömmlichen Texteditoren geöffnet werden. Dabei wird zunächst festgestellt, dass für die getroffene Auswahl an Gebäudeelementen insgesamt 22.550 IFC-Textzeilen erzeugt werden. Damit soll an dieser Stelle der Arbeit auf den grundlegenden Informationsumfang hingewiesen werden, der sich bereits bei einer geringen Anzahl von Elementen einstellt. Mithilfe von Suchfunktionen werden gezielt die Elementbezeichnungen gefiltert, was unmittelbar zu den zugehörigen IFC-Klassen führt. Exemplarisch wird dieses Vorgehen am Quelltext 4.1 für ein erzeugtes Gebäudeelement verdeutlicht, wobei die gesuchte Bezeichnung grün und die zugehörige IFC-Klasse blau hervorgehoben werden[2].

Quelltext 4.1: Darstellung eines Gebäudeelements durch die IFC

```
#471 = IFCSLAB('0iFuOXwidzIhaetDff_ZCU',#12,'Decke-001',$,$,#143,#465,
       '2C3F8621-EAC9-FD4AB928-DCDA69FA331E',.FLOOR.);
```

Die modellierte Decke wird durch die *Industry Foundation Classes* als IfcSlab mit der Entitätennummer #471 dargestellt. In den Klammern dieser Entität sind einzelne Attribute sowie Referenzen zu weiteren Entitäten hinterlegt. Beispielsweise wird der Entität eine eindeutige *Globally Unique Identifier* ('0iFuOXwidzIhaetDff_ZCU') als Attribut zugeordnet. Die Referenzierungen #143 und #465 geben Aufschluss über die örtliche Lage und die geometrischen Abmessungen der erzeugten Decke. Diese beiden Entitätennummern führen zu weiteren Referenzierungen innerhalb der IFC-Austauschdatei, was durch den Quelltext 4.2 veranschaulicht wird.

[1] Die Version ist von der Implementierungsmöglichkeit in einer Software abhängig (IFC2x3/4).
[2] Im Anlagenteil A.2 sind die IFC-Textzeilen für alle erzeugten Gebäudeelemente einsehbar.

Quelltext 4.2: Örtliche Lage und geometrische Abmessungen der IfcSlab

```
#143 = IFCLOCALPLACEMENT(#128,#142);
#142 = IFCAXIS2PLACEMENT3D(#140,#138,#136);
#140 = IFCCARTESIANPOINT((0.,0.,0.));
#138 = IFCDIRECTION((0.,0.,1.));
#136 = IFCDIRECTION((1.,0.,0.));

#465 = IFCPRODUCTDEFINITIONSHAPE($,$,(#437,#462));
#462 = IFCSHAPEREPRESENTATION(#457,'Box','BoundingBox',(#461));
#461 = IFCBOUNDINGBOX(#459,10.,6.33229828605,0.3);
#459 = IFCCARTESIANPOINT((0.,0.,-0.3)
```

Die Lage der Decke wird in den IFC als *IfcLocalPlacement* mit der Entitätennummer #143 initiiert. Über eine weitere Referenzierung werden konkrete Zahlenwerte zum lokalen Koordinatensystem der Decke übermittelt. Im vorliegenden Fall liegt der Koordinatenursprung bei $0.000,\ 0.000,\ 0.000\ [m]$ bezogen auf das globale Koordinatensystem x_0, y_0, z_0 (vgl. Abbildung 4.1). Die Achsrichtungen entsprechen den Richtungen des globalen Koordinatensystems, was in xy-Ebene durch die Entität #138 und in der z-Richtung durch die Entität #136 ausgedrückt wird. Analog können die Abmessungen des Deckenelements ermittelt werden. Die *IfcProductDefinitionsShape* stellt hierfür mit der Entitätennummer #462 eine *BoundingBox* mit den Abmessungen $10.000,\ 6.332,\ 0.300\ [m]$ zur Verfügung. Mit den Angaben wird eine quaderförmige Geometrie bezogen auf das lokale Koordinatensystem aufgespannt. Ergänzend wird ein Versatz von $0.300m$ entgegen (-) der z-Richtung des lokalen Koordinatensystems vorgenommen.

Die Entitäten #143 und #465 stellen darüber hinaus weitere Referenzierungen (#128, #437), die eine geometrische Beschreibung der Decke vorsehen, bereit. Zum Beispiel werden relative Platzierungen im Verhältnis zu anderen Gebäudeelementen angegeben und Konturen über konkrete Koordinatenpunkte beschrieben. Die vorliegende Arbeit fokussiert sich in einem ersten Schritt auf den zuvor beschriebenen Ansatz, da aus lokalen Koordinaten und quaderförmigen Abmessungen einzelne FDS-Eingabebefehle unkompliziert gebildet werden können. Beispielhaft wird im Quelltext 4.3 das Zusammenspiel der Informationsdaten für das vorliegende Deckenelement gezeigt.

Quelltext 4.3: Generierung der FDS-Eingabebefehle aus der IfcSlab

```
#136 = IFCDIRECTION((1.,0.,0.));          // Achsausrichtung in xy-Ebene
#138 = IFCDIRECTION((0.,0.,1.));          // Achsausrichtung in z-Richtung
#140 = IFCCARTESIANPOINT((0.,0.,0.));     // Koordinatenursprung
#459 = IFCCARTESIANPOINT((0.,0.,-0.3));   // lokaler Versatz
#461 = IFCBOUNDINGBOX(#459,10.,6.33229828605,0.3); // Abmessungen

&OBST XB = X1,X2,Y1,Y2,Z1,Z2
&OBST XB = (0.+0.),(0.+0.+10.),(0.+0.),(0.+0.+6.332),(0.-0.3),(0.-0.3+0.3)
&OBST XB = 0.000,10.000,0.000,6.332,-0.300,0.000
```

Die &OBST-Anweisung enthält Koordinatenpunkte bezogen auf das globale Koordinatensystem nach Abbildung 4.1, wobei dieses mit dem Koordinatensystem des *Fire Dynamics*

Simulator gleichgesetzt werden kann. Die Vorzeichen in der Rechenvorschrift werden von den Achsrichtungen des lokalen Koordinatensystems bestimmt und erweisen sich für das Deckenelement aufgrund gleicher Achsausrichtungen ($x_1 = x_0$; $y_1 = y_0$; $z_1 = z_0$) als sehr einfach. Die Abbildung 4.2 zeigt die Auswirkung des FDS-Eingabebefehls. Die Grafik verdeutlicht damit die erfolgreiche Transformation der, aus der *Industry Foundation Classes* bereitgestellten, Informationsdaten in das Simulationsprogramm.

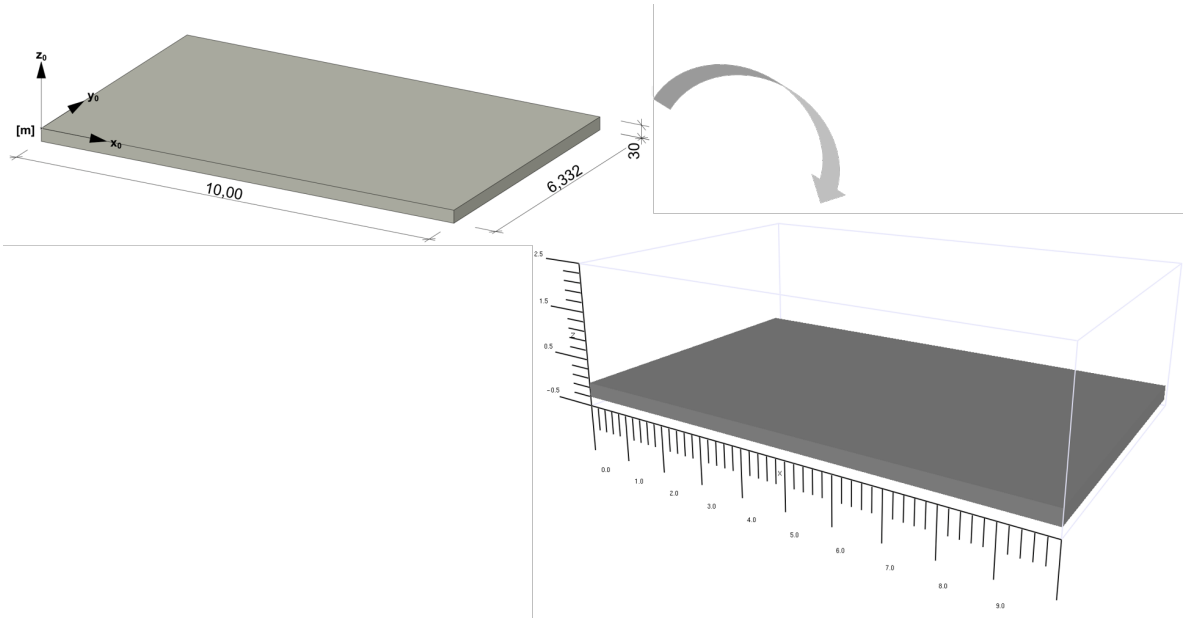

Abb. 4.2.: Transformation der Informationsdaten von der *IfcSlab* in den FDS.

Insgesamt wird damit ein Vorgehen beschrieben, das ein geometrisches Abbild eines einzigen Gebäudeelements durch eine globale ID und der IFC-Datenstruktur ermöglicht. Wird der anfangs beschriebene Suchvorgang wiederholt, folgen weitere IFC-Klassen für die einzelnen in Abbildung 4.1 eingeführten Gebäudeelemente. Die Struktur, der in den Klammern hinterlegten Entitäten und Referenzen, erweist sich als überwiegend gleichbleibend auf. Dies ist auf den hierarchischen Aufbau der *Industry Foundation Classes* sowie der daraus resultierenden Vererbungsbeziehung zurückzuführen. In diesem Zusammenhang zeigt Abbildung 4.3 exemplarisch die Vererbungsstruktur verschiedener Gebäudeelemente. Damit erben z.B. Decken und Wände die Entitäten, bzw. die Attribute, der *IfcBuildingElement*. Alle in dieser Arbeit untersuchten Gebäudeelemente weisen in ihren Vererbungshierarchien als Entität die *IfcProduct* auf und referenzieren somit jeweils auf Objekte aus *IfcObjectPlacement* und *IfcProductRepresentation*. Die beiden IFC-Klassen spiegeln dabei die Abmessungen und Platzierungen der einzelnen Gebäudeelemente wieder, was exemplarisch im Quelltext 4.2 mit den Entitätennummern #143 und #465 gezeigt wird. Bedingt durch die IFC-Datenstruktur erfolgen diese Referenzierungen immer an der gleichen Stelle der jeweiligen IFC-Klasse des Gebäudeelements.

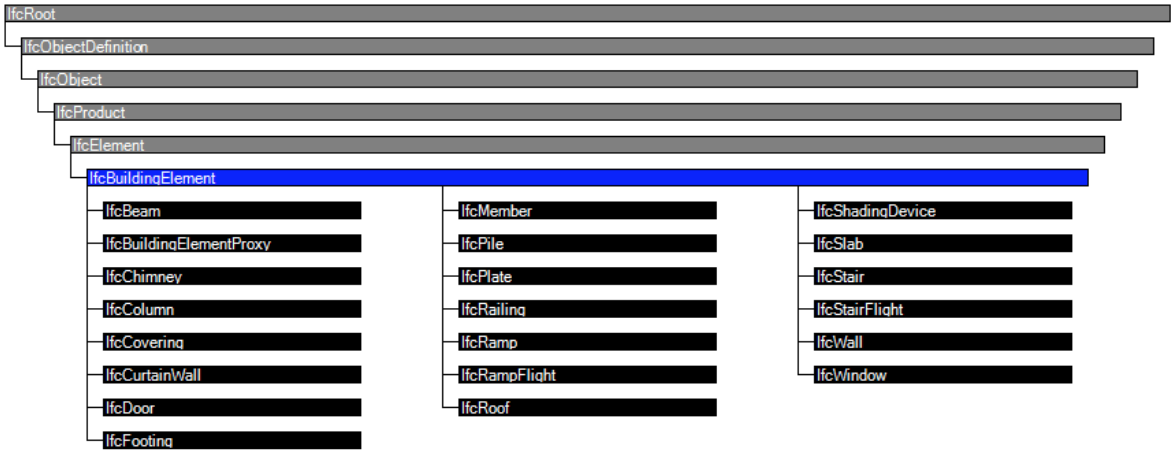

Abb. 4.3.: Vererbung der Entitäten der *IfcBuildingElement* [B218].

Für die Arbeit bietet diese Eigenschaft Vorteile bei der Erstellung von Filterroutinen. Die geometrischen Abmessungen der einzelnen Elemente können durch die Vererbungsbeziehungen der IFC-Datenstruktur in einer gemeinsamen Filterroutine abgefragt werden. Für die Transformationen der Informationsdaten in FDS-Eingabebefehle werden jedoch weitere Analysen notwendig, auf welche in den folgenden Abschnitten eingegangen wird. Diese orientieren sich an dem hier beschriebenen Vorgehen und zeigen dabei Möglichkeiten auf, konkrete FDS-Eingabebefehle aus den gewonnenen Informationsdaten zu generieren.

4.2.1. Darstellung der Räume durch die IfcSpace

Mit der Betrachtung des modellierten Raums werden zwei Ansätze verfolgt: Die räumliche Begrenzung eines Simulationsprofils und die Festlegung einer Wärmefreisetzungsrate in einer Nutzungseinheit (NE) durch eine eigenschaftsbasierte Fläche (vgl. Abschnitt 3.2 und 3.4). Damit soll zum einen die Möglichkeit gegeben werden, eine Simulation auf einen einzelnen Raum oder auf den Zusammenschluss mehrerer Räume zu begrenzen[3]. Zum anderen erlaubt dieser Ansatz, die Beurteilung eines Brandereignisses auf Basis normativer Wärmefreisetzungsraten[4] ohne dabei brennbare Gegenstände in einer NE klassifizieren zu müssen. Die *Industry Foundation Classes* stellen den modellierten Raum durch die *IfcSpace* mit Entitätennummer #40038 dar. Mit den gewonnenen Informationsdaten können Bestandteile zweier grundlegender FDS-Eingabebefehle generiert werden. Die &MESH-Anweisung legt dabei die räumliche Begrenzung (XB) in einem Simulationsprofil fest und der &VENT-Befehl beschreibt die geometrischen Abmessungen (XB) einer Fläche. Die Möglichkeit der Generierung dieser FDS-Eingabebefehle verdeutlicht die Abbildung 4.4 zusammen mit dem Quelltext 4.4.

[3] Beim Zusammenschluss einzelner Räume sind maximale Abmessungen zu bestimmen (Programmroutine).
[4] Werte der Wärmefreisetzungsraten z.B. nach DIN EN 1991-1-2/NA:2010-12 4.10, Tabelle BB.2.

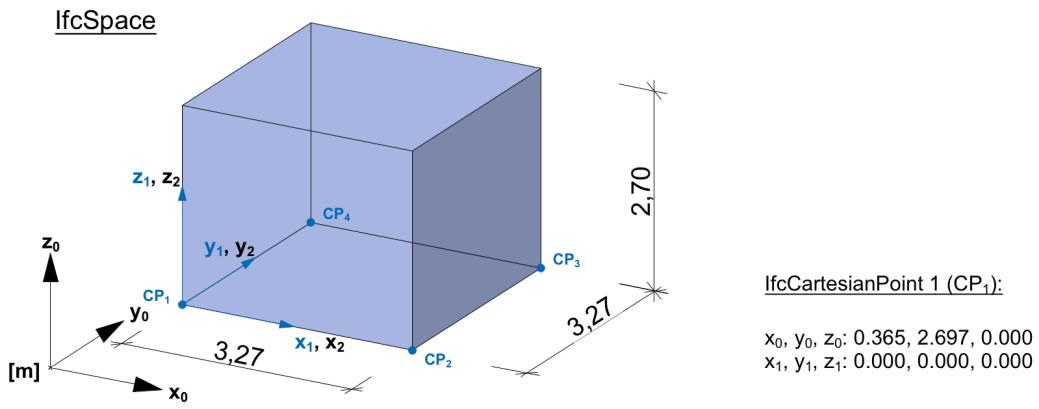

Abb. 4.4.: Geometrische Angaben aus der *IfcSpace*.

Quelltext 4.4: Generierung der FDS-Eingabebefehle aus der IfcSpace

```
#40038 = IFCSPACE('1tV0ztwINiHgG7hrZD7239',#12,'R-002',$,$,#39948,#40033,
         'B\X2\00FC\X0\ro',.ELEMENT.,$,$);
#39948 = IFCLOCALPLACEMENT(#128,#39947);
#39947 = IFCAXIS2PLACEMENT3D(#39945,#39943,#39941);
#39945 = IFCCARTESIANPOINT((0.365,2.69729828605,0.));      // Koordinatenursprung
#39943 = IFCDIRECTION((0.,0.,1.));                        // Achsausrichtung in z-Richtung
#39941 = IFCDIRECTION((1.,0.,0.));                        // Achsausrichtung in xy-Ebene
#40033 = IFCPRODUCTDEFINITIONSHAPE($,$,(#40007,#40015,#40030));
#40015 = IFCSHAPEREPRESENTATION(#457,'Box','BoundingBox',(#40014));
#40014 = IFCBOUNDINGBOX(#40012,3.27,3.27,2.7);            // Abmessungen
#40012 = IFCCARTESIANPOINT((0.,0.,0.));                   // lokaler Versatz

&MESH XB = X1,X2,Y1,Y2,Z1,Z2
&MESH XB = (0.365+0.),(0.365+0.+3.27),
           (2.697+0.),(2.697+0.+3.27),
           (0.+0.),(0.+0.+2.7)
&MESH XB = 0.365,3.635,2.697,5.967,0.000,2.700
&MESH XB = 0.365,3.635,2.697,5.967,-0.300,2.700          // Modifizierung Z1

&VENT XB = X1,X2,Y1,Y2,Z12,Z12
&VENT XB = (0.365+0.),(0.365+0.+3.27),
           (2.697+0.),(2.697+0.+3.27),
           (0.+0.),(0.+0.)
&VENT XB = 0.365,3.635,2.697,5.967,0.000,0.000

#40030 = IFCSHAPEREPRESENTATION(#40017,'FootPrint','GeometricCurveSet',(#40028));
#40028 = IFCGEOMETRICCURVESET((#40026));
#40026 = IFCPOLYLINE((#40018,#40020,#40022,#40024,#40018));
#40018 = IFCCARTESIANPOINT((0.,0.));                      // CP1
#40020 = IFCCARTESIANPOINT((3.27,0.));                    // CP2
#40022 = IFCCARTESIANPOINT((3.27,3.27));                  // CP3
#40024 = IFCCARTESIANPOINT((0.,3.27));                    // CP4
#40018 = IFCCARTESIANPOINT((0.,0.));                      // CP1

&VENT XB = X1,X2,Y1,Y2,Z12,Z12
&VENT XB = (0.365+0.+0.),(0.365+0.+3.27),
           (2.697+0.+0.),(2.697+0.+3.27),
           (0.+0.),(0.+0.)
&VENT XB = 0.365,3.635,2.697,5.967,0.000,0.000
```

In Abbildung 4.4 wird ein zusätzliches Koordinatensystem (x_2, y_2, z_2) eingeführt, das den Versatz zum lokalen Koordinatensystem beschreibt. Quelltext 4.4 zeigt die Möglichkeit, beide FDS-Eingabebefehle mithilfe der *IfcBoundingBox* zu formulieren. Alternativ kann die &VENT-Anweisung durch die Eckpunkte (CP_1, CP_2, CP_3 und CP_4) beschrieben werden. Aufbauend auf der im Abschnitt 4.2 gezeigten Transformation, wird mit Abbildung 4.6 die Auswirkung der beiden Eingabebefehle im *Fire Dynamics Simulator* verdeutlicht. Die erzeugte &MESH-Anweisung schneidet dabei umliegende Bauteile ab, sodass dadurch die zuvor generierte Decke ausgeblendet wird. Dies wird durch die Modifizierung der Z_1-Angabe verhindert (vgl. Quelltext 4.4).

Mit diesem Quader wird die einfachste Form eines Raums dargestellt. Demnach stellt sich die Frage, mit welchen Informationsdaten nicht-rechteckige Geometrieverläufe (z.B. L-förmig) realisiert werden. Weiterhin ist zu hinterfragen, wie sich Elemente auf anderen Höhenniveaus darstellen. Der Abschnitt 4.2.2 widmet sich dieser Thematik und beschreibt in dem Zusammenhang die entstehende Problematik[5].

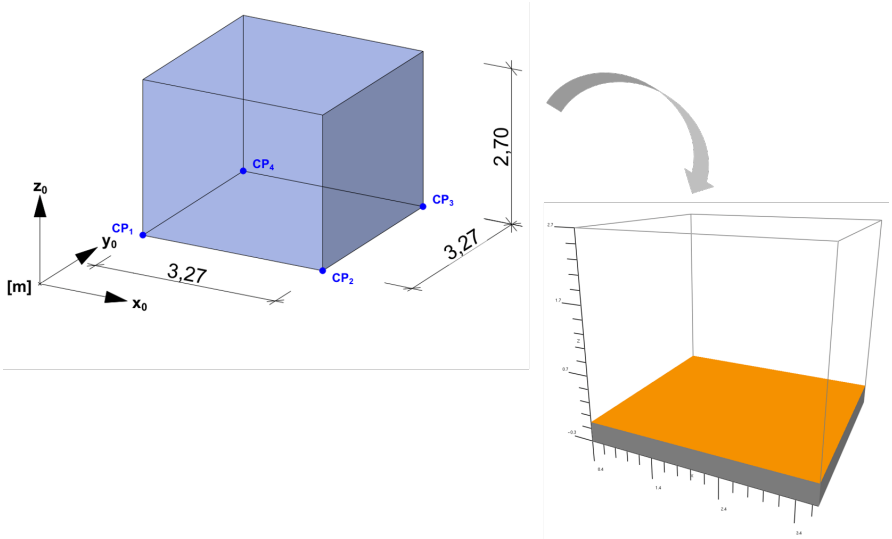

Abb. 4.5.: Transformation der Informationsdaten von der *IfcSpace* in den FDS.

4.2.2. Höhenversetzte L-förmige Geometrie am Beispiel der IfcSlab

In Abbildung 4.1 wird ein L-förmiges Deckenelement auf einem zweiten Höhenniveau eingeführt, wobei sich die Abmessungen aus den Entitätennummern #40260 und #40254 ergeben (vgl. Quelltext 4.5). Informationsdaten werden dabei über eine *BoundingBox* und einen *SweptSolid* gewonnen, wobei sich das Vorgehen an den vorangegangen Abschnitten orientiert und nicht weiter erörtert wird.

[5] Die Untersuchung ist, wegen der im Abschnitt 4.2 beschriebenen Vererbungsstruktur, an einem weiteren Gebäudeelement möglich. Das resultierende Ergebnis findet sinngemäß Anwendung.

Quelltext 4.5: Geometrische Angaben aus der IfcSlab

```
#40260 = IFCSHAPEREPRESENTATION(#457,'Box','BoundingBox',(#40259));
#40254 = IFCSHAPEREPRESENTATION(#145,'Body','SweptSolid',(#40253));
```

Als Visualisierung dient die Abbildung 4.6, welche einen aufgespannten Deckenraum durch die *BoundingBox* und sechs Koordinatenpunkte (CP) zeigt. Dabei ist ein signifikanter Versatz zwischen dem lokalen Koordinatensystem ($x_1,\ y_1,\ z_1$) und dem Bezugspunkt der *BoundingBox* ($x_2,\ y_2,\ z_2$) festzustellen. Die Aussparung wird durch deren Abmessungen $3.000,\ 3.000,\ 0.300\ [m]$ vernachlässigt und wird lediglich über die Koordinatenpunkte CP_1 - CP_16 beschrieben. Die direkte Umwandlung in eine XB-Anweisung ist ausgeschlossen, da entweder die Aussparung unberücksichtigt bleibt oder konkrete Abgaben zu $X_1,\ X_2,\ Y_1,\ Y_2,\ Z_1$, und Z_2 fehlen. Hinsichtlich der Generierung eines FDS-Eingabebefehls bestehen zwei Möglichkeiten: Die Zerteilung des Deckenelements in mehrere Einzelteile oder das Erzeugen einer Aussparung durch eine zusätzliche &HOLE-Anweisung[6]. Beide Varianten erfordern Überlegungen hinsichtlich geometrischer Transformationsroutinen von Polygonen, wovon sich die vorliegende Arbeit an dieser Stelle abgrenzt.

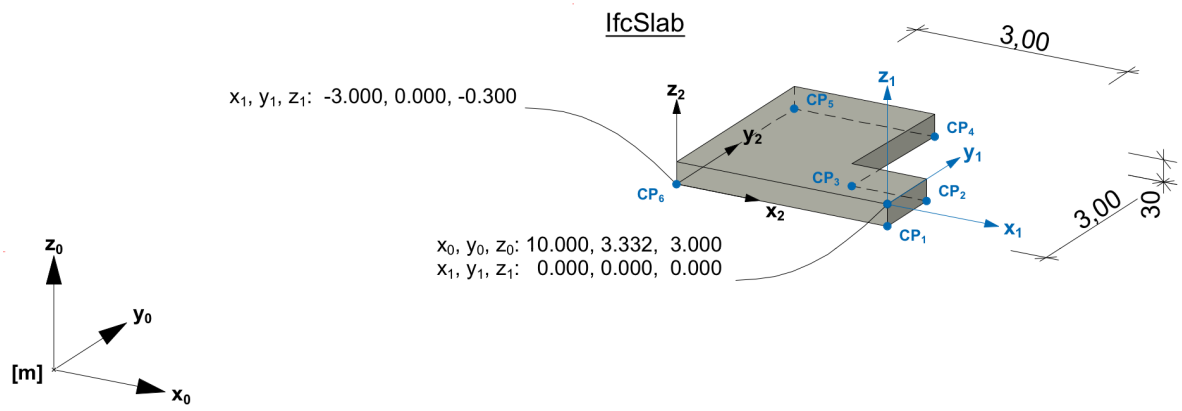

Abb. 4.6.: Geometrische Angaben aus der *IfcSpace*.

Informationsdaten über das Höhenniveau der Decke werden über zwei Wege der IFC-Datenstruktur zugänglich, die durch den Quelltext 4.6 verdeutlicht werden sollen. Die erste Möglichkeit besteht aus der Referenzierung auf die Entität #40072, welche über eine optionale Beziehung zu einer *IfcObjectPlacement* sowie weiteren Referenzierungen führt. Im Gegensatz dazu besteht eine weitere Variante, durch eine inverse Beziehung innerhalb der IFC-Datenstruktur. Dabei verweist die Entitätennummer #40160 auf eine *IfcBuildingStory*, welche Auskunft über das ‚innerhalb der CAD/BIM Software, eingestellte Geschoss gibt. Beide Wege führen zu der Information, dass die Höhenlage des lokalen Koordinatensystems um den Wert von $3.000\ m$ zu ergänzen ist.

[6] Einsatzmöglichkeiten für die &HOLE-Anweisung werden im nachfolgenden Abschnitt des Kapitels erläutert.

Quelltext 4.6: Angaben zu der Höhenlage in der IfcSlab.

```
#40266 = IFCSLAB('1bOkeNMipiHQU2BZbUPAmo',#12,'Decke-002',$,$,#40228, // direkte Beziehungen
         #40262,'6562EA17-5ACC-EC45-A782-2E395E64AC32',.FLOOR.);
#40228 = IFCLOCALPLACEMENT(#40072,#40227);
#40072 = IFCLOCALPLACEMENT(#113,#40071);
#40071 = IFCAXIS2PLACEMENT3D(#40069,#40067,#40065);
#40069 = IFCCARTESIANPOINT((0.,0.,3.));

#40160 = IFCRELCONTAINEDINSPATIALSTRUCTURE('2Zq$Pgax35G1lJl5zs_3xT',  // inverse Beziehung
         #12,$,$,(#40157,#40266),#40073);
#40073 = IFCBUILDINGSTOREY('2u8QLkhhx4HBrLqA_wnknJ',#12,'1. OG',
         $,$,#40072,$,$,.ELEMENT.,3.);
```

Im Hinblick einer Konzeptumsetzung ist zu vermerken, dass inverse Beziehungen zu zusätzlichen Filterroutinen innerhalb der IFC-Datenstruktur führen. Im vorliegenden Fall können dadurch objektorientierte Abfragen in einem Programm reduziert werden, was gegenüber der ersten Variante als Vorteil angesehen werden kann.

4.2.3. Öffnungen in Wänden am Beispiel der IfcWindow

Mithilfe der &HOLE-Anweisung können im *Fire Dynamics Simulator* einzelne Öffnungen in vorhandene Gegenstände vorgesehen werden. Im leichtesten Fall wird hierdurch die Modellierung von offenen Fenstern oder Türen in Wänden möglich, was die Einführung der Abbildung 4.7 begründet. Informationsdaten zu diesen Gebäudeelementen sind der *IfcDoor* bzw. der *IfcWindow* zu entnehmen. Exemplarisch wird die Datengewinnung an einem Fenster verdeutlicht, wobei der Ansatz eine rechteckige Aussparung mit Wandtiefe verfolgt. Der Quelltext 4.7 zeigt hierzu den Informationsgehalt der *IfcWindow* mit der Entitätennummer #2360, durch die unmittelbar die Höhe sowie Breite des Fensters angegeben wird. Als weiteres Attribut wäre eine direkte Angabe zu der Wandbreite wünschenswert, was jedoch die *IfcWindow* nicht unmittelbar bereitstellt. Als Lösung sieht die Arbeit in einem ersten Schritt den Filtervorgang von zwei inversen Beziehungen vor, was die Entitäten #2296 und #2247 ausdrücken sollen. Über die Entität der *IfcOpeningElement* kann das Wandbauteil, in der das Fenster vorgesehen ist, ermittelt werden. Damit kann die Wandbreite abgefragt und die Abmessungen der &HOLE-Anweisung vervollständigt werden. Weiterhin ist es möglich den geometrischen Ort der Fensteröffnung zu bestimmen. Im Quelltext 4.7 wird das mit den Schritten 2 und 3 dargelegt, wobei Abbildung 4.7 das Vorgehen grafisch untermauert. Bezogen auf das lokale Koordinatensystem der Wand (x_1, y_1, z_1) zeigt die Grafik ein zweites Koordinatensystem (x_2, y_2, z_2), das durch die Entität der *IfcOpeningElement* hervorgerufen wird. Der Bezugspunkt für die Abmessungen des Fensters ist durch einen lokalen Versatz beschrieben und in der Abbildung durch das Koordinatensystem x_3, y_3, z_3 verdeutlicht. Ausgehend von diesem Punkt wird eine $1.20\ m$ breite, $1.50\ m$ hohe sowie $0.365\ m$ tiefe Wandöffnung dargestellt. Die Berechnungsvorschrift für die Transformation ist im Quelltext 4.7 enthalten[7].

[7] Die Ansätze werden im Kapitel 6 mit einem Programm verifiziert.

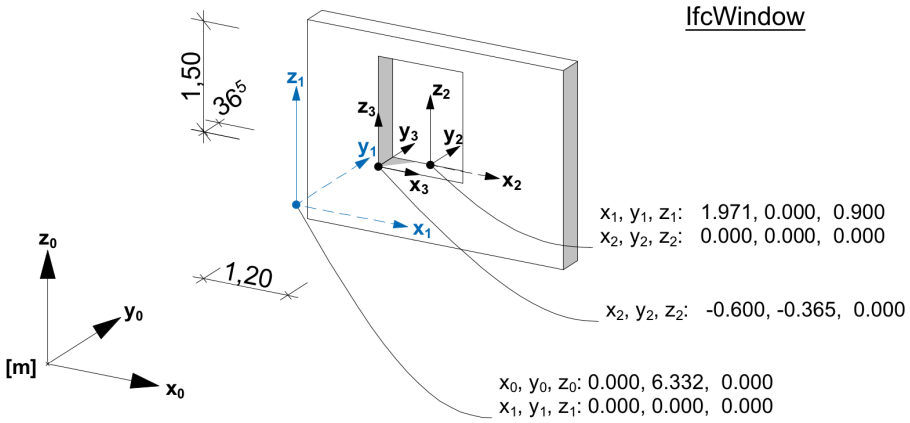

Abb. 4.7.: Geometrische Angaben zur Beschreibung einer Wandöffnung.

Quelltext 4.7: Generierung einer &HOLE-Anweisung aus der IfcWindow

```
// 1. Inverse Beziehungen in der IFC-Datenstruktur
#2360 = IFCWINDOW('0$JCI5IpacIe2Dmc_6N3ge',#12,'Fenster-002',$,$,
        #2342,#2356,'3F4CC485-4B39264A-808D-C26F865C3AA8',1.5,1.2,$,$,$);
#2363 = IFCRELFILLSELEMENT('1mvmaNDHmVV2byuEpzlBkh',#12,$,$,#2296,#2360);
#2296 = IFCOPENINGELEMENT('2de6u47YaGE7$x0s9LiRch',#12,'Fenster-002',$,$,
        #2267,#2292,'A7A06E04-1E29-1038-7FFB-036255B1B9AB',$);
#2299 = IFCRELVOIDSELEMENT('3_MfeWWNfsfqln5a2w$yXU',#12,$,$,#2247,#2296);
#2247 = IFCWALL('1fYCD$XnKhJeI9hiJq13Sm',#12,'Au\X2\00DF\X0\enwand-002',
        $,$,#2153,#2242,'6988C37F-8715-2B4E-8489-AEC4F4043730',$);

// 2. Direkte Beziehungen der IfcOpeningElement und IfcWall
#2267 = IFCLOCALPLACEMENT(#2153,#2266);
#2266 = IFCAXIS2PLACEMENT3D(#2264,#2262,#2260);
#2264 = IFCCARTESIANPOINT((1.970833661116,0.,0.9));
#2292 = IFCPRODUCTDEFINITIONSHAPE($,$,(#2285,#2290));
#2290 = IFCSHAPEREPRESENTATION(#457,'Box','BoundingBox',(#2289));
#2289 = IFCBOUNDINGBOX(#2287,1.2,1.095,1.5);
#2287 = IFCCARTESIANPOINT((-0.6,-0.365,0.));

#2153 = IFCLOCALPLACEMENT(#128,#2152);
#2152 = IFCAXIS2PLACEMENT3D(#2150,#2148,#2146);
#2150 = IFCCARTESIANPOINT((0.,6.33229828605,0.));
#2242 = IFCPRODUCTDEFINITIONSHAPE($,$,(#2225,#2231,#2239));
#2231 = IFCSHAPEREPRESENTATION(#457,'Box','BoundingBox',(#2230));
#2230 = IFCBOUNDINGBOX(#2228,4.,0.365,2.7);

// 4. Inverse Beziehung Hoehenlage
#486  = IFCRELCONTAINEDINSPATIALSTRUCTURE('1aaIejLZM8y90Bvmu5Zi20',#12,$,$,
        (#471,#657,#2080,#2247,#2360,#2519,#7399,#12909,#13004,#13099,#23455,#39901),#130);
#130  = IFCBUILDINGSTOREY('2Wk49hkHPC2hvWchm5NASJ',#12,'EG',$,$,#128,$,$,.ELEMENT.,0.);

// 5. Generierung FDS-Eingabebefehl
&HOLE XB = X1,X2,Y1,Y2,Z1,Z2
&HOLE XB =  (0.+1.970-0.6),(0.+1.970-0.6+1.2),(6.332+0.-0.365),
            (6.332+0.-0.365+0.365),(0.+0.+0.9+0.),(0.+0.+0.9+0.+1.5)
&HOLE XB = 1.370,2.570,5.967,6.332,0.900,2.400
```

4.2.4. Achsausrichtungen der lokalen Koordinatensysteme

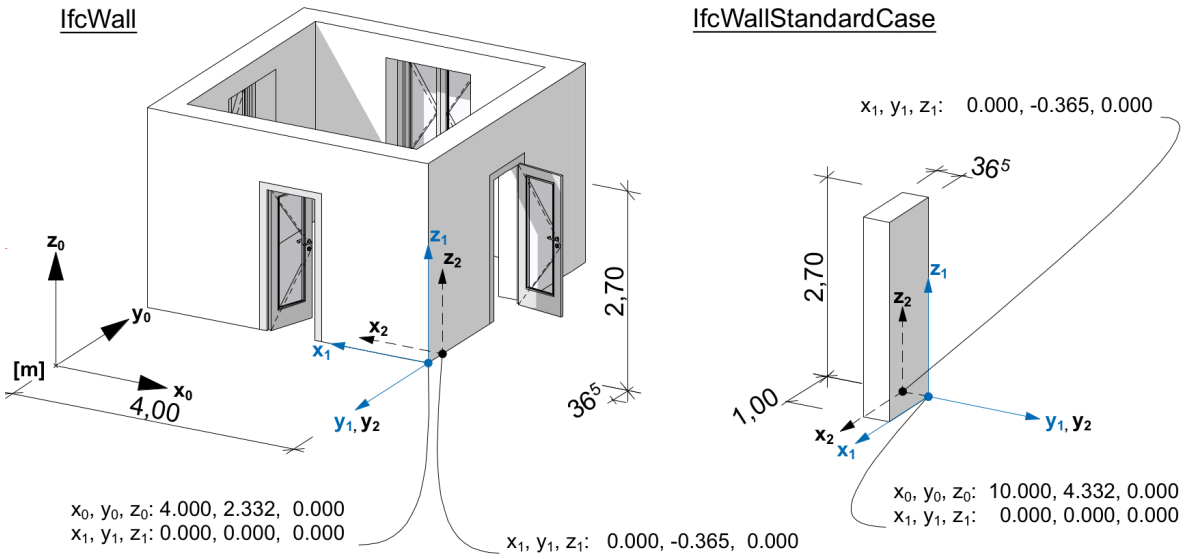

Abb. 4.8.: Beispiele für die Achsausrichtungen der lokalen Koordinatensysteme.

Die bisher betrachteten Geometrien weisen lokale Koordinatensysteme (x_1, y_1, z_1) auf, deren Achsen in den Richtungen des globalen Koordinatensystems (x_0, y_0, z_0) verlaufen. Folglich werden die Rechenvorschriften für Transformationen von Informationsdaten in konkrete FDS-Eingabefehle problemlos ermöglicht. Die Abbildung 4.8 durchbricht diese Darstellungsform und zeigt Möglichkeiten in der Rotation der lokalen Achsen auf. Für das, durch die *IfcWallStandardCase* abgebildete, Wandelement wird beispielsweise die globale Achsausrichtung x_0 durch die Richtung y_1 ersetzt. Die übrigen Achsrichtungen ergeben sich aufgrund des kartesischen Koordinatensystems nach der *Rechten Handregel*. Ein Versatz wird, wie auch in den vorherigen Grafiken gezeigt, durch ein zusätzliches Koordinatensystem (x_2, y_2, z_2) beschrieben und orientiert sich an der Verdrehung der lokalen Achsen. Das Verhalten wird durch den Quelltext 4.8 verdeutlicht, in dem die Entitätennummer #12953 die Rotation des Koordinatensystems vorschreibt. Für die nach Abbildung 4.1 eingeführten Gebäudeelemente können demnach vier Rotationen ermittelt werden: ($1.$, $0.$, $0.$), ($0.$, $1.$, $0.$), ($-1.$, $0.$, $0.$) und ($0.$, $-1.$, $0.$). Die z-Richtung gestaltet sich für alle Elemente gleichbleibend ($0.$, $0.$, $1.$) und wird somit für den weiteren Verlauf der Arbeit unrelevant. Die vier Rotationsvarianten werden in der Abbildung 4.9 ohne lokale Versätze dargestellt. Schräge Geometrieverläufe können durch Angaben im Intervall $[0, 1]$ erreicht werden. Zum Beispiel wird durch die Zahlenwerte (0.707, 0.707, $0.$) eine im $45°$-Winkel verlaufende Geometrie in Richtung des globalen Koordinatensystems dargestellt[8].

[8] Angaben im Bogenmaß; eine weitere Betrachtung wird nach Abschnitt 4.1 ausgeschlossen.

Quelltext 4.8: Beispiel für eine Achsausrichtung im lokalen Koordinatensystem

```
#13004 = IFCWALLSTANDARDCASE('1eTCwoamvkJPqjhvH1AY3V',#12,'Au\X2\00DF\X0\enwand-004',$,$,
         #12960,#12999,'6874CEB2-930E-6E4D-9D2D-AF94412A20DF',$);

#12960 = IFCLOCALPLACEMENT(#128,#12959);
#12959 = IFCAXIS2PLACEMENT3D(#12957,#12955,#12953);
#12957 = IFCCARTESIANPOINT((10.,4.33229828605,0.));
#12953 = IFCDIRECTION((0.,-1.,0.));                    // Rotation  in xy-Ebene
#12955 = IFCDIRECTION((0.,0.,1.));

#12999= IFCPRODUCTDEFINITIONSHAPE($,$,(#12982,#12988,#12996));
#12988= IFCSHAPEREPRESENTATION(#457,'Box','BoundingBox',(#12987));
#12987= IFCBOUNDINGBOX(#12985,1.,0.365,2.7);
#12985= IFCCARTESIANPOINT((0.,-0.365,0.))              // Versatz (rotationsabhaengig)
```

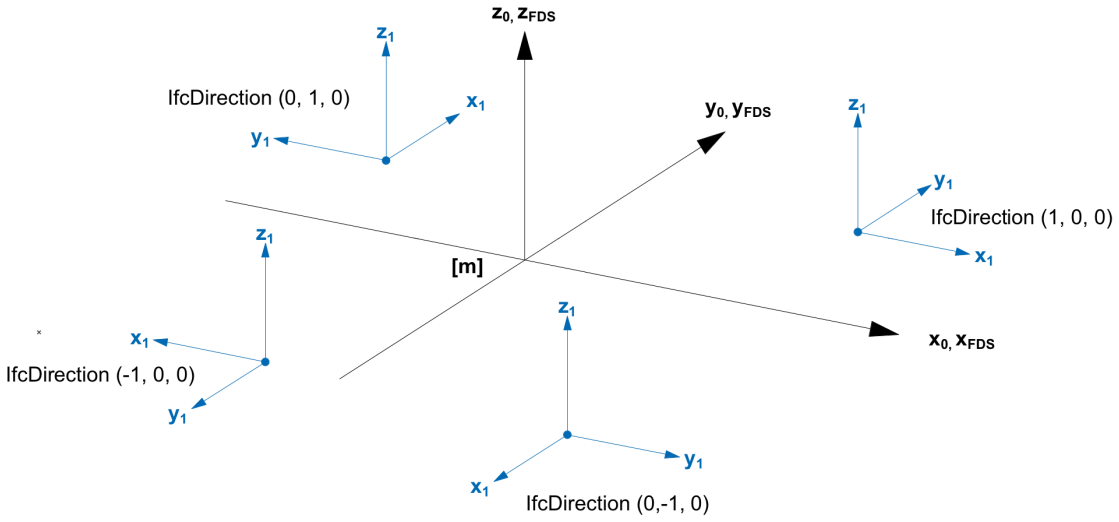

Abb. 4.9.: Grafische Darstellung der Rotationsvarianten in der xy-Ebene.

Die Rotationen sind für die Generierung von FDS-Eingabebefehlen zu berücksichtigen und führen zu der Notwendigkeit, Rechenvorschriften für die Transformationen der Informationsdaten zu entwickeln. In dem Zusammenhang wird auf die Darstellung der einzelnen FDS-Eingabebefehle verzichtet, um in dem Abschnitt 4.3 allgemeingültige Vorschriften einzuführen.

4.2.5. Informationen aus der IfcPropertySet

Die *Industry Foundation Classes* klassifizieren Eigenschaften der Bauteile u.a. mithilfe der *IfcPropertySet*, was im Quelltext 4.9 durch die Entitätennummer #553 veranschaulicht wird[9]. Eine *IfcRelDefinesByProperties* fungiert dabei als Bindeglied zwischen der Deckenentität und den zugewiesenen Eigenschaften, wodurch eine inverse Relation entsteht.

Quelltext 4.9: Beispiel für eine Bauteilklassifizierung durch die IfcPropertySet

```
#471 = IFCSLAB('0iFuOXwidzIhaetDff_ZCU',#12,'Decke-001',$,$,
       #143,#465,'2C3F8621-EAC9-FD4A-B928-DCDA69FA331E',.FLOOR.);
#568 = IFCRELDEFINESBYPROPERTIES('3Ec8EZytBuR8C6UssRZqYU',#12,$,$,
       (#471),#553);
#553 = IFCPROPERTYSET('3INLCU4RB3VSgocrgyS463',#12,'Pset_SlabCommon',$
       (#545,#546,#547,#548,#549,#550,#551,#552));
#546 = IFCPROPERTYSINGLEVALUE('SurfaceSpreadOfFlame',$,IFCLABEL('1,5'),$);
#548 = IFCPROPERTYSINGLEVALUE('Combustible',$,IFCBOOLEAN(.T.),$);
#550 = IFCPROPERTYSINGLEVALUE('FireRating',$,IFCLABEL('REI 90'),$);
```

In der *IfcPropertySet* wird ein *Set* an Entitäten von der *IfcPropertySingleValue* hinterlegt, deren Inhalte Informationen zu der brandschutztechnischen Infrastruktur ermöglichen. In dem Zusammenhang enthält die Tabelle 4.1 Hinweise zur Generierung einzelner Simulationsparameter. Insgesamt ist damit die Generierung von FDS-Eingabebefehlen nur indirekt möglich, was die Bereitstellung weiterer Informationsdaten erfordert.

Tabelle 4.1.: Brandschutztechnische Infrastruktur aus der *IfcPropertySet*.

Parameter	Kurzbeschreibung	Einheit
Fire Rating	Feuerwiderstandsklasse	REI90
Combustible	Brennbares Material	TRUE
Surface Spread of Flame	Flammenausbreitung Oberfläche	1.5
Fire Exit	Notausgang, Rettungsweg	TRUE
Self Closing	Selbstschließende (Tür)	FALSE
Smoke Stop	Rauchdichte (Tür)	TRUE

An dieser Stelle der Arbeit endet die Untersuchung des Informationsraums der *Industry Foundation Classes* für Anwendungen mit dem *Fire Dynamics Simulator*. Abschließend werden die wesentlichen Erkenntnisse zusammengeführt und auf die Notwendigkeit weiterer Informationsdaten eingegangen.

[9] Zum Beispiel Materialien und Materialschichten durch die inverse Beziehung der *IfcRelAssociatesMaterial*.

4.3. Zusammenfassung der Erkenntnisse

Die Untersuchung ermöglicht das Aufstellen von allgemeinen Rechenvorschriften, welche Transformationen von einzelnen IFC-Informationsdaten in konkrete FDS-Eingabebefehle vornehmen. Es wird der Ansatz verfolgt, mithilfe einer *IfcBoundingBox* die Eckpunkte für eine XB-Anweisung zu generieren. Die Vorschriften beschränken sich auf die im Abschnitt 4.2.4 gezeigten Rotationen und nehmen Bezug auf die in den Abbildungen eingeführten Koordinatensysteme. Aufgrund der im Abschnitt 4.2 gezeigten Vererbungshierarchie können Gebäudeelemente der *IfcBuildingElement*, unter Ausnutzung der Filterroutinen, in den *Fire Dynamics Simulator* automatisiert übertragen werden. Die Rechenvorschriften sind im Anlagenteil vollständig hinterlegt und werden im Kapitel 6 verifiziert.

Formeln 4.1.: Beispiel für Transformationsvorschriften aus der *IfcDirection* (-1, 0, 0).

$$
\begin{aligned}
X_{1,fds} &= x_0 - x_1 \\
X_{2,fds} &= x_0 - x_1 - x_{dim} \\
Y_{1,fds} &= y_0 - y_1 \\
Y_{2,fds} &= y_0 - y_1 - y_{dim} \\
Z_{1,fds} &= z_0 + z_1 + z_{elevation} \\
Z_{2,fds} &= z_0 + z_1 + z_{elevation} + z_{dim}
\end{aligned}
$$

$$
\begin{aligned}
X_{1,fds} &= x_{0,wall} - x_{1,wall} - x_{1,opening} - x_2 \\
X_{2,fds} &= x_{0,wall} - x_{1,wall} - x_{1,opening} - x_2 - x_{dim,opening} \\
Y_{1,fds} &= y_{0,wall} - y_{1,wall} \\
Y_{2,fds} &= y_{0,wall} - y_{1,wall} - y_{dim,wall} \\
Z_{1,fds} &= z_{0,wall} + z_{1,wall} + z_{1,opening} + z_2 + z_{elevation} \\
Z_{2,fds} &= z_{0,wall} + z_{1,wall} + z_{1,opening} + z_2 + z_{elevation} + z_{dim,opening}
\end{aligned}
$$

Die vorliegende Arbeit distanziert sich von der Verwendung der *IfcPropertySet*. Einerseits wird dies mit den inversen Beziehungen zwischen den Gebäudeelementen und deren Eigenschaften begründet, was im Hinblick einer Programmentwicklung zu weiteren Filterroutinen führt (vgl. Quelltext 4.9). Zum anderen können FDS-Eingabebefehle nur durch zusätzliche Informationen außerhalb der *Industry Foundation Classes* generiert werden. Abschließend ist festzuhalten, dass die in Tabelle 4.1 dargestellten Parameter, Bestandteile einer eigenständigen Brandschutzfachplanung darstellen und damit die Grundidee von Multimodellen tangieren. Mithilfe des Multimodellansatzes kann auf die Übertragung von fachspezifischen Informationsdaten in ein vorhandenes Datenmodell (z.B. in die IFC), durch gezielte Verlinkungen einzelner Fachmodelle verzichtet werden.

5. Das Multimodell mit dem Variationsprinzip

Nimm an, was nützlich ist. Lass weg, was unnütz ist.
Und füge das hinzu, was dein Eigenes ist.

(Bruce Lee)

5.1. Bildung der Fachmodelle

Multimodelle bündeln die Informationsräume der Fachmodelle in einer einzigen Informationsressource (vgl. Abschnitt 2.3.1). In dem Zusammenhang zeigt Kapitel 4, wie Informationen für die Generierung von FDS-Eingabebefehlen aus einem Bauwerksmodell gewonnen werden. Die vorliegende Arbeit stellt mit diesem Abschnitt drei weitere Modelle vor, die ein eigenständiges Fachgebiet repräsentieren und deren Informationsgehalt das Erzeugen einzelner Anweisungen für den *Fire Dynamics Simulator* unterstützen. Die nachfolgenden Abschnitte zeigen, wie aus Informationen konkrete Datenstrukturen entstehen und wie sie einen Mehrwert für das Simulationsprogramm bieten. Hierfür werden notwendige Daten erschlossen, in Beziehung gesetzt und deren Bedeutung anhand konkreter FDS-Eingabebefehle veranschaulicht.

Die Datenstrukturen der Fachmodelle werden in der Modellierungssprache *EXPRESS* erstellt, wodurch eine Schnittmenge zu den *Industry Foundation Classes* gebildet wird[1]. Mit der Programmierschnittstelle *JSDAI* [LKS] können *EXPRESS*-Schemen geschrieben, grafisch dargestellt und deren Funktionen in Programmroutinen geprüft werden. Auf der Compact Disk sind die jeweiligen Datenschemen mit den zugehörigen *Java*-Programmen hinterlegt. Aufgrund der Modellierungssprache *EXPRESS* werden die Fachmodelle jeweils in *STEP Physikal Files* geschrieben, die der Compact Disk entnommen werden können.

Die Gemeinsamkeit aller Fachmodelle besteht in der eindeutigen Identifikation einzelner Elemente, was Abschnitt 2.3.1 verdeutlicht. In Abbildung 5.1 wird hierfür eine abstrakte Klasse *Root* eingeführt, deren Attribute die Vergabe von eindeutigen Identifikatoren (id) sowie deren Beschreibungen (fyi) ermöglichen.

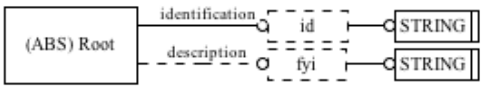

Abb. 5.1.: *EXPRESS*-Datenschema für die Vergabe eindeutiger Identifikatoren.

Innerhalb der Datenstrukturen können Klassen von *Root* erben und deren Attribute erhalten, was bei komplexen Datenstrukturen die mehrfache Vergabe von Attributen reduziert. Die nachfolgenden Datenstrukturen stellen sich überschaubar dar, womit im Hinblick auf deren grafischen Darstellung auf diese Möglichkeit verzichtet wird. Eine ID wird demnach nur für Klassen vorgesehen, deren Objekte für die spätere Verlinkung von Interesse sind[2].

[1] Aufgrund des Multimodellansatzes ist die Modellierungssprache variabel.
[2] Mit der abstrakten Klasse werden weitere ID-Attribute umgangen.

5.1.1. Brandschutztechnische Infrastruktur

Der § 66 MBO regelt die Einhaltung der Anforderungen an den Brandschutz durch bautechnische Nachweise (2002, S. 54ff). Für Anlagen und Räume besonderer Art und Nutzung gemäß § 2 (4) MBO können hieraus bauwerksbezogene Brandschutzkonzepte resultieren (2002, S. 6f). „Brandschutzkonzepte bestehen aus dem Zusammenwirken einzelner Brandschutzkomponenten, abgeleitet entweder aus Brandschutzvorschriften oder im Einzelfall aus Erfahrungswerten“ [MB18, S. 4 (8.1)]. Die vorliegende Arbeit bedient sich dieser Thematik, in dem sie das Aufstellen eines Brandschutzkonzepts als eigenständige Fachplanung ansieht. Bauteilanforderungen sollen durch farbliche Kennzeichnungen im Simulationsprogramm hinterlegt werden und ein grundlegendes Anforderungsprofil wiedergeben. In dem Zusammenhang zeigt Tabelle 5.1 elementare Informationsdaten für ein Fachmodell, das die Beschreibung einer brandschutztechnischen Infrastruktur in einem Gebäude ermöglicht[3].

Tabelle 5.1.: Informationen für eine brandschutztechnische Infrastruktur.

Information	Kurzbeschreibung	Datenquelle
Gebäudeklasse	1a, 1b, 2, 3, 4, 5	[ARG02, S. 5f]
Sonderbau	Hochhäuser, Krankenhäuser, Schulen, ...	[ARG02, S. 6f]
Nutzungseinheit	Wohnungen, Büros, Praxen, ...	[Inn17, S. 1]
	Angabe einer Brutto-Grundfläche	
Bauteile	Wände, Decken, Stützen, Unterzüge, ...	[MB18, S. 1 (4.5)]
Anforderungen	Brandverhalten, Tragwerk, Raumabschluss	[MB18, S. 3 (4.4)]
Klassifizierung	feuerbeständig, feuerhemmend ...	[ARG02, S. 21]
	DIN 4102-2, DIN 4102-4, DIN EN 13501-2	[MB18, S. 2 (4.4)]
Visualisierung	Farbliche Kodierung	[MB18, S. 34 (5.3.3)]

Die Abbildung 5.2 stellt die, aus den gewonnenen Informationen, erstellte Datenstruktur dar. Das Modell beschreibt durch eine Instanz der Klasse *Fire_Protection_Infrastructure* die Brandschutzanforderungen an ein Gebäude. Mit den beiden Enumerationen *class_of_building* und *special_building* sind die Zuweisungen einer Gebäudeklasse und optional eines Sonderbautatbestands möglich. Der Instanz ist mindestens ein Objekt aus der Klasse *Unit_Of_Use* zuzuordnen, das durch Attribute eine Nutzungseinheit und Brutto-Grundfläche beschreibt. Damit werden in der Instanz Informationen hinterlegt, die eine bauordnungsrechtliche Einordnung widerspiegeln. Aufgrund der Einordnung resultieren bauaufsichtliche Anforderungen, welche in der Instanz durch eine Vielzahl an Referenzierungen gelistet werden können. Eine Besonderheit liegt in der gewählten Vererbungsstruktur, die eine Generierung der Anforderungen an raumabschließende (RC) und tragende (LB) Bauteile vornimmt. Raumabschlüsse sind nur durch Decken-

[3] Die Arbeit fokussiert sich auf die Beschreibung von Bauteilanforderungen, was die Betrachtung weiterer Brandschutzvorgaben, z.B. im Hinblick der Rettungswegführung, ausschließt.

und Wandbauteile (component_group1) möglich, wobei diese eine tragende Funktion einnehmen können. Hingegen sind Stützen und Unterzüge (component_group2) Bestandteile von Tragwerken, womit eine raumabschließende Funktion auszuschließen ist. Auf europäischer Ebene werden die Funktionen durch unterschiedliche Bezeichnungen klassifiziert (din_en_13501_rei/ei/r), was zu den drei dargestellten Klassen führt. Durch die Enumerationen werden die Bezeichnungen eindeutig einer Funktion zugeordnet und fehlerhafte Klassifizierungen vermieden. Die abstrakte Klasse ((ABS)_Component_Request) erlaubt die Vererbung von bauaufsichtlichen und national normativen Bezeichnungen. Jede Bauteilanforderung verweist auf ein Objekt aus der Klasse *Rgb*, wobei sich diese über drei definierte Datentypen (*rgb_value) darstellen.

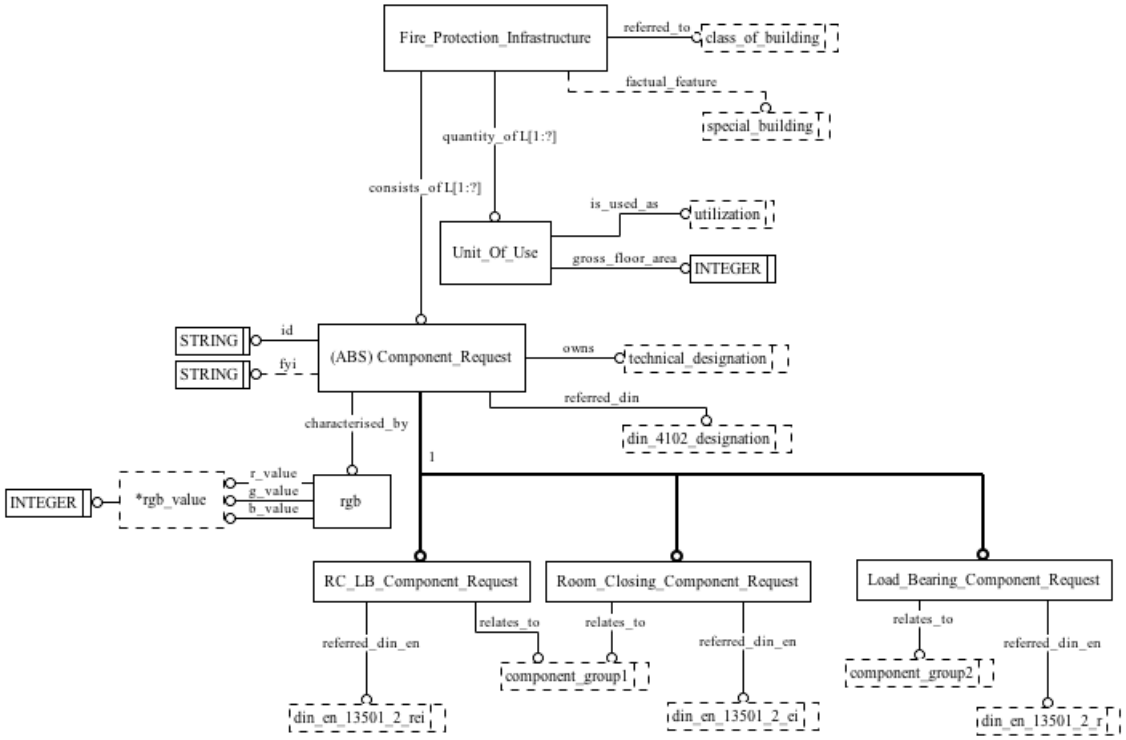

Abb. 5.2.: *EXPRESS*-Datenschema für eine brandschutztechnische Infrastruktur.

Quelltext 5.1: Darstellung der Bauteilanforderungen im FDS durch RGB-Anweisungen

```
#8 = FIRE_PROTECTION_INFRASTRUCTURE(.GK_3.,.NR_4_VERKAUFSTAETTEN.,(#1),
     (#5,#6,#7));
#1 = UNIT_OF_USE(.BUERO.,400);
#5 = RC_LB_COMPONENT_REQUEST('rc1b1_265398g5dr4','Trennwaende zwischen
     Nutzungseinheit EG',.FEUERHEMMEND.,.F_30B.,#2,.WAND.,.REI_30.);
#2 = RGB(255,185,15);
&OBST XB = X1,X2,Y1,Y2,Z1, 2, RGB = 255,185,15 /
```

Die Datenstruktur stellt eine Basis für Informationen aus einem Brandschutzkonzept dar. Durch die Tabelle 5.1 werden größtenteils in sich geschlossene Informationsbereiche beschrieben, womit die überwiegende Verwendung von Enumerationen begründet ist. Es ist festzuhalten, dass Bauteilanforderungen durch die *Industry Foundation Classes* bereit gestellt werden können. Die vorliegende Arbeit zeigt mit dem Fachmodell eine Möglichkeit auf, den Informationsursprung zu berücksichtigen und ein Fachgebiet vollständig zu repräsentieren. Bauteilanforderungen können durch Informationswerte aus einem RGB-Farbraum problemlos für FDS-Eingabebefehle verwendet werden, was abschließend in Quelltext 5.1 verdeutlicht wird.

5.1.2. Informationen zu der Entstehung von Bränden

McGrattan et al. zeigen Möglichkeiten auf, Brandereignisse mithilfe von Pyrolyse-Modellen im *Fire Dynamics Simulator* zu simulieren. Die Eigenschaften von Oberflächen werden mit flächenbezogenen Wärmefreisetzungsraten definiert, wobei zeitliche Funktionsverläufe zugelassen werden (2017, S. 78f). Die Arbeit sieht hierin einen Ansatz, die Entstehung von Bränden zu simulieren.

Das Simulationsprogramm erfordert Eingabeparameter für eine maximale Wärmefreisetzungsrate und zeitabhängige Funktionswerte. Die Parameter können prinzipiell frei gewählt werden, was hinsichtlich einer realistischen Simulation jedoch in Frage zu stellen ist. Die Parametrisierung erfolgt daher in Anlehnung an natürliche Brandereignisse, was die Einführung der Abbildung 5.3 begründet und eine Schnittmenge zu den Ingenieurmethoden des Brandschutzes bildet.

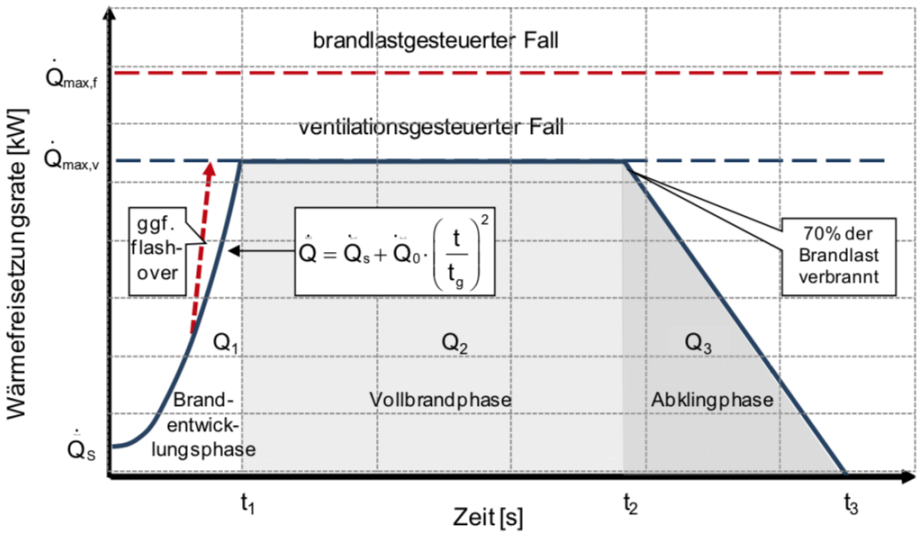

Abb. 5.3.: Schematisierter Brandverlauf für einen natürlichen Brand [Hos13, S. 63].

Die Abbildung 5.3 zeigt Hosser im Zusammenhang mit einem vereinfachten Naturbrandmodell für die Bemessung von Bauteilen (2013, S. 62). Die Grafik unterscheidet zwischen einem brandlast- und ventilationsgesteuerten Brandereignis. „Bei einem brandlastgesteuerten Brand ist die Wärmefreisetzung durch die brennende Oberfläche der Brandlast begrenzt“ [Hos13, S. 59]. Im Gegensatz dazu beschreibt Hosser einen ventilationsgesteuerten Brand mit einer Begrenzung, der in einem Raum zur Verfügung stehenden, Verbrennungsluft (2013, S. 60). „Die Verbrennung im Raum wird somit durch die über die Öffnungen ein- und ausströmenden Gasanteile limitiert“ [Hos13, S. 60]. Beide Fälle beschreiben ein Brandereignis jeweils in drei Phasen. Die Brandentwicklungsphase zeigt dabei einen quadratischen Verlauf bis zum Erreichen einer max. Wärmefreisetzungsrate (Zeitpunkt t_1). Während der Vollbrandphase wird eine konstante Wärme freigesetzt, die dabei der maximalen Wärmefreisetzungsrate entspricht. Nach dem 70% der Brandlast abgebrannt sind (Zeitpunkt t_2), klingt das Brandereignis durch einen linearen Funktionsverlauf vollständig ab (Zeitpunkt t_3). Insgesamt wird damit der zeitliche Verlauf einer Wärmefreisetzungsrate in drei Phasen und durch die Zeitpunkte t_1, t_2, und t_3 beschrieben. Die unterhalb des Funktionsverlaufs eingeschlossene Fläche entspricht der Brandlast Q, die sich innerhalb der Phasen in Q_1, Q_2 und Q_3 unterteilt.

Für die Modellierung von Entstehungsbränden bedient sich die Arbeit an dem Funktionsverlauf für ein brandlastgesteuertes Brandereignis (vgl. Abbildung 5.3). Damit wird vorausgesetzt, dass für die Entstehung eines Brandes ausreichend Verbrennungsluft zur Verfügung steht[4]. In dem Fachmodell sollen Informationen zu maximalen Wärmefreisetzungsraten sowie den darauf bezogenen Brandlasten bereitgestellt werden. Damit wird die Berechnung der Zeitpunkte t_1, t_2, und t_3 ermöglicht[5]. Die Erschließung der hierfür notwendigen Daten stützt sich auf zwei Ansätze, die nachfolgend näher erläutert werden.

Dem Anhang E der DIN EN 1992-1-2/NA:2010-12 können Brandlastdichten und flächenbezogene Wärmefreisetzungsraten (Tabelle E.4 und E.5) entnommen werden. Die Angaben beziehen sich auf Nutzungen, wie beispielsweise Büro- und Wohnungseinheiten und deren Grundflächen. Für einen brandlastgesteuerten Brand verdeutlicht Mehl in dem Zusammenhang folgende Beziehung (2017, S. 75):

$$\dot{Q}_{max,f,k} = RHR_f \cdot A_f \tag{5.1}$$

Das Produkt aus dem charakteristischen Wert einer flächenbezogenen Wärmefreisetzungsrate RHR_f und der Grundfläche eines Raumes A_f führt zu einem Maximalwert einer Wärmefreisetzungsrate $\dot{Q}_{max,f,k}$. Die in einem Raum vorhandene Brandlast Q ergibt sich mit der Angabe einer Brandlastdichte $q_{f,k}$ nach:

$$Q = q_{f,k} \cdot A_f \tag{5.2}$$

[4] Raumöffnungen können vernachlässigt werden.
[5] Berechnung der Zeitpunkte t_1, t_2, und t_3 innerhalb einer Programmroutine.

Für die Beschreibung der Brandentwicklungsphase werden in der Tabelle E.5 [Nor10, S. 57] Wachstumsraten mit einer Referenzzeit t_α angegeben. Unter der Voraussetzung, dass Angaben zu Q_s und Q_0 unberücksichtigt bleiben ($Q_s = 0$, $Q_0 = 1$) können die Zeitpunkte t_1, t_2, und t_3 wie folgt ermittelt werden:

$$t_1 = t_\alpha \cdot \sqrt{\dot{Q}_{max,f,k}} \tag{5.3}$$

$$Q_1 = \frac{{t_1}^3}{3 \cdot {t_\alpha}^2} \tag{5.4}$$

$$Q_2 = 0,7 \cdot Q - Q_1 \tag{5.5}$$

$$t_2 = t_1 + \frac{Q_2}{\dot{Q}_{max,f,k}} \tag{5.6}$$

$$Q_3 = 0,3 \cdot Q \tag{5.7}$$

$$t_3 = t_2 + \frac{2 \cdot Q_3}{\dot{Q}_{max,f,k}} \tag{5.8}$$

Der Zeitpunkt t_1 kann mit dem Maximalwert $\dot{Q}_{max,f,k}$ durch die Umstellung der, in Abbildung 5.3 gezeigten, quadratischen Funktion ermittelt werden. Das bestimmte Integral der Funktion führt zu der Brandlast Q_1 und ermöglicht die Berechnung von Q_2. Als maßgebendes Kriterium gilt, dass der Vollbrandphase 70% der vorhandenen Brandlast zugeordnet werden. Analog ergeben sich die Werte Q_3, t_2 und t_3 für den vorgegebenen Funktionsverlauf. Insgesamt wird damit ermöglicht, ein Brandereignis auf die Grundfläche eines Raums unter Berücksichtigung normativer Vorgaben zeitlich zu begrenzen.

Ein weiterer Ansatz besteht in der Bereitstellung von Informationen zu experimentellen Ermittlungen von flächenbezogenen Wärmefreisetzungsraten, was bereits durchgeführte Brandversuche einschließen soll. Als Informationsquelle ist beispielsweise die Publikation VdS 2827:2000-05(01) zu nennen, in denen Wärmefreisetzungsraten angegeben werden [Sch00]. Die in einem Versuch abgebrannten Gegenstände können durch eine Brandlastberechnung beschrieben werden[6].

$$\frac{\sum_{i=1}^{n}(a_i \cdot M_i \cdot Q_i)}{A_{R,j}} \tag{5.9}$$

Mit der Formel 5.9 verdeutlicht Beilicke den Zusammenhang zwischen der Masse M_i , dem Heizwert Q_i und einem Anteilsfaktor a_i von Stoffen (2010, S. 26). Eine rechnerische Brandbelastung resultiert demnach durch die Aufsummierung einzelner Brandlasten, die auf die Fläche $A_{R,j}$ eines Raums bezogen werden. In seinem Werk listet Beilicke eine Sammlung spezifischer Stoffparameter auf und kategorisiert diese beispielsweise

[6] Nach der Auffassung des Kandidaten.

in Möbel- und Einrichtungsgegenstände (2010, S. 97-190). Die Arbeit sieht in der experimentellen Ermittlung von Wärmefreisetzungsraten zusammen mit der Berechnung von Brandlasten eine Möglichkeit, den Funktionsverlauf nach Abbildung 5.3 zu beschreiben[7].

Die Abbildung 5.4 fasst die Ansätze in einer Datenstruktur zusammen. Informationen zu einer Brandentstehung werden durch Instanzen der Klassen *Load_Controlled_Fire* und *Fire_Experiment* bereitgestellt. Objekte aus diesen Klassen erben die Attribute der abstrakten Klasse *Fire_Event*, mit denen Angaben zu flächenbezogenen Wärmefreisetzungsraten und Brandentwicklungszeiten ermöglicht werden. Eine Instanz aus der *Load_Controlled_Fire* bezieht sich durch die Enumeration *utilization* auf eine normativ festgelegte Nutzung, der als weiteres Attribut eine Brandlastdichte zugeschrieben wird. Die Generierung eines Objekts der Klasse *Fire_Experiment* repräsentiert einen Versuchsbrand und kann mehrere Instanzen von *Fire_Load* enthalten. Durch die Attribute dieser Klassen werden Material- und Stoffparameter einzelner Brandlasten festgelegt. Damit soll die Beschreibung von brennbaren Gegenständen durch eine Aufsummierung mehrerer Brandlasten ermöglicht werden.

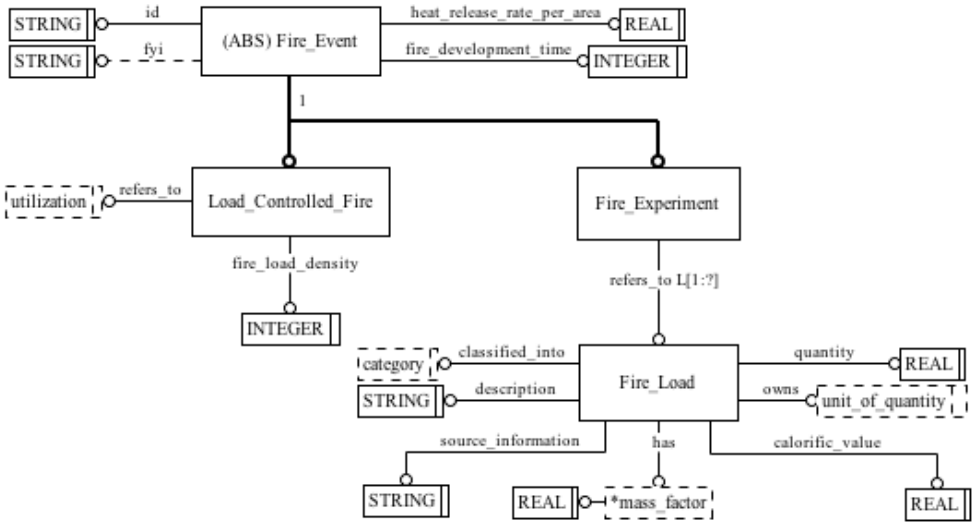

Abb. 5.4.: *EXPRESS*-Datenschema für die Modellierung einer Brandentstehung.

Die Datenstruktur nach Abbildung 5.4 enthält keine Angaben zu der in den Formeln 5.1 und 5.2 verwendeten Grundfläche, da der Ursprung dieser Daten einem Bauwerksmodell (IFC) zugeordnet werden kann. Die Berechnung der Zeitpunkte t_1, t_2, und t_3 ist somit erst durch Verknüpfung der beiden Informationsräume möglich. Der Quelltext 5.2 zeigt abschließend den durch dieses Fachmodell repräsentierten Informationsgehalt sowie Möglichkeiten für die Generierung konkreter FDS-Eingabebefehle.

[7] Berechnung mit Formeln 5.1-5.8 mit RHR_f aus Experiment und Q aus $\sum_{i=1}^{n}(a_i \cdot M_i \cdot Q_i)$.

Quelltext 5.2: Informationsraum für die Modellierung einer Brandentstehung

```
// 1. Moeglichkeit: normative Vorgaben
#1 = LOAD_CONTROLLED_FIRE('lcf1-dfh98emshe2',
     'DIN EN 1991-1-2/NA:2010-12E.4,E.5',250.0,300,.WOHNGEBAUDE.,780);

// 1.1 FDS-Eingabefehl fuer Waermefreisetzungsrate
&VENT XB = X1,X2,Y1,Y2,Z12,Z12, SURF_ID = 'NORM'/
&SURF ID = 'NORM', HRRPUA = 250.0, RAMP_Q = 'Funktionsverlauf'/

// 1.2 Programmroutine Funktionsverlauf ermitteln
Eingangsparamter: 300,780
Berechnung nach : Formeln 5.1-5.8
Ergebnis        : t1,t2,t3

// 1.3 FDS-Eingabefehl fuer Funktionsverlauf
&RAMP ID = 'Funktionsverlauf', T = 0.0, F = 0.0/
&RAMP ID = 'Funktionsverlauf', T = t1,  F = 1.0/
&RAMP ID = 'Funktionsverlauf', T = t2,  F = 1.0/
&RAMP ID = 'Funktionsverlauf', T = t3,  F = 0.0/


// 2. Moeglichkeit: experimentelle Vorgaben
#4 = FIRE_EXPERIMENT('fe1-sef12ztbg9','VdS 2827:2005-05',1249.0,300,(#3));
#3 = FIRE_LOAD('Holzflachpalette','G. Beilicke S. 116',369.6,1.0,.STCK.,1.0,
     .STOFF_GEGENSTAND.);

// 2.1 FDS-Eingabefehl fuer Waermefreisetzungsrate
&VENT XB = X1,X2,Y1,Y2,Z12,Z12, SURF_ID = 'Versuch'/
&SURF ID = 'Versuch', HRRPUA = 1249.0, RAMP_Q = 'Funktionsverlauf'/

// 2.2 Programmroutine Funktionsverlauf ermitteln
Eingangsparamter: 300,369.6,1.0,.STCK.,1.0
Berechnung nach : Formeln 5.1-5.9
Ergebnis        : t1,t2,t3

// 2.3 FDS-Eingabefehl fuer Funktionsverlauf
&RAMP ID = 'Funktionsverlauf', T = 0.0, F = 0.0/
&RAMP ID = 'Funktionsverlauf', T = t1,  F = 1.0/
&RAMP ID = 'Funktionsverlauf', T = t2,  F = 1.0/
&RAMP ID = 'Funktionsverlauf', T = t3,  F = 0.0/
```

5.1.3. Bereitstellung der Materialdaten

In diesem Abschnitt wird ein Fachmodell beschrieben, das unmittelbar Informationswerte zu Materialangaben im *Fire Dynamics Simulator* bereitstellt. Im Vergleich zu den bisher vorgestellten Modellen (vgl. Abschnitt 5.1.1, 5.1.2) handelt es sich um eine Datenstruktur, in der die Informationsursprünge unberücksichtigt bleiben. Warum? McGrattan et al. beschreiben die Festlegung von thermischen Eigenschaften als eine der größten Herausforderungen bei der Erstellung von Simulationsmodellen. Sie begründen dies einerseits mit der Entwicklung eines Brandereignisses aufgrund umliegender Materialien bzw. deren Eigenschaften. Zum anderen werden Simulationen, auch bei vorhandenen Kenntnissen über physikalische Materialeigenschaften, durch Modellalgorithmen und Auflösungen numerischer Netze beeinflusst (2017, S. 69). „*It is your responsibility to supply the thermal properties of the materials, and then assess the performance of the model to ensure that the phenomena of interest are being captured*“ [McG+17, S. 69]. Die vorliegende Arbeit sieht die Erschließung konkreter Materialangaben beim Ersteller des Simulationsmodells, die beispielsweise auf Fachliteratur oder experimentellen Untersuchungen basieren kann. Die Gemeinsamkeit jeder Recherche liegt in den spezifischen Parametern des FDS, die eine Generierung von Eingabebefehlen erlauben. Eine hierauf ausgelegte Datenstruktur ist für die Verwaltung der Daten sowie deren Variationsmöglichkeiten vorteilhaft und wird nachfolgend erläutert.

Im Quelltext 5.3 ist ein Eingabebefehl für den *Fire Dynamics Simulator* dargestellt, der nach McGrattan et al. typische Angaben zu Materialparametern enthält (2017, S. 72). Neben der Materialdichte werden Werte für eine spezifische Wärmeleitfähigkeit (CONDUCTIVITY) und Wärmekapazität (SPECIFIC_HEAT) angegeben.

Quelltext 5.3: Typische Materialparameter im FDS in einer MATL-Anweisung

```
&MATL ID            = 'INSULATOR'
      CONDUCTIVITY  = 0.041
      SPECIFIC_HEAT = 2.09
      DENSITY       = 229. /
```

McGrattan et al. erweitern die Parameter, in dem sie mithilfe von &RAMP-Anweisungen das Materialverhalten in Abhängigkeit der Temperatur beschreiben. Im Quelltext 5.4 führen sie für die spezifische Wärmeleitfähigkeit und Wärmekapazität Funktionswerte (F) im Bezug auf Temperaturen (T) ein. Mit dem Parameter EMISSIVITY zeigen sie die Möglichkeit auf, den Emissionsgrad von Materialien und damit ein Maß für den Austausch von Wärmestrahlungen zu definieren (2017, S. 73).

Quelltext 5.4: Erweiterte Materialparameter im FDS in einer MATL-Anweisung

```
&MATL ID                 = 'MARINITE'
      EMISSIVITY         = 0.8
      DENSITY            = 737.
      SPECIFIC_HEAT_RAMP = 'c_ramp'
      CONDUCTIVITY_RAMP  = 'k_ramp' /

&RAMP ID='k_ramp', T= 24., F=0.13  /
&RAMP ID='k_ramp', T=149., F=0.12  /
&RAMP ID='k_ramp', T=538., F=0.12  /
&RAMP ID='c_ramp', T= 93., F=1.172 /
&RAMP ID='c_ramp', T=205., F=1.255 /
&RAMP ID='c_ramp', T=316., F=1.339 /
&RAMP ID='c_ramp', T=425., F=1.423 /
```

Bei Simulationen von Brandereignissen dienen die Parameter in den Quelltexten 5.3 und 5.4 im Wesentlichen zur Beschreibung von thermischen Randbedingungen[8]. Hierauf aufbauend zeigen McGrattan et al. ergänzende Anweisungen, welche die Bildung von komplexen Pyrolyse-Modellen erlauben. Die Parametrisierung ermöglicht den Übertrag von Bränden auf einzelne Gegenstände und wird durch den Quelltext 5.5 sowie dessen Kommentierung näher erörtert (2017, S. 84f)[9].

Quelltext 5.5: Materialparameter im FDS für komplexe Pyrolyse-Modelle

```
&MATL ID                    = 'FOAM'
      SPECIFIC_HEAT         = 1.0
      CONDUCTIVITY          = 0.05
      DENSITY               = 40.0
      N_REACTIONS           = 1       --> Anzahl der Reaktionsprodukte
      SPEC_ID               = 'FUEL'  --> Zuordnung Reaktionsmechanismus
      NU_SPEC               = 1.      --> Relative Reaktionsausbeute
      REFERENCE_TEMPERATURE = 350.    --> Temperatur bei Abnahme der Masse
      HEAT_OF_REACTION      = 1500.   --> Eigenschaft Hitzereaktion
      HEAT_OF_COMBUSTION    = 30000./ --> Eigenschaft Verbrennungswaerme
```

Die Abbildung 5.5 bildet eine Datenstruktur für die Bereitstellung von Materialdaten auf Grundlage der vorgestellten Parameter ab. Sämtliche Parameter werden durch definierte Datentypen beschrieben, was deren Aussagekraft innerhalb der Datenstruktur erhöhen soll. Aufgrund des Datentypens *behaviour_select* können Instanzen der Klasse *Material* temperaturabhängige (Temperature_Dependent_Material) und temperaturunabhängige (Temperature_Independent_Material) Parameter annehmen. Für die Beschreibung von temperaturabhängigen Materialangaben sind Objekte der Klassen *Specific_Heat_Ramp* und *Conductivity_Ramp* zu generieren. Jeder dieser Instanzen stehen Attribute für eine Temperaturangabe (t) sowie einen zugehörigen Funktionswert (f) zur Verfügung. An dieser Stelle vermeiden die beiden Datentypen ein mehrfaches Vorsehen gleicher Attribute in den Klassen *Specific_Heat_Ramp* sowie *Conductivity_Ramp*. Eine Instanz der Klasse *Pyrolysis_Material* erbt von der Klasse Material und erhält deren Attribute. Zusätzlich

[8] Die Materialdichte ist hiervon auszuschließen.
[9] Möglichkeit die Brandausbreitung aufgrund vorhandener Brandlasten zu simulieren.

werden Objekte dieser Klasse durch weitere Materialparameter beschrieben, die eine Modellierung von komplexen Pyrolyse-Modellen erlauben. Insgesamt ist der erforderliche Informationsgehalt für die hier fokussierten &MATL-Anweisungen wiedergegeben, was zusätzlich im Quelltext 5.6 veranschaulicht wird.

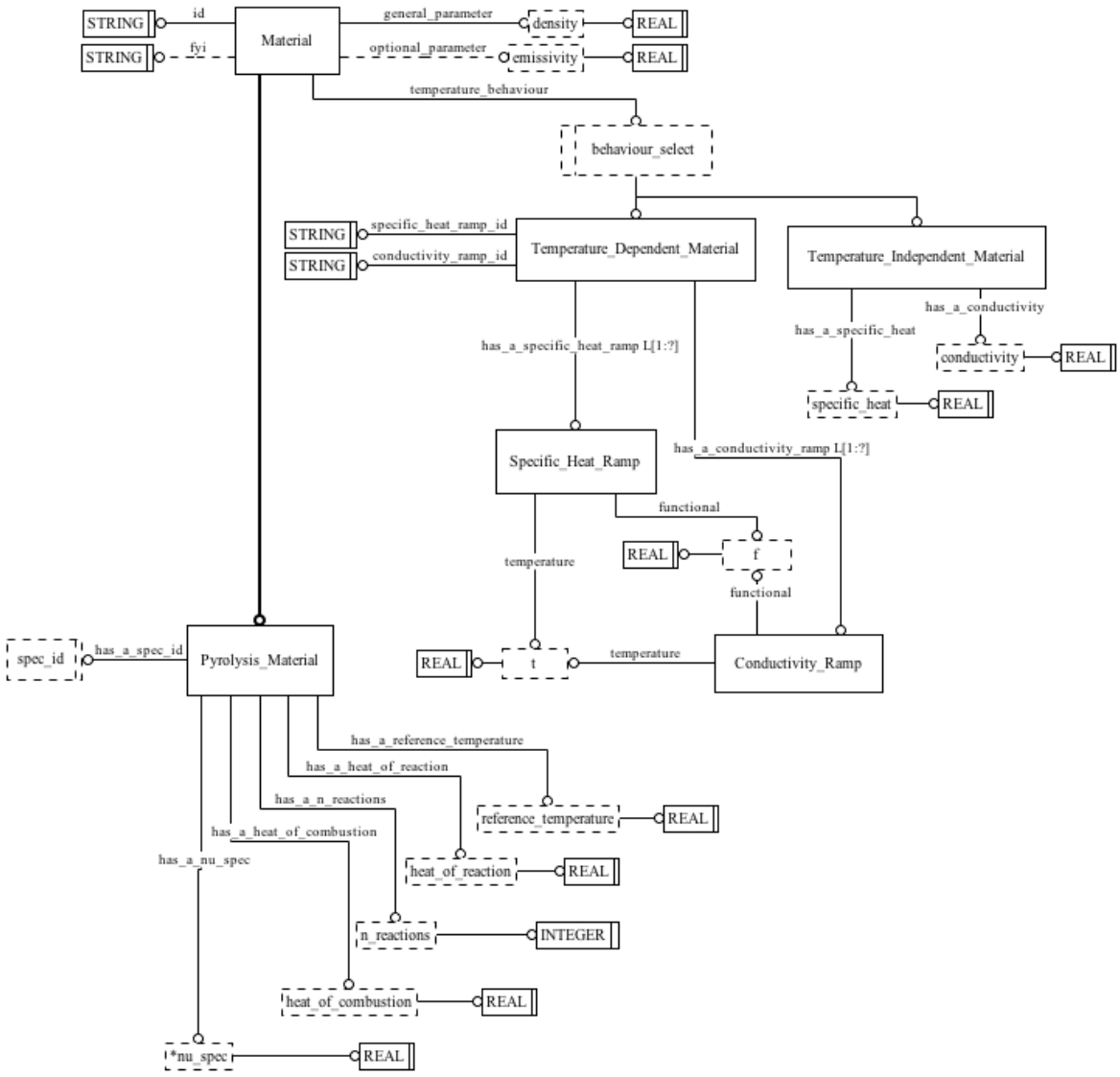

Abb. 5.5.: *EXPRESS*-Datenschema für die Bereitstellung der Materialdaten.

Quelltext 5.6: Informationen für die Bereistellung der Materialdaten

```
// Informationsraum fuer FDS-Eingabebefehl nach Quelltext 5.3
#2  = MATERIAL('sm1_dfepkvz56-e109u','INSULATOR',229.0,#1,$);
#1  = TEMPERATURE_INDEPENDENT_MATERIAL(2.09,0.041);

// Informationsraum fuer FDS-Eingabebefehl nach Quelltext 5.4
#11 = MATERIAL('cm2_evddwfg12-e018z','MARINITE',737.0,#10,0.8);
#10 = TEMPERATURE_DEPENDENT_MATERIAL('c_ramp','k_ramp',
      (#3,#4,#5,#6),(#7,#8,#9));
#3  = SPECIFIC_HEAT_RAMP(93.0,1.172);
#4  = SPECIFIC_HEAT_RAMP(205.0,1.255);
#5  = SPECIFIC_HEAT_RAMP(316.0,1.339);
#6  = SPECIFIC_HEAT_RAMP(425.0,1.423);
#7  = CONDUCTIVITY_RAMP(24.0,0.13);
#8  = CONDUCTIVITY_RAMP(149.0,0.12);
#9  = CONDUCTIVITY_RAMP(538.0,0.12);

// Informationsraum fuer FDS-Eingabebefehl nach Quelltext 5.5
#13 = PYROLYSIS_MATERIAL('pm1_dfepkvz56-e109u','FOAM',40.0,#12,$,1,
      .FUEL.,1.0,350.0,1500.0,30000.0);
#12 = TEMPERATURE_INDEPENDENT_MATERIAL(1.0,0.05);
```

5.2. Bündelung der Informationsräume

Die vorliegende Arbeit fokussiert sich mit den folgenden Abschnitten auf die gezielte Verlinkung von einzelnen Informationsdaten aus den Fachmodellen. Dabei geht sie auf den schematisierten Aufbau von Multimodellen ein und zeigt anhand eines *EXPRESS*-Datenschemas die prinzipielle Machbarkeit des Multimodellansatzes[10]. Im weiteren Verlauf stellt sie ein neutrales Dateiformat für ein Linkmodell vor und zeigt, wie mithilfe von Linkkategorien eine sinnvolle Bündelung von Fachmodellinformationen realisiert werden kann[11]. Auf diese Wiese entsteht ein Multimodell, das eine Datenbasis für die Generierung von FDS-Eingabebefehlen bietet.

5.2.1. Datenstruktur zur Verlinkung der Fachmodelle

Der schematisierte Aufbau eines Multimodells zeigt Anforderungen für die Verlinkung von fachspezifischen Informationen aus eigenständigen Fachmodellen (vgl. Abbildung 2.3). Ein Linkmodell kann aus mehreren Links bestehen, in dem die einzelnen Links mindestens zwei Referenzierungen enthalten. Als wesentliches Kriterium gilt, dass die Verlinkungen Angaben zu vorhandenen Fachmodellen und eindeutigen Identifikatoren aufweisen (vgl. Abschnitt 2.3.1). Diese Voraussetzungen können innerhalb einer Datenstruktur problemlos repräsentiert werden, was konkret die Abbildung 5.6 zeigen soll.

[10] Das *EXPRESS*-Datenschema ist im Anlagenteil einsehbar.
[11] Das Programm für die Erstellung des Linkmodells liegt der Arbeit mit der Compact Disk bei.

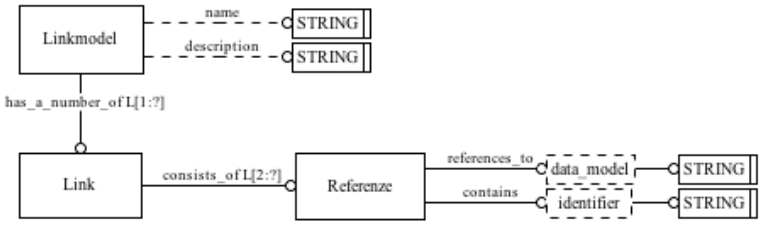

Abb. 5.6.: *EXPRESS*-Datenschema für die Umsetzung der Linkmodelle.

Einer Instanz der Klasse Linkmodel sind Modellname und Beschreibung als optionale Attribute zuweisbar. In der Modellinstanz ist mindestens ein Objekt aus der Klasse Link zu hinterlegen, das sich geringstenfalls durch zwei Instanzen der Klasse *Reference* darstellt. Die Informationen zu den jeweiligen Fachmodellen werden durch die definierten Datentypen *data_model* und *identifier* zugänglich, womit die eingangs beschriebenen Voraussetzungen vollständig umgesetzt sind[12].

Die in Abbildung 5.6 gezeigte Datenstruktur basiert auf einem *EXPRESS*-Schema, das mit der Programmierschnittstelle *JSDAI* [LKS] erstellt und grafisch dargestellt wird. Hieraus erzeugte Instanzen werden in einer *STEP Physikal File* hinterlegt, wobei die Programmschnittstelle neben dem Schreiben auch das Lesen der Dateiinhalte ermöglicht. An dieser Stelle der Arbeit wird auf die zeilenweise Darstellung konkreter Inhalte verzichtet, da sich das prinzipielle Vorgehen aus Abschnitt 5.1 wiederholt.

Folgende Erkenntnis wird gewonnen: Sowohl die Datei (.ifc) des Bauwerksmodells als auch die Dateien (.p21) der vorgestellten Fachmodelle sind *STEP Physikal Files*. Alle Instanzen der Modelle enthalten Identifikatoren und werden damit für ihren Anwendungsbereich eindeutig beschrieben. Das Linkmodell aus Abbildung 5.6 setzt die Voraussetzungen des schematisierten Aufbaus von Multimodellen um und stellt sich durch eine *STEP Physikal File* dar. Insgesamt stehen fünf Dateien zur Verfügung, die durch die Programmierschnittstelle *JSDAI* [LKS] generiert und ausgelesen werden können[13]. Informationsdaten aus den Dateien der Fachmodelle werden zugänglich, wobei das Linkmodell als Bindeglied zwischen den Informationen fungiert. In der Konsequenz entsteht ein Multimodell, das durch eine gezielte Verlinkung der Fachmodellinformationen eine Datenbasis zur Generierung von FDS-Eingabebefehlen erzeugt. Dies erfordert die Entwicklung eines Programms, das die Informationen aus den Fachmodellen filtert und in konkrete Simulationsanweisungen wandelt. Der Umfang der Applikation bestimmt dabei den Detaillierungsgrad von Simulationsmodellen für den *Fire Dynamics Simulator*. Alle Modelle sind austausch- und erweiterbar sowie unabhängig von einzelnen Programmroutinen, was aus Sicht der Arbeit als signifikanter Vorteil gewertet wird[14].

[12] Die definierten Datentypen erhöhen die Aussagekraft innerhalb der Datenstruktur.
[13] Die IFC-Datei wird durch eine CAD/BIM Software erzeugt (Ausnahme).
[14] Ein generisches Verhalten des Programms wird vorausgesetzt.

5.2.2. Neutrales Dateiformat für ein Linkmodell

Im Abschnitt 5.2.1 wird der schematisierte Aufbau von Multimodellen in ein *EXPRESS*-Datenschema übertragen, mit dem die Arbeit die grundsätzliche Realisierbarkeit des Multimodellansatzes aufzeigt. „Im Forschungsprojekt Mefisto wurden [...] neutrale Dateiformate für die Formalisierung der Modellabhängigkeiten in Linkmodellen (linkXML) und für den Austausch der entstandenen Multimodelle in Multimodell-Containern (mmcXML) entwickelt. Mit der einheitlichen Anwendung dieser Formate durch unterschiedliche Fachanwendungen und Projektbeteiligte ergeben sich zahlreiche Möglichkeiten die modellbasierte Zusammenarbeit im Bauprojekt zu verbessern“ [B318]. Damit steht ein neutrales Dateiformat für die Generierung von Linkmodellen zur Verfügung, dessen Datenschema mithilfe der Abbildung 5.7 verdeutlicht wird.

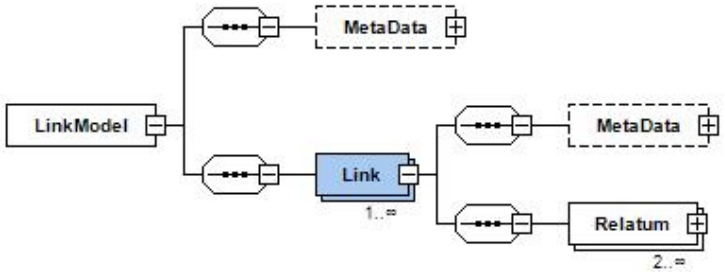

Abb. 5.7.: Grafische Darstellung der *XML Schema Definition LinkModel-2.0.0.xsd* [buib].

Das Datenschema basiert auf einer *XML Schema Definition* (XSD) [buib] und beschreibt somit die Strukturen eines XML-Dokumentes. Jedes hieraus generierte Dokument wird aus dem Wurzelverzeichnis *LinkModel* erzeugt und optional durch Metadaten beschrieben. Einer Instanz aus dem Verzeichnis *LinkModel* können eine Vielzahl an *Links* zugeordnet werden, wobei diese mindestens zwei Elemente aus zwei Anwendungsmodellen verknüpfen. Ein *Relatum* stellt dabei ein Datenelement aus einem beliebigen Anwendungsmodell dar und ist Teil der mehrwertigen Linkbeziehung. Hierfür stehen einem *Relatum* die Attribute *id* und *m* sowie die beiden optionalen Attribute *f* und *r* zur Verfügung[15]. Ein *Relatum* kann wahlweise gegenüber einem anderen *Relatum* bewertet und quantifiziert werden, was in Abbildung 5.8 veranschaulicht wird. Sowohl die damit in Verbindung stehenden *Rates* als auch die gezeigten Annotationen von Metadaten spielen für die Arbeit ein untergeordnete Rolle und bleiben deshalb unberücksichtigt.

[15] Keine explizite Darstellung in Abbildung 5.7.

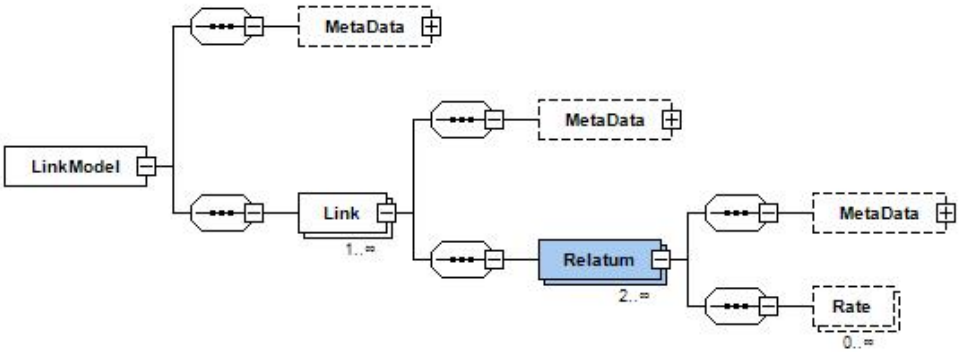

Abb. 5.8.: Bewertung und Quantifizierung eines Relatums im *LinkModel-2.0.0.xsd* [buib].

Insgesamt ist festzustellen, dass der schematisierte Aufbau von Multimodellen abgebildet wird und in den Grundzügen identisch mit der vorgestellten *EXPRESS*-Datenstruktur ist (vgl. Abbildung 5.6). Für den weiteren Verlauf ist die Wahl eines Datenschemas zweitrangig, da beide Schemen die Umsetzung des Multimodellansatzes ermöglichen. Die Arbeit fokussiert das hier vorgestellte Dateiformat und schließt damit die Optionen für eine spätere Vergabe von Metadaten und Rates ein. Neben den bereits vorhandenen *STEP Physikal Files* entsteht eine weitere Datei (XML-Dokument), die Informationen zu den Identifikatoren von Fachmodellen in einzelnen Links hinterlegt. Damit rückt die semantische Bedeutung der Verlinkungen in den Vordergrund, um sinnvoll eine Informationsressource für den *Fire Dynamics Simulator* zu bilden.

5.2.3. Informationsressource für den Fire Dynamics Simulator

Mit der Einführung des FDS-Minimalbeispiels im dritten Kapitel werden, neben grundsätzlichen Funktionalitäten des Simulationsprogramms, auch essenzielle Zusammenhänge für die Verlinkungen der Informationsdaten offengelegt. Beispielsweise sind brennbare Gegenstände von geometrischen Abmessungen und Pyrolysedaten abhängig. Sowohl der Informationsraum der *Industry Foundation Classes* als auch die vorgestellten Fachmodelle stellen durch instanziierte Objekte aus ihren Klassen einzelne Informationsdaten bereit.

Die Tabelle 5.2 unterteilt die Informationen in vier Linkkategorien (LK), mit denen Daten für die Generierung einzelner FDS-Anweisungen sinnvoll gebündelt werden. Ein Link aus der LK1-4 enthält dabei Objekte der dargestellte Klassen, wobei aus einem Fachgebiet nur eine Instanz verlinkt werden kann. Die gezeigten Klassen der jeweiligen Modelle beschränken sich auf den, durch diese Arbeit untersuchten, Informationsraum der *Industry Foundation Classes* sowie der erstellten Fachmodelle[16].

[16] Das Maß an möglichen Linkkategorien wird nur durch den Informationsraum der Fachmodelle und dem Funktionsumfang des Programms bestimmt (Transformation der Informationsdaten in FDS-Anweisungen).

Tabelle 5.2.: Übersicht der Linkkategorien.

Fachmodell/*Modellklassen*	LK1	LK2	LK3	LK4
building model				
IfcColumn	x		x	x
Ifcslab	x		x	x
IfcSpace		x		
IfcWall	x		x	x
IfcWallStandardCase	x		x	x
IfcFurnishingElement		x	x	
IfcBuildingElementProxy	x	x	x	x
fire protection concept				
RC_LB_Component_Request	x			x
Room_Closing_Component_Request	x			x
Load_Bearing_Component_Request	x			x
initial firewall		x		
Load_Controlled_Fire		x		
Fire_Experiment				
material				
Material			x	x
Pyrolysis_Material			x	x

[1] LK1: Verknüpfung der Gebäudeelemente mit Bauteilanforderungen.
[2] LK2: Verknüpfung der Gebäudeelemente mit einer Brandentstehung (Raumnutzung/Einrichtungen).
[3] LK3: Verknüpfung der Gebäudeelemente mit Material- und Pyrolyseangaben für Bauteile.
[4] LK4: Verknüpfung der Gebäudeelemente mit Anforderungen, Material- und Pyrolyseangaben für Bauteile.

Der Quelltext 5.7 zeigt eine Möglichkeit für ein Linkmodell auf Basis der eingeführten *XML Schema Definition* (vgl. Abschnitt 5.2.2). Mithilfe der Programmierschnittstelle *JAXB* [JAX] werden Daten dieses Schemas automatisch an *Java*-Klassen gebunden, was die Generierung von XML-Dokumenten in einer Entwicklungsumgebung ermöglicht (XML-Datenbindung). Alle Links orientieren sich an den eingeführten Linkkategorien. Jedes Relatum besitzt einen Identifikator (id) für ein Fachmodell (m), wobei deren Gesamtheit den vollständigen Informationsraum eines Links repräsentiert.

Quelltext 5.7: Generierung eines Linkmodells gemäß XML Schema Definition

```
<LinkModel formatVersion="2-0-0">
    <Link>
        <Relatum id="1fYCD$XnKhJeI9hiJq13Sm" m="building_model"/>
        <Relatum id="rclb1_265398g5dr4" m="fire_protection_concept"/>
    </Link>
    <Link>
        <Relatum id="1tV0ztwINiHgG7hrZD7239" m="building_model"/>
        <Relatum id="1cf1-dfh98emshe2" m="initial_fire"/>
    </Link>
    <Link>
        <Relatum id="2hWTgkIaY0JB8j4RUlX_LU" m="building_model"/>
        <Relatum id="fe1-sef12ztbg9" m="initial_fire"/>
    </Link>
    <Link>
        <Relatum id="1eTCwoamvkJPqjhvH1AY3V" m="building_model"/>
        <Relatum id="lb1_245788t5wr9" m="fire_protection_concept"/>
        <Relatum id="pm1_dfepkvz56-e109u" m="material"/>
    </Link>
</LinkModel>
```

Eine Filterroutine kann jeden Identifikator (id) und das Fachmodell (m) eines Relatums auslesen. Die Informationen dienen als Eingangsparameter für Routinen, welche in den einzelnen Fachmodellen die ausgelesene ID suchen und weitere Informationen aus der Datenstruktur entnehmen. Dieser Prozess wird abschließend im Quelltext 5.8 für den Link 4 veranschaulicht und zeigt den vollständigen Informationsgehalt für einen FDS-Eingabebefehl[17].

[17] Fachmodellinformationen zur Generierung von FDS-Eingabebefehlen nach den Kapiteln 3, 4 und 5.

Quelltext 5.8: Informationsgehalt aus dem Link 4 mit FDS-Anweisungen

```
// 1. Relatum: id=1eTCwoamvkJPqjhvH1AY3V  m=building_model
-----------------------------------------------------------------------------------------
#13004 = IFCWALLSTANDARDCASE('1eTCwoamvkJPqjhvH1AY3V',#12,'Au\X2\00DF\X0\enwand-004',
         $,$,#12960,#12999,'6874CEB2-930E-6E4D-9D2D-AF94412A20DF',$);

#12960 = IFCLOCALPLACEMENT(#128,#12959);
#12959 = IFCAXIS2PLACEMENT3D(#12957,#12955,#12953);
#12957 = IFCCARTESIANPOINT((10.,4.33229828605,0.));
#12955 = IFCDIRECTION((0.,0.,1.));
#12953 = IFCDIRECTION((0.,-1.,0.));

#12999 = IFCPRODUCTDEFINITIONSHAPE($,$,(#12982,#12988,#12996));
#12988 = IFCSHAPEREPRESENTATION(#457,'Box','BoundingBox',(#12987));
#12987 = IFCBOUNDINGBOX(#12985,1.,0.365,2.7);
#12985 = IFCCARTESIANPOINT((0.,-0.365,0.));

#486   = IFCRELCONTAINEDINSPATIALSTRUCTURE('1aaIejLZM8y90Bvmu5Zi20',#12,$,$,(#471,#657,#2080,
         #2247,#2360,#2519,#7399,#7622,#12499,#12909,#13004,#13099,#23455,#39901),#130);
#130   = IFCBUILDINGSTOREY('2Wk49hkHPC2hvWchm5NASJ',#12,'EG',$,$,#128,$,$,.ELEMENT.,0.);

// 2. Relatum: id=lb1_245788t5wr9  m=fire_protection_concept
-----------------------------------------------------------------------------------------
#7     = LOAD_BEARING_COMPONENT_REQUEST('lb1_245788t5wr9','Stuetzen im EG',
         .FEUERBESTAENDIG.,.F_90AB.,#4,.STUETZE.,.R_90.);
#4     = RGB(205,133,0);

// 3. Relatum: id=pm1_dfepkvz56-e109u  m=material
-----------------------------------------------------------------------------------------
#13    = PYROLYSIS_MATERIAL('pm1_dfepkvz56-e109u','FOAM',40.0,#12,$,1,.PROPANE.,
         1.0,350.0,1500.0,30000.0);
#12    = TEMPERATURE_INDEPENDENT_MATERIAL(1.0,0.05);

// Transformationsvorschrift nach Formeln 4.4
-----------------------------------------------------------------------------------------
X1     = 10.   + (-0.365)                 =  9.635
X2     = 10.   + (-0.365) +   0.365       = 10.000
Y1     = 4.332 -   0.                     =  4.334
Y2     = 4.332 -   0.     -   1.          =  3.332
Z1     = 0.    +   0.     +.  0.          =  0.000
Z2     = 0.    +   0.     +   0.    +  2.7 =  2.700


// Generierung der FDS-Eingabebefehle fuer eine feuerbestaendige Wand mit Materialangaben
-----------------------------------------------------------------------------------------
&OBST XB = 9.635,10.000,4.334,3.332,0.000,2.700, SURF_ID = 'Eigenschaft'/
&SURF ID = 'Eigenschaft', RGB = 205,133,0, MATL_ID = 'FOAM'/
&MATL ID                    = 'FOAM',    // --> frei gewaehlt, nur exemplarisch
      SPEC_ID               = 'PROPANE',
      DENSITY               =    40.000,
      SPECIFIC_HEAT         =     1.000,
      CONDUCTIVITY          =     0.050,
      HEAT_OF_REACTION      =  1500.000,
      HEAT_OF_COMBUSTION    = 30000.000,
      REFERENCE_TEMPERATURE =   350.000,
      N_REACTIONS           =     1,
      NU_SPEC               =     1.000/
```

5.3. Effizienz von Brandsimulationen

Die vorliegende Arbeit beschreibt ein Konzept für die effiziente Durchführung von Brandsimulationen mit dem *Fire Dynamics Simulator*. Sie definiert Effizienz mit einer möglichst hohen Aussagekraft zu einem Brandereignis aufgrund einer fest zur Verfügung stehenden Informationsressource (vgl. Abschnitt 2.1). Konkret: Die Variation der Simulationsparameter führt zu einer Vielzahl an Brandszenarien auf Basis vorhandener Informationsdaten.

Mithilfe des Multimodellansatzes werden Informationen aus den Fachmodellen gezielt verknüpft und für einzelne FDS-Anweisungen verwendet. Exemplarisch zeigt der Quelltext 5.8 den Informationsraum für die Modellierung einer feuerbeständigen Wand. Die Anzahl aller, in einem Linkmodell vorhandenen, Links bildet eine Ressource zur Generierung eines vollständigen Simulationsmodells. Streng genommen spiegelt eine FDS-Eingabedatei dieses Modell wieder. Mit der Einführung des FDS-Minimalbeispiels liegt der Arbeit eine kompilierbare Datei vor, die als Orientierung für die Auslegung von Simulationsmodellen dient (vgl. Anhang A.1).

Das im Abschnitt 2.3.2 vorgestellte Variationsprinzip bietet grundlegend die Möglichkeit, Varianten der Datenmodelle zu erzeugen. In dem Zusammenhang verfolgt die Arbeit den Ansatz, aus einem Grundmodell untergeordnete Simulationsmodelle abzuleiten, was schematisiert die Abbildung 5.9 veranschaulicht. Dabei sind die Modellstrukturen des Grundmodells (basemodel) sowie der Untermodelle (submodel) identisch, wobei die jeweiligen Parameter variieren. Jedes Modell stellt ein Simulationsmodell in Form einer FDS-Eingabedatei, also einer kompilierbaren Textdatei, dar.

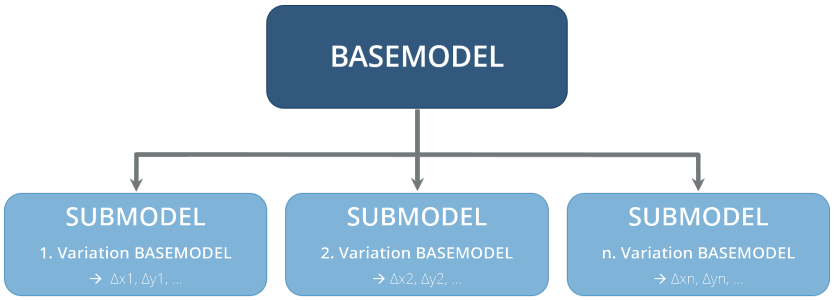

Abb. 5.9.: Schematischer Aufbau von Grund- und Untermodellen.

Die Anwendungsmöglichkeiten des Variationsprinzips für Simulationen mit dem *Fire Dynamics Simulator* wird nachfolgend mithilfe der *XML Schema Definition* (XSD) [Sch+] analysiert. Im weiteren Verlauf der Arbeit werden essenzielle Variationsparameter und ein allgemeingültiges Simulationsmodell vorgestellt.

5.3.1. XML Schema Definition zum Variationsmodell

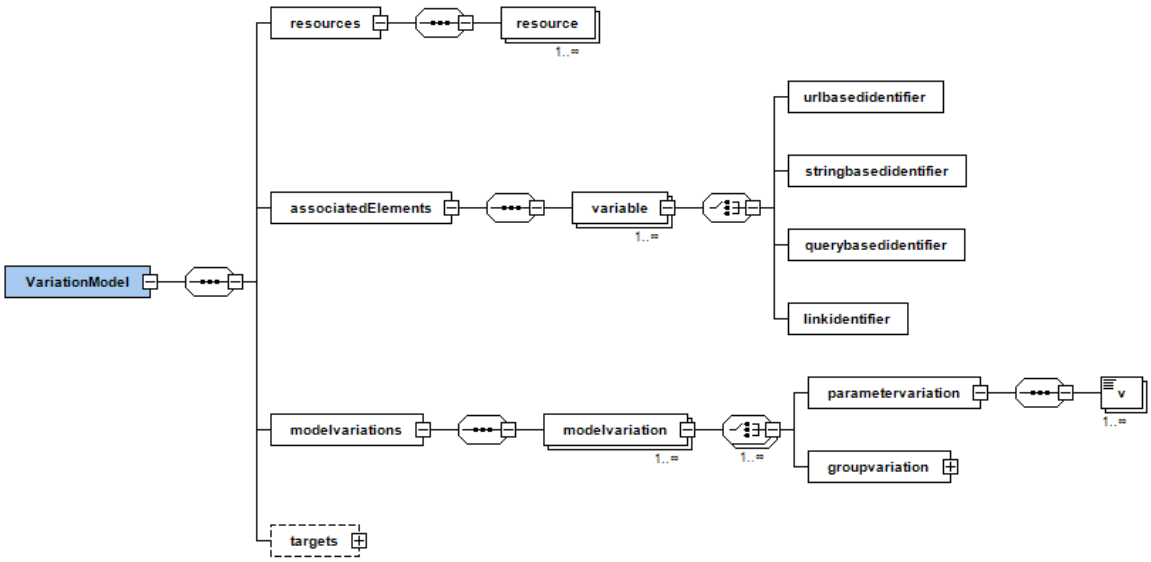

Abb. 5.10.: *XML Schema Definition* zum Variationsmodell [Sch+].

Die Abbildung 5.10 stellt die *XML Schema Definition* (XSD) des Variationsmodells [Sch+], mit Ausnahme der Attribute, grafisch dar. Über *resources* ist einem Element aus dem Verzeichnis *VariationModel* eine Liste von Fachmodellen zuweisbar. Hierzu wird eine *resource* mit einer *id* versehen, die den Namen des jeweiligen Datenmodells enthält. Einer Instanz aus *AssociatedElements* ist mindestens ein Element aus *variable* zuzuordnen, die neben einer eigenen *id* auch den Identifikator zu einer *resource* enthält (resource_id). Über diese Beziehung wird ein Element aus *variable* eindeutig einer *resource*, d.h. einem Daten- bzw. Fachmodell, zugeordnet. Nach Abschnitt 2.3.2 fungieren Variablen lediglich als Zeiger auf Parameter in einem Datenmodell. Hierfür werden in dem Schema mehrere *identifier* vorgesehen, wobei die Arbeit sich an einem *stringbasedidentifier* bedient. Über die Attribute *identifier* und *identifierField* wird eine einseitige Relation zu dem Instanzenparameter in einem Fachmodell hergestellt. In einem Variationsmodell werden damit Angaben zu Informationsquellen hinterlegt und mithilfe von Variablen auf einzelne, in den Quellen vorhandenen, Parametern verwiesen. Durch Elemente von *modelvariations* wird geringstenfalls eine Modellvariation dargestellt. Hier wird unterschieden zwischen Elementen aus den Verzeichnissen *groupvariation* sowie *parametervariation*. Instanzen von *groupvariation* kombinieren Parameter aus den verschiedenen Fachmodellen und berücksichtigen somit eine gleichzeitige Veränderung mehrerer Parameter[18]. Ein Element aus *parametervariation* enthält das Attribut *variableID* und nimmt Bezug auf eine im Variationsmodell deklarierte Variable. Damit besteht eine Verbindung zur Datenquelle und des jeweiligen Parameters. Eine Parametervariation beinhaltet eine Vielzahl an Wertevorgaben (v), die in den Datenquellen seriell als Parameter vorgeben werden.

[18] Gruppenvariationen werden im Rahmen der Arbeit ausgeschlossen.

Eine nach der *XML Schema Definition* (XSD) erzeugte Datenstruktur wird exemplarisch im Quelltext 5.9 gezeigt. Das Variationsmodell beinhaltet Angaben zu den Fachmodellen *initial_fire.p21* und *material.p21* (resource_id). Über *v1_345960dre2* (id) wird eine Variable zugänglich, die Bezug auf das Fachmodell *initial_fire* (resourceID) nimmt. Sie zeigt dabei auf den Parameter *heat_release_rate_per_unit_area* (identifierField) einer Instanz mit der Kennung *lcf1-dfh98emshe2* (identifier). Eine Parametervariation referenziert auf diese Variable und schreibt die Werte (v) 250.0, 350.0 und 450.0 vor.

Quelltext 5.9: Generierung eines Variationsmodells gemäß XML Schema Definition

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<VariationModel>
    <resources>
        <resource id="initial_fire.p21"/>
        <resource id="material.p21"/>
    </resources>
    <associatedElements>
        <variable xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:type="Variable" id="v1_345960dre2" resourceID="initial_fire.p21">
            <stringbasedidentifier identifier="lcf1-dfh98emshe2"
                                   identifierField="heat_release_rate_per_area"/>
        </variable>
        <variable xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                  xsi:type="Variable" id="v1_345960dre2" resourceID="material.p21">
            <stringbasedidentifier identifier="pm1_dfepkvz56-e109u"
                                   identifierField="heat_of_combustion"/>
        </variable>
    </associatedElements>
    <modelvariations>
        <modelvariation namepattern="heat release rate per area value">
            <parametervariation variableID="v1_345960dre2">
                <v xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                   xmlns:xs="http://www.w3.org/2001/XMLSchema"
                   xsi:type="xs:double">250.0</v>
                <v xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                   xmlns:xs="http://www.w3.org/2001/XMLSchema"
                   xsi:type="xs:double">350.0</v>
                <v xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                   xmlns:xs="http://www.w3.org/2001/XMLSchema"
                   xsi:type="xs:double">450.0</v>
            </parametervariation>
        </modelvariation>
    </modelvariations>
</VariationModel>
```

Insgesamt enthält das Variationsmodell drei flächenbezogene Wärmefreisetzungsraten für das Fachmodell *initial_fire*. Diese Informationen ermöglichen, für jede einzelne Werteangabe ein eigenständiges Simulationsmodell zu generieren. Dem ersten Modell wird dabei der Parameterwert 250.0 zugewiesen, dem zweiten 350.0 und dem letzten Modell der Wert 450. Aus Sicht der Arbeit sind hierfür Filterroutinen zu entwickeln, um die Werteangaben in dem XML-Dokument auszulesen und in separate Eingabedateien zu schreiben. Weiterhin ist grundlegend zu hinterfragen, welche Parameter für eine Variation geeignet sind und wann, innerhalb eines Programmablaufes, die Variation zu realisieren ist.

5.3.2. Erkennen der Variationsmöglichkeiten

Die Parametervariation schreibt einzelne Werte für Variablen vor, die jeweils auf einen Parameter eines Datenmodells zeigen. Damit entsteht mit jedem vorgegebenen Wert eine Variation des Modells. In dem Zusammenhang werden aus dem Informationsraum der *Industry Foundation Classes* die Parameter bzw. Attribute X_{DIM}, Y_{DIM}, Z_{DIM} einer *IfcBoundingBox* relevant. Eine Brandlast kann beispielsweise durch einen Einrichtungsgegenstand (IfcFurnishingElement) oder durch eine beliebige modellierte Geometrie (IfcBuildingElementProxy) dargestellt werden[19]. Beide Klassen beschreiben die Abmessungen ihrer Instanzen durch eine *BoundingBox*, womit Brandereignisse beeinflusst werden können. Zum Beispiel verringert sich mit einer Erhöhung der Brandlastbreite der Abstand zu brennenden Gegenständen, was die Brandausbreitung begünstigt. Es ist zwingend zu berücksichtigen, dass die Variation der Gebäudeelemente die Möglichkeit für geometrische Überschneidungen in der Bauwerksstruktur zulässt[20].

Die Tabelle 5.3 enthält eine Auflistung an Parametern, deren Variation maßgebend ein Brandereignis beeinflussen. Das Fachmodell *initial fire* gibt dabei Werte vor, welche die maximale Wärmefreisetzung und die Branddauer beschreiben. Die angegeben Werte des Materialmodells hingegen, ob und wie intensiv sich umliegende Gegenstände an einem Brandereignis beteiligen.

Tabelle 5.3.: Übersicht der Variationsparameter in den Fachmodellen.

Fachmodell	Parameter	Beschreibung
initial fire	heat release rate per unit area	Wärmefreisetzungsrate
	fire development time	Brandentwicklungszeit
	fire load density	Brandlastdichte
	quantity	Menge einer Brandlast
	calorific value	Heizwert einer Brandlast
material	reference temperature	Temperatur bei Abnahme der Masse
	heat of reaction	Hitzereaktion
	heat of combustion	Verbrennungswärme
	spec id	Reaktionsmechanismus

[19] Voraussetzung ist die Verlinkung der Instanz mit Pyrolyseparametern aus dem Materialmodell.
[20] Geometrische Überschneidungen in der Bauwerksstruktur sind zu prüfen und auszuschließen.

Neben dem Erkennen von Variationsparametern ist zu hinterfragen, auf welche Weise ein Variationsmodell mit dem Informationsraum eines Multimodells in Einklang gebracht werden kann. Die Arbeit hält an dieser Stelle fest: Es liegen insgesamt vier voneinander unabhängige Fachmodelle bzw. Datenquellen vor. Mithilfe eines Linkmodells werden Informationen aus den Modellen verbunden (Multimodellansatz). Ein Variationsmodell enthält konkrete Werte für Parameter in den einzelnen Datenquellen und schreibt diese dem jeweiligen Modell vor (Variationsprinzip). Sowohl das Link- als auch das vorgestellte Variationsmodell bedienen sich an den Informationsräumen der einzelnen Fachmodelle. Für die Kombination der beiden Modelle verfolgt die vorliegende Arbeit zwei Ansätze, die nachfolgend näher erläutert werden[21].

Der erste Ansatz besteht in dem Erzeugen von mehreren Varianten der Fachmodelle. Hierzu sind die Informationen aus dem Variationsmodell auszulesen und für jede einzelne Parameterangabe eine Variante des Fachmodells in Form einer eigenen Datei zu generieren. In jeder der Dateien bleiben die Identifikatoren der Instanzen unverändert, da lediglich die Parameter bzw. Attribute variiert werden. Die mehrmalige Anwendung des Linkmodells ermöglicht, die Informationen für jede einzelne Datei zu bündeln. Im Gegensatz zu dem Linkmodell findet das Variationsmodell ein einziges Mal Anwendung. Liegen beispielsweise drei variable Werte einer Wärmefreisetzungsrate aus dem Fachmodell *initial fire* vor, resultieren daraus drei eigenständige Dateien (initial_fire.p21). Die Informationen der übrigen Fachmodelle bleiben unverändert, sodass deren Dateien für die Auswertung des Linkmodells zur Verfügung stehen. Die Variation mehrerer Parameter aus unterschiedlichen Fachmodellen (groupvariation) führt in der Konsequenz zu einer Vielzahl an Dateien der jeweiligen Datenquellen. Bei Umsetzung des Ansatzes erhöht sich die Durchlaufzeit einer Programmroutine um ein vielfaches, da das Linkmodell für jeden einzelnen Variationsschritt angewendet wird[22].

Ein weiterer Ansatz wird mit der einmaligen Auswertung des Linkmodells verfolgt. In einem ersten Schritt werden alle Fachmodellinformationen für jeden einzelnen Link im Linkmodell ausgewertet. Der Informationsgehalt wird in ein eigenständiges Datenmodell übertragen, was als einzige Ressource dem Variationsmodell hinzugefügt wird. Das Modell repräsentiert das Simulationsmodell für den *Fire Dynamics Simulator* und stellt die Grundvariante für das Brandereignis dar. Alle Variablen im Variationsmodell beziehen sich ausschließlich auf Parameter in diesem Grundmodell. Der Ansatz zeigt jedoch einen wesentlichen Nachteil auf: Das Variationsmodell kann erst erzeugt werden, wenn das Grundmodell existiert, was zu Zielkonflikten innerhalb eines Programmablaufs führen kann. Vorteilhaft ist, dass die Variationen sich ausschließlich auf ein einziges Datenmodell beschränken und damit die Vielzahl an Fachmodelldateien umgangen wird.

[21] Die vorliegende Arbeit fokussiert sich auf die Variation einzelner Parameter.
[22] Mehrmaliges Durchlaufen der Filterroutinen in dem jeweiligen Fachmodell.

Insgesamt zeigt die vorliegende Arbeit damit essenzielle Variationsmöglichkeiten auf, wobei sie sich an der *XML Schema Definition* (XSD) des Variationsprinzips [Sch+] bedient. Die hier gezeigten Parameter sollen eine grundlegende Basis für Modellvarianten zu einzelnen Brandszenarien bieten. Mit den vorgestellten Ansätzen ist eine Programmentwicklung möglich, die das Variationsmodell mit dem Multimodellansatz vereint. Die Gemeinsamkeit beider Ansätze besteht in einem übergeordneten Simulationsmodell, was in dem nachfolgenden Abschnitt vorgestellt wird.

5.3.3. Simulationsmodell für den Fire Dynamics Simulator

Die Eingabedatei für den *Fire Dynamics Simulator* stellt das Simulationsmodell dar und besteht aus der gezielten Zusammenführung von FDS-Anweisungen. In dem Zusammenhang zeigt die Arbeit, wie Anweisungen aus der Kombination von Informationsdaten einzelner Fachmodelle generiert und variiert werden können. Im Hinblick der Programmumsetzung dieses Konzepts ist die Entwicklung einer Datenstruktur empfehlenswert, welche die FDS-Eingabebefehle in die Struktur einer Eingabedatei überführt. Mit der Programmierschnittstelle *JSDAI* [LKS] wird ein Datenschema entwickelt, das sich an dem Aufbau der FDS-Eingabedatei aus Kapitel 3 orientiert[23].

Die Abbildung 5.11 zeigt einen Ausschnitt des *EXPRESS*-Datenschemas, wobei das Schema vollständig im Anlagenteil einzusehen ist (vgl. Anlage A.7). Das Simulationsmodell beinhaltet vier elementare Bestandteile: Gebäudeelemente, eine Instanz zur Beschreibung einer Brandentstehung, brennbare Gegenstände sowie ein Simulationsprofil. Die Grundidee der Datenstruktur ist es, einzelne &-Anweisungen in eigene Klassen zu hinterlegen. Dadurch wird die mehrfache Nutzung der Anweisungen für unterschiedliche Anwendungsfälle ermöglicht. Zum Beispiel wird durch die Instanz der Klasse *Fire_Development* eine Brandentstehung modelliert, die sich mindestens aus einer &OBST-, &VENT- und &SURF-Anweisung zusammensetzt (vgl. Quelltext 3.3)[24]. Eine &OBST-Anweisung dient jedoch auch zur Beschreibung von Gebäudeelementen und muss demnach für Klasse *Buildung_Element* zugänglich sein. Die definierten Datentypen erlauben die mehrfache Verwendung von Attributen, was beispielsweise an der &HOLE-Anweisung verdeutlicht werden kann: Eine Instanz aus *Building_Element* stellt sich entweder durch eine &OBST- oder als &HOLE-Anweisung dar. Beide Befehle beziehen den Parameter XB und beschreiben damit eine Zahlenfolge aus insgesamt sechs Werten des Datentypen REAL.

[23] Programmierschnittstelle frei gewählt.
[24] Keine Darstellung in Abbildung 5.11, vgl. Anlage A.7

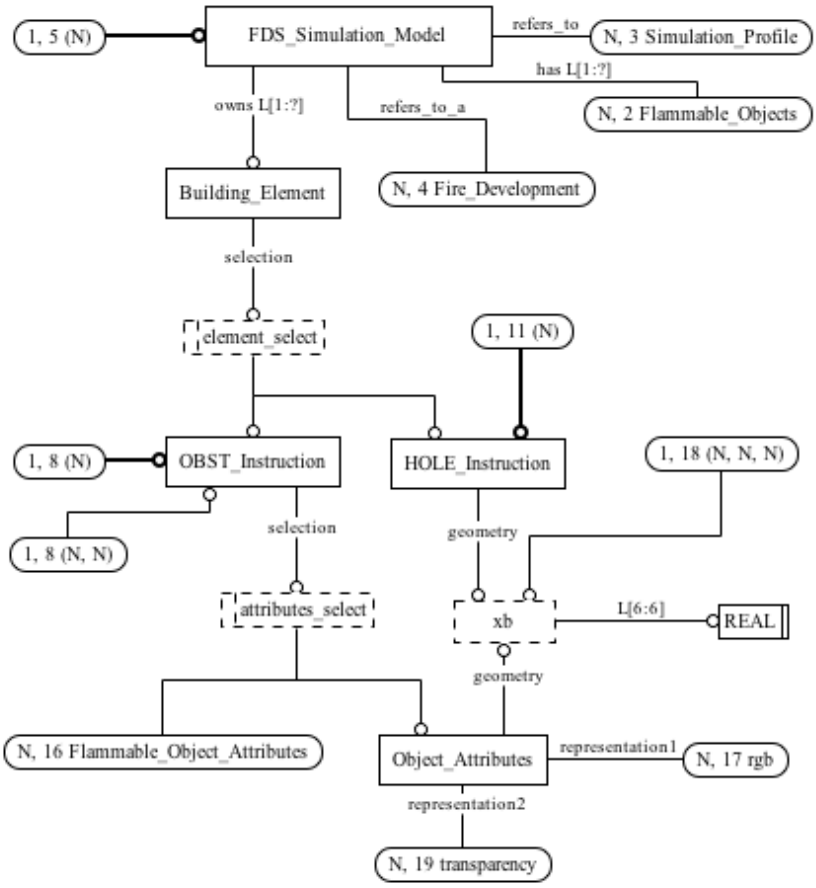

Abb. 5.11.: *EXPRESS*-Datenschema für das FDS-Simulationsmodell (Ausschnitt).

Insgesamt ermöglicht die Datenstruktur, die Generierung übergeordneter Elemente durch die gezielte Zusammensetzung einzelner FDS-Eingabebefehle. Das Schema ist problemlos mit den Anweisungen der Tabellen 18.1-32 [McG+17, S. 265-288] erweiterbar, da jeder Parameter unmittelbar einer FDS-Anweisung zugeordnet werden kann. Im Hinblick auf die Programmierung werden durch das *EXPRESS*-Datenschema *Java*-Klassen gebunden, die eine Hilfestellung bei der Erstellung von Programmroutinen bzw. dem Generieren einer kompilierbaren Eingabedatei bieten. Das Simulationsmodell ist damit auch in eine *STEP Physikal File* überführbar, welche die vollständige Informationsressource für die Simulation eines Brandereignisses mit dem *Fire Dynamics Simulator* repräsentieren kann. Schlussendlich stellt das Schema die sinnvolle Zusammensetzung der einzelnen FDS-Eingabebefehle dar und ermöglicht die Generierung der Eingabedateien.

5.4. Zusammenfassung der Erkenntnisse

Dieses Kapitel beschreibt insgesamt einen konzeptionellen Ansatz, der ein Multimodell für die Simulation mit dem *Fire Dynamics Simulator* ermöglichen soll. Ergänzend erlaubt die Implementierung des Variationsprinzips die Bildung von Varianten für ein Brandereignis. Die Abbildung 5.12 fasst die Erkenntnisse grafisch zusammen und zeigt gleichermaßen eine Möglichkeit auf, die theoretischen Überlegungen zu verifizieren.

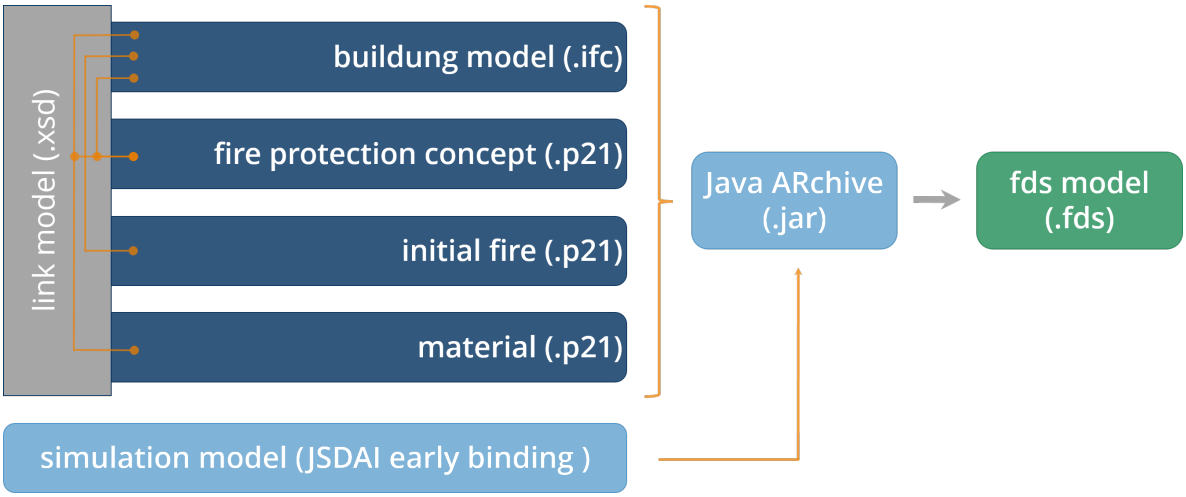

Abb. 5.12.: Das Multimodell für Simulationen mit dem *Fire Dynamics Simulator*.

Die Fachmodelle werden in einer *STEP Physikal File* hinterlegt. Durch die Überführung der Bauwerksgeometrie in die *Industry Foundation Classes* (IFC) sind Informationsdaten zu einzelnen Gebäudeelementen zugänglich. Die Informationen beziehen sich dabei auf geometrische Abmessungen und werden durch drei weitere Datenmodelle ergänzt. Ein Brandschutzkonzept (fire protection concept) stellt eine eigenständige Fachplanung dar und liefert Informationen zu Bauteilanforderungen in einem Gebäude. Für die Modellierung von Entstehungsbränden nähert sich die Arbeit mithilfe der Ingenieurmethoden des Brandschutzes an und stellt hierfür eine eigene Datenstruktur bereit (initial fire). Weiterhin geht ein Materialmodell auf spezifische Simulationsparameter für Materialangaben ein. Die Gemeinsamkeit aller Modelle besteht in der eindeutigen Identifizierungsmöglichkeit von Elementen durch Identifikatoren (ID). Zusammen mit den Kenntnissen über die jeweiligen Datenstruktur können Informationen aus jedem Anwendungsbereich gefiltert werden. In dem Linkmodell werden Elemente lediglich durch die Angaben ihres Modells sowie ihrer ID hinterlegt und gezielt zu eigenen Informationsräumen gebündelt. Auf diese Weise entsteht eine neue Informationsressource für die Generierung von einzelnen FDS-Eingabebefehlen, womit der Multimodellansatz vollständig Anwendung findet.

Ein *simulation model* stellt die Datenstruktur für die Zusammenführung von Simulationsbefehlen bereit und repräsentiert den Aufbau einer kompilierbaren FDS-Eingabedatei (fds model). Die vorliegende Arbeit betrachtet diese Datei als Simulationsmodell und

sieht sowohl für die Realisierung als auch Verifizierung der Ansätze ein *Java*-Programm (*Java ARchiv*) vor. Die Erstellung der Daten erfolgt hierbei durch frühzeitiges Binden der *Java*-Klassen, womit die jeweiligen Modellklassen im Code verwendet werden (early binding)[25].

In der Abbildung 5.13 wird das Konzept durch das Variationsmodell erweitert[26]. Sie zeigt drei Variationen von *initial fire*, die sich jeweils durch eine separate Datei darstellen. In den Dateien existieren jeweils unterschiedliche Werte für den Parameter *heat release rate*, wodurch die Wärmefreisetzungsrate in dem Anwendungsmodell variiert wird. In der Konsequenz ist das Linkmodell mehrmalig auszuwerten, sodass insgesamt drei Varianten für ein Brandereignis vorliegen und simuliert werden können.

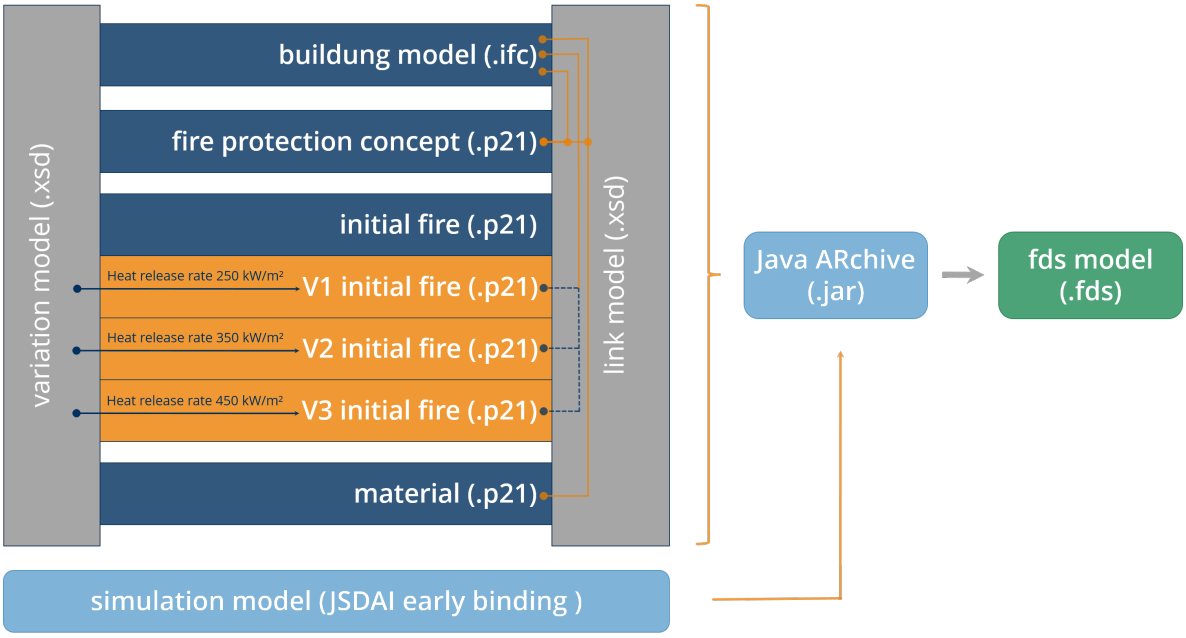

Abb. 5.13.: Das Multimodell unter Ausnutzung des Variationsprinzips.

Alle vorgestellten Modelle weisen ein generisches Verhalten auf, wodurch der Informationsgehalt erweiterbar ist. Damit liegt ein allgemeingültiges Konzept vor, das durch ein Programm verifiziert werden kann. Kapitel 6 nimmt diese Herausforderung an und zeigt die prinzipiellen Anwendungsmöglichkeiten.

[25] Die Vorgabe einer Datenstruktur reduziert die Fehleranfälligkeit.
[26] Im Abschnitt 5.3.2 beschreibt die Arbeit zwei grundsätzliche Möglichkeiten für die Implementierung des Variationsprinzips. Nachfolgend wird der erste Ansatz dargestellt, der für die Konzeptumsetzung bzw. hinsichtlich einer Programmentwicklung empfohlen wird.

6. Programmentwicklung

Für das Können gibt es nur einen Beweis: das Tun.

(Marie von Ebner-Eschenbach)

6.1. Vorbemerkung

Im Abschnitt 2.1 wird für die Konzeptumsetzung ein Programm vorgesehen, das ein Multimodell anwendet und kompilierbare FDS-Eingabedateien erzeugt. Die vollständige Berücksichtigung aller in der Arbeit gezeigten Datenmodelle, führt in der angedachten Bearbeitungszeit zu einem Zielkonflikt. Einerseits ist dies mit Erstellung von Filter- und Transformationsroutinen sowie den damit verbundenen Funktionsprüfungen zu begründen. Andererseits reicht eine rudimentäre Lösung aus, um die prinzipielle Machbarkeit der vorgestellten Ansätze zu zeigen bzw. diese zu verifizieren.

In dem Zusammenhang wird für vorliegende Arbeit festgelegt: Das Programm verifiziert das Multimodell nach Abbildung 5.12, da der Ansatz die Integration des Variationsmodells bestimmt[1]. Datenelemente aus dem Informationsraum der *Industry Foundation Classes* sind Bestandteil jeder Linkkategorie, womit die Übertragung der Bauwerksgeometrie vorauszusetzen ist (vgl. Abschnitt 5.2.3). Mit der erfolgreichen Transformation, von IFC-Daten in eine kompilierbare FDS-Eingabedatei, kann problemlos auf die Implementierung weiterer Datenmodelle geschlossen werden. Die vorliegende Arbeit fokussiert sich demnach ausschließlich auf den Zugriff und die Transformation der Informationen aus einem Bauwerksmodell. Die Erstellung der Datenmodelle wird vollständig in den Kapiteln 4 und 5 veranschaulicht, womit das Programm ein vorhandenes Linkmodell sowie eine IFC-Austauschdatei als gegeben voraussetzt. Alle Programmroutinen und Datenmodelle sind mit der Compact Disk der Arbeit beigelegt.

6.2. Beschreibung der Programmabläufe

Der Programmablauf setzt ein Linkmodell (LinkModel200.xml) sowie eine IFC-Austauschdatei (buildung_model.ifc) in einem lokalen Ordnerverzeichnis voraus. In einem ersten Schritt werden dem Linkmodell Informationen entnommen, was schematisiert die Abbildung 6.1 veranschaulicht. Ein dreidimensionaler *Array*, nachfolgend als *LinKCubE* bezeichnet, dient dabei der Speicherung sowie dem systematischen Zugriff von bzw. auf Modellinformationen.

Der Zugriff (ACCESS) auf das XML-Dokument erfolgt mit der Programmierschnittstelle *DOM* [DOM], die das Auslesen (SEARCH) der Links bzw. Relata ermöglicht. Einem existierenden Relatum werden die beiden Attribute *data model* und *id* entnommen (GET) und dem *LinKCubE* zugewiesen. Die Bezeichnung des Datenmodells wird auf das Feld *LinKCubE[link][0][relatum]*, die ID hingegen auf *LinKCubE[link][1][relatum]* gelegt. Da sich mindestens ein Link und zwei Relata in einem Modell befinden, durchlaufen die beiden

[1] Das Variationsmodell stellt ein Hilfsmodell mit Vorgaben für Parametervariationen dar und setzt keine Informationsdaten zueinander in Verbindung. Die Verifizierung des Multimodells lässt auf Erweiterungsmöglichkeiten von Informationen durch zusätzliche Hilfsmodelle schließen, sodass von einer Programmimplementierung des Variationsprinzips abgesehen wird.

Filterroutinen eine Programmierschleife. Im *LinKCubE* werden die aktuellen Suchdurchläufe mit *[link]* bzw. *[relatum]* gekennzeichnet. Der dreidimensionale *Array* dient dem Programmverlauf als interner Informationsspeicher, welcher die Bezeichnungen der Datenmodelle sowie die jeweiligen Identifikatoren (ID) für alle Links enthält. Zur besseren Veranschaulichung wird der *LinKCubE* in Abbildung 6.1 dargestellt. Die Notwendigkeit für einen internen Informationsspeicher ist kritisch zu hinterfragen, da alle Informationen mit dem Linkmodell bereits vorliegen. Innerhalb der Programmentwicklung erweist sich ein Verschachteln von Datenabfragen als nicht zielführend, womit das hier vorgestellte Verfahren favorisiert wird.

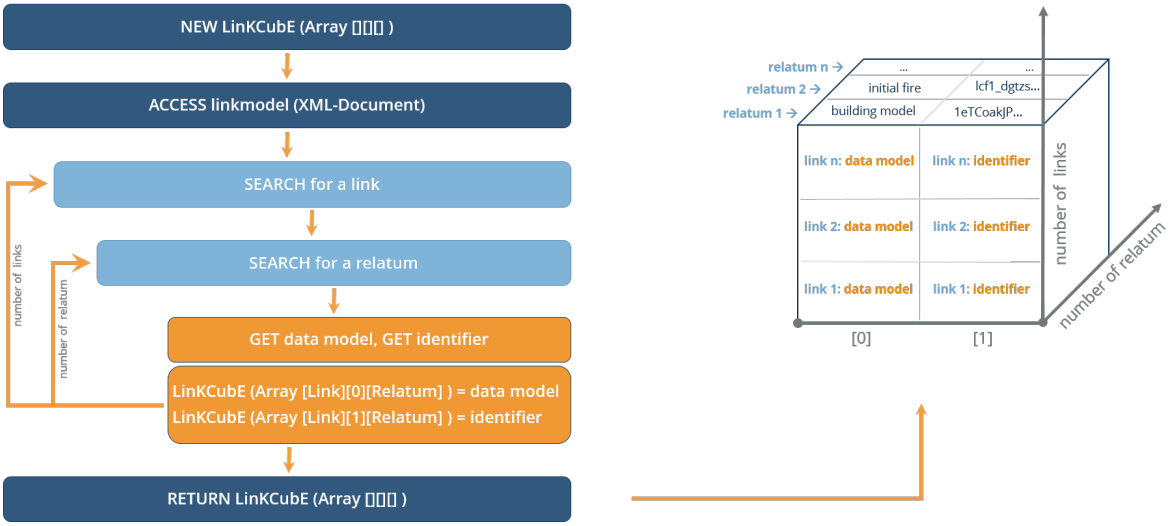

Abb. 6.1.: Auswertung eines Linkmodells mit dem *LinKCubE*.

Die Abbildung 6.2 zeigt den weiteren Programmverlauf für die systematische Abfrage einzelner Linkinformationen. Der *LinKCubE* dient dabei als zentrale Eingangsgröße und steht dem Programm global zur Verfügung (GET). In einer Programmierschleife erfolgt für jeden Link eine Auswertung der abgelegten Informationen (INTERPRETATION). An dieser Stelle wird ein weiterer *Array* als Informationsspeicher eingeführt, der nachfolgend als *LinKBuffeR* bezeichnet ist. Die Grundidee besteht darin, alle Informationen aus den Datenmodellen in einem Datenbündel zu hinterlegen. Das Vorgehen orientiert sich an dem Quelltext 5.8 und soll die Generierung von FDS-Anweisungen ermöglichen.

In einer weiteren Programmierschleife folgen Filterroutinen für die Abfrage einzelner Informationsdaten. Über *LinKCubE[link][0][relatum]* liegt dem Programm der Name des Datenmodells vor, wodurch eine Unterroutine für das Modell aufgerufen werden kann. Hierin wird die Instanz mit der ID gesucht, die dem Wert im Feld *LinKCubE[link][1][relatum]* entspricht. Mit der Kenntnis über die jeweiligen Datenstrukturen können Informationen gezielt abgefragt (SEARCH) und im *LinKBuffeR* abgelegt werden. Vererbungshierarchien reduzieren dabei die Anzahl notwendiger Filterroutinen. Für die Gebäudeelemente werden zum Beispiel Datenabfragen hinsichtlich der *IfcElement* ausgelegt. Damit werden

Informationsdaten zu Decken, Wänden und weiteren Bauteilen in einer einzigen Routine zugänglich. Ein weiterer Vorteil besteht darin, dass beispielsweise Objekte aus der *IfcDoor* ebenfalls von *IfcElement* erben und innerhalb der Programmroutine berücksichtigt werden können.

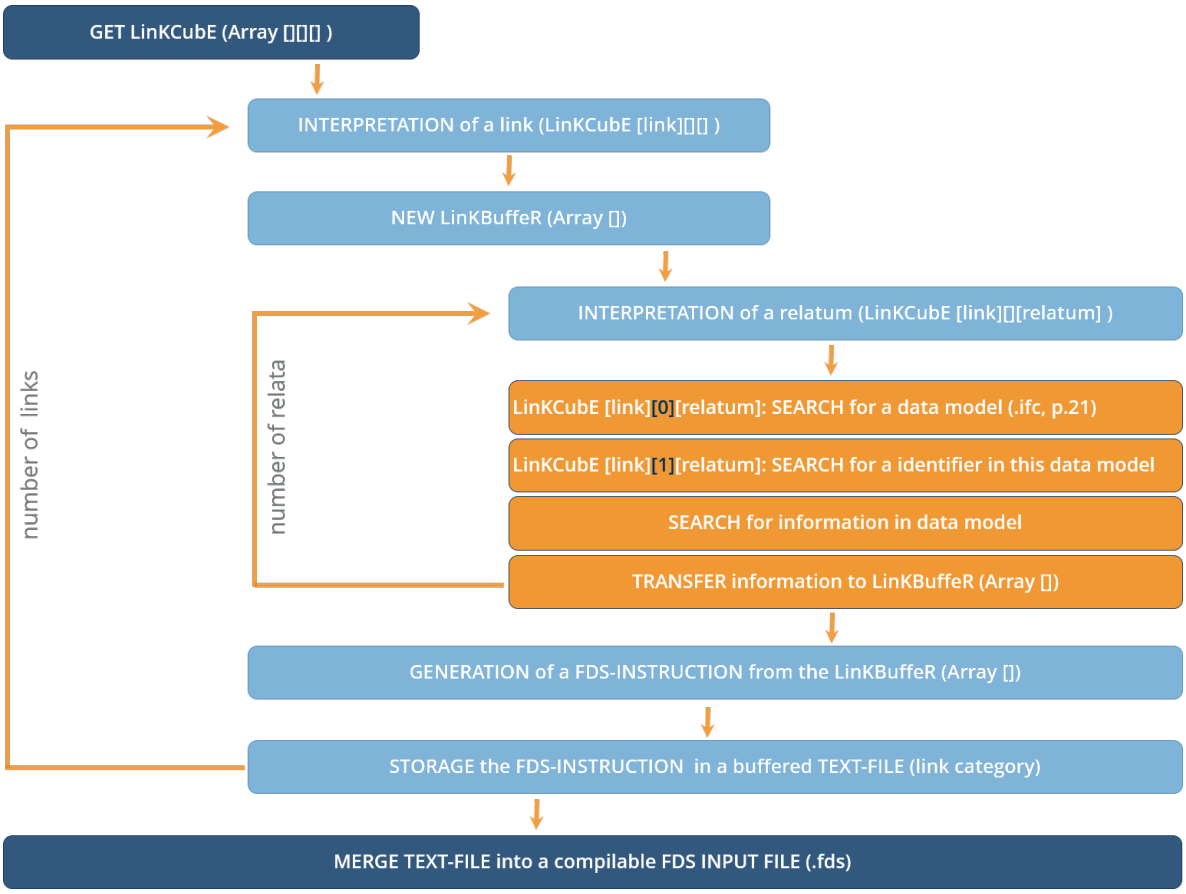

Abb. 6.2.: Auswertung der Fachmodellinformationen mit dem *LinKBuffeR*.

Nachdem alle Relata die Schleife durchlaufen haben, wird für den Link eine FDS-Anweisung erstellt (GENERATION) und diese in eine Textdatei hinterlegt (STORAGE). Eine FDS-Anweisung wird mithilfe des vorgestellten *simulation model* generiert (vgl. Abschnitt 5.3.3)[2]. Mit der Textdatei soll eine Linkkategorie repräsentiert werden, wobei diese für jeden Link sequentiell fortgeschrieben wird. Durch die vorgesehenen Linkkategorien entstehen vier Textdateien, die zu einer FDS-Eingabedatei zusammen geführt werden (MERGE)[3]. Der Ansatz ist notwendig, da Links in einem Linkmodell beliebig abgelegt werden können, die Struktur der Eingabedatei jedoch zwingend einzuhalten ist[4]. In dem Ordnerverzeichnis wird eine kompilierbare FDS-Eingabedatei hinterlegt. Insgesamt erlaubt das Programm die Verifizierung der vorgestellten Ansätze, was mithilfe der in Kapitel 4 eingeführten Gebäudeelemente gezeigt wird.

[2] Im Programm wird ein modifiziertes Datenschema angewendet (Vereinfachung).
[3] Bei vollständiger Umsetzung des Programms.
[4] Das Simulationsprofil nach Abschnitt steht beispielsweise am Anfang der Eingabedatei.

6.3. Verifikation der vorgestellten Ansätze

Das Kapitel 4 führt Gebäudeelemente ein, welche die Verifikation des Multimodells nach Abbildung 5.12 hinsichtlich einer Bauwerksgeometrie ermöglichen. Die IFC-Austauschdatei (building_model.ifc) wird hierzu in das Ordnerverzeichnis des Programms gelegt. Um den Ansatz verifizieren zu können, wird das Verzeichnis durch die vorgestellten Fachmodelle erweitert (vgl. Kapitel 5)[5]. In dem Linkmodell (linkmodel200.xml) werden Links mit mindestens zwei Relata nach den eingeführten Linkkategorien eingetragen (vgl. Tabelle 5.2). Insgesamt enthält das Ordnerverzeichnis fünf Dateien, die dem Programm zur Verfügung stehen.

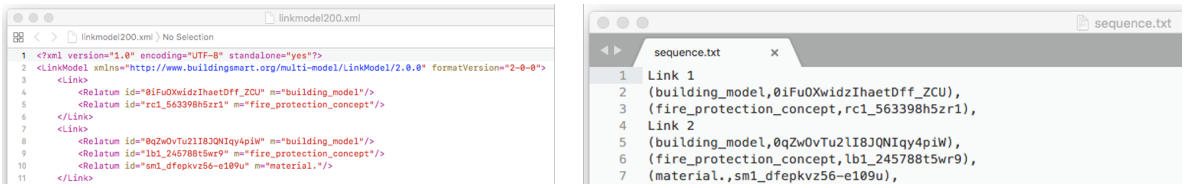

Abb. 6.3.: Datenübertrag aus dem Linkmodell in den *LinKCubE*.

Die Abbildung 6.3 stellt den Datenübertrag aus dem Linkmodell in den Informationsspeicher (*LinKCubE*) dar. Links wird ein Ausschnitt der XML-Datei (linkmodel200.xml) gezeigt. Der erfolgreiche Übertrag in den *LinKCubE* wird in dem rechten Teil der Abbildung veranschaulicht. Zu sehen ist eine programminterne Kontrollsequenz, die für jeden Link die ausgelesenen Informationen anzeigt. Für den weiteren Programmablauf liegen somit einzelne Informationsbündel (data model, identifier) vor. Der Ansatz des Multimodells ist bestätigt, da aus der *XML Schema Definition* [buib] ein Linkmodell erstellt ist und die Daten von dem Programm ausgelesen werden können. Mithilfe von Filter- und Transformationsroutinen werden Informationen zu Gebäudeelementen dem Bauwerksmodell entnommen. Es entsteht eine FDS-Eingabedatei, mit der sich ein Ergebnis nach Abbildung 6.4 einstellt.

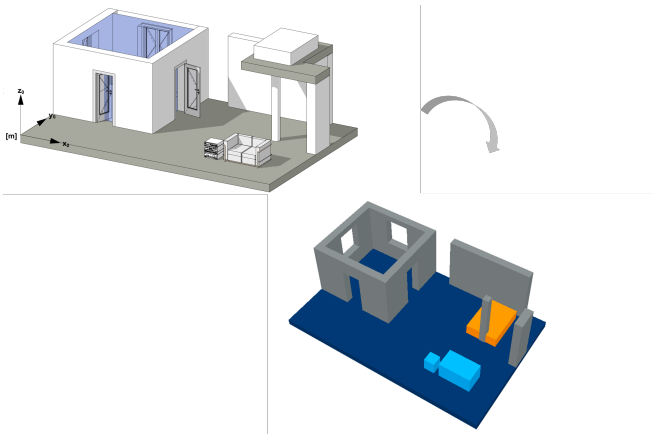

Abb. 6.4.: Ergebnis der FDS-Eingabedatei im Vergleich zu der Bauwerksmodellierung.

[5] fire_protection_concept.p21, initial_fire.p21, material.p21

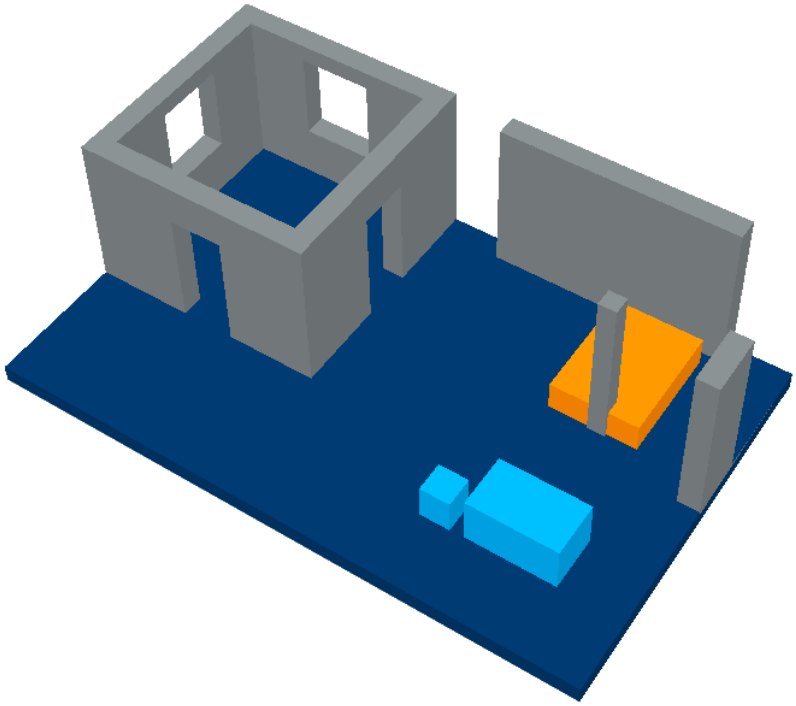

Abb. 6.5.: Ergebnis der FDS-Eingabedatei.

Die Abbildung 6.5 zeigt das grafische Transformationsergebnis in das entstandene FDS-Simulationsmodell. Die mit der CAD/BIM Software [Gra] erstellten Gebäudeelemente werden gegenübergestellt, um einen Vergleich der beiden Modelle zu ermöglichen. Was ist zu sehen? Mit Ausnahme des L-förmigen Deckenteils werden alle Elemente in das Simulationsmodell übertragen, wobei die Abmessungen und Platzierungen eins zu eins übernommen werden[6]. Eine Ausnahme bildet die Brandlast, die sich auf einem falschen Höhenniveau befindet[7]. Der Grund hierfür besteht in einer fehlenden Filterroutine, die eine inverse Beziehung für die Höhenlage berücksichtigt (vgl. Abschnitt 4.2.2). Damit werden in dem Informationsspeicher *LinKBuffeR* keine Angaben zu den jeweiligen Bezugshöhen der Bauteile hinterlegt, wodurch sich alle Gebäudeelemente auf $0.0\,m$ beziehen. Mit dieser Erkenntnis kann auch das fehlende Deckenteil begründet werden. In Abbildung 6.6 wird hierfür die Bauwerksgeometrie über die Außenkanten dargestellt. Unterhalb der Brandlast befindet sich ein weiterer Quader, der die *BoundingBox* des Deckenteils repräsentiert. Das Deckenteil liegt demnach in der unteren Decke.

In dem Zusammenhang ensteht ein Konflikt bei der Zuweisung von Materialparametern. Die Maße einer *BoundingBox* beziehen sich auf die äußeren Bauteilabmessungen. Folglich überschneiden sich die Bauteile in den Eckbereichen. Überlagern sich zwei Geometrien mit unterschiedlichen Materialeigenschaften entsteht, aus Sicht der Arbeit, ein undefinierter Bereich. McGrattan et al. weisen darauf hin, dass in diesem Fall das zweite

[6] Vergleich zwischen den &OBST-Anweisungen in der FDS-Eingabedatei und den Maßen im Bauwerksmodell.
[7] In der Abbildung 6.5 orange dargestellt.

Bauteil die Eigenschaften des ersten Bauteils überschreibt[8]. Dies kann mit der Anweisung `OVERLAY=.FALSE.` verhindert werden (2017, S. 58). Damit wird seitens des *Fire Dynamics Simulator* eine Lösungsmöglichkeit geboten.

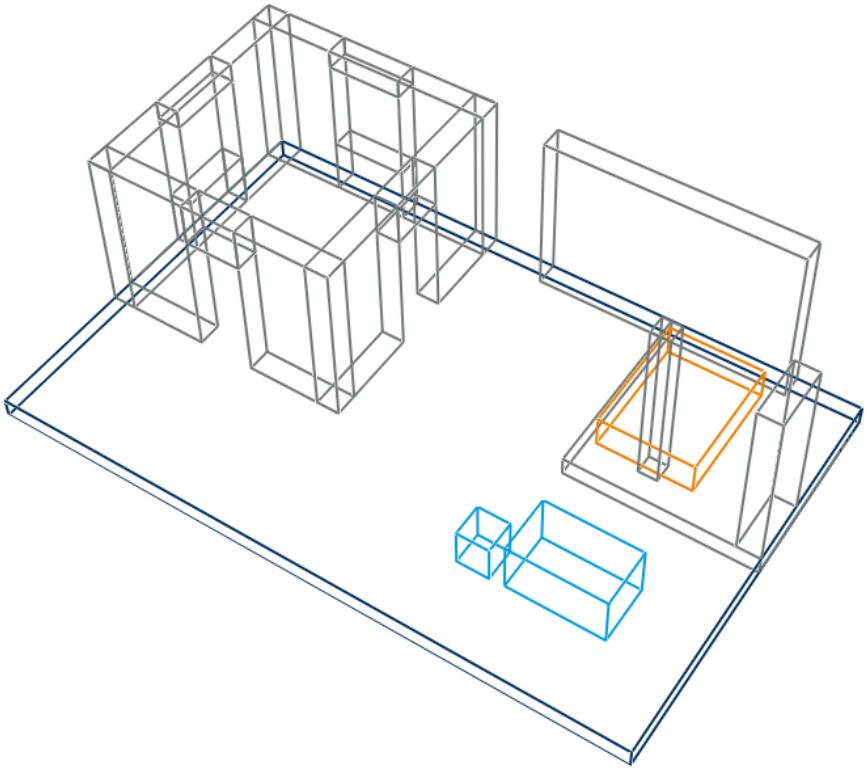

Abb. 6.6.: Geometrische Überschneidungen im FDS-Simulationsmodell.

Das Programm zeigt, dass der Zugriff auf Informationen in einem Datenmodell lediglich über einen Identifikator (ID) möglich ist. Die Datenstrukturen der Fachmodelle aus Kapitel 5 sehen eine eindeutige ID vor, womit auf die Implementierungsmöglichkeit der übrigen Fachmodelle geschlossen werden kann. Insgesamt werden alle Gebäudeelemente nach den entwickelten Rechenvorschriften transformiert (vgl. Abschnit 4.3)[9]. Das vorgestellte Multimodell ist verifiziert, da Informationen aus unabhängigen Fachmodellen zu einer einheitlichen Informationsressource kombiniert werden können (vgl. Abbildung 6.3). Die erfolgreiche Anwendung dieser Ressource zeigen die weiteren Abbildungen.

Schlussendlich wird mit dem vorgestellten Programm das Multimodell nach Abbildung 5.12 sowie die Transformationsvorschriften nach Abschnitt 4.3 verifiziert. Der Hauptteil der Arbeit endet hier. Abschließend werden die Ergebnisse zusammengefasst und ein Ausblick auf weitere Entwicklungsmöglichkeiten gegeben.

[8] Das erste Bauteil steht in der FDS-Eingabedatei vor dem zweiten Bauteil.
[9] Mit Ausnahme der Bezugshöhe $Z_{elevation}$; eine zusätzliche Filterroutine ist zu entwickeln.

7. Schlussbetrachtung

Man sieht die Blumen welken, und die Blätter fallen, aber man sieht auch Früchte reifen und neue Knospen keimen.

(Johann Wolfgang von Goethe)

7.1. Zusammenfassung

Die Diplomarbeit beschreibt ein Konzept für die effiziente Durchführung von Brandsimulationen. Dabei definiert sie Effizienz als eine möglichst hohe Aussagefähigkeit zu einem Brandereignis auf Grundlage einer zur Verfügung stehenden Informationsressource. Die Entstehung dieser Ressource wird anhand des Multimodellansatzes sowie dem Variationsprinzip für Anwendungen mit dem *Fire Dynamics Simulator* (FDS) diskutiert. Hierbei handelt es sich um eine gemeinfreie Software zur Berechnung von thermisch angetriebenen Strömungen, die das Visualisierungsprogramm *Smokeview* (SMV) veranschaulicht. Für die Konzeptumsetzung entwickelt die Arbeit ein eigenes Programm, mit dem gezielt Informationen aus Datenmodellen gebündelt und in eine FDS-Eingabedatei überführt werden.

Ein FDS-Minimalbeispiel dient dabei der Einführung in das Simulationsprogramm. Hieraus resultiert eine kompilierbare Eingabedatei, deren Berechnungsergebnisse einen Zimmerbrand zeigen. Mit dem Beispiel wird ein Bauwerk dargestellt, eine Zündquelle modelliert und eine Brandausbreitung durch brennbare Gegenstände verdeutlicht. Demzufolge werden Informationen für die Generierung der programmspezifischen Eingabebefehle, die als Mindestmaß für die Erstellung eines Multimodells dienen, offengelegt. Ein wesentlicher Bestandteil stellt dabei der Informationsraum der *Industry Foundation Classes* (IFC) dar, mit dem Daten für die Bauwerksmodellierung gewonnen werden. Aus der Untersuchung der IFC-Klassen ergeben sich Möglichkeiten, Informationen zielgerichtet für die Generierung der einzelnen FDS-Anweisungen zu nutzen. Insgesamt resultieren hieraus Rechenvorschriften für die Transformation der Geometrien in die FDS-Eingabesyntax sowie Erkenntnisse über fehlende Informationsdaten, die eine vollständige Generierung der Eingabedatei ermöglichen. In dem Zusammenhang werden drei weitere Datenmodelle entwickelt, die einen für sich abgeschlossenen Fachbereich repräsentieren. Ein Multimodell bietet die Möglichkeit, diese Informationsräume durch eine gezielte Verlinkung von Datenobjekten in Beziehung zu setzen. Die vorliegende Arbeit zeigt, wie durch die Verwendung eines Linkmodells einzelne Informationen zu einer einheitlichen Ressource gebündelt werden. Demzufolge entsteht eine Datenbasis, die sowohl eine zielgerichtete Generierung von FDS-Anweisungen ermöglicht als auch essenzielle Variationsmöglichkeiten in den Parametern aufzeigt. Damit entsteht ein weiteres Datenmodell, das die Variation eines Brandereignisses zulässt und realitätsnahe Schwankungen einzelner Parameter berücksichtigt.

Insgesamt wird ein Konzept für die Durchführung von FDS-Simulationen, das auf Informationsdaten aus einem Multi- und Variationsmodell basiert, beschrieben. Die Konzeptumsetzung erlaubt, die theoretischen Ansätze zu verifizieren sowie Antworten auf die, für diese Arbeit relevanten, Fragen zu geben.

7.2. Ergebnisse der Arbeit

Mit dieser Arbeit soll untersucht werden, inwieweit die *Industry Foundation Classes* (IFC) als Standard für Brandsimulationen im Bauwesen geeignet sind. In dem Zusammenhang werden mit dem vorliegenden Werk Rechenvorschriften dargelegt, die Transformationen von IFC-Daten in die programmspezifische Eingabesyntax des *Fire Dynamics Simulator* ermöglichen. Die Transformationen beschränken sich auf Instanzen der *IfcElement*, wodurch Maße und Platzierungen von beispielsweise Wänden und Türöffnungen in das Simulationsprogramm übertragen werden. Dabei stützt sich der Ansatz auf Angaben einer *BoundingBox*. Die Ergebnisse aus der Programmentwicklung zeigen die erfolgreiche Umsetzung der theoretischen Überlegungen. Es ist festzustellen, dass in den Eckbereichen der Geometrien Überschneidungen entstehen. Aus Sicht der Arbeit werden die Überlappungen nur bei der Vergabe unterschiedlicher Materialparameter relevant. Weiterhin sind im *Fire Dynamics Simulator* interne Lösungsmöglichkeiten geboten, welche die Überschreibung von Materialparametern unterbinden. Die Arbeit zeigt, dass die *Industry Foundation Classes* eine automatisierte Übertragung der Gebäudeelemente in den *Fire Dynamics Simulator* ermöglichen. Sie beschränkt sich jedoch ausschließlich auf orthogonal verlaufende Geometrien.

Insgesamt beschäftigt sich die vorliegende Arbeit mit der Fragestellung, inwieweit der Multimodellansatz und das Variationsprinzip für Anwendungen im Bereich der Brandsimulationen geeignet sind. Hierzu werden neben einem Bauwerksmodell drei weitere Fachmodelle und ein Linkmodell vorgestellt. Die Instanzen der Fachmodelle weisen durch die Datenstrukturen eindeutige Identifikatoren auf, was als Kriterium für den Multimodellansatz vorauszusetzen ist. Durch das Programm wird das Linkmodell erfolgreich ausgewertet. Die Datenschemen der Fachmodelle sind bekannt, sodass mit Filterroutinen einzelne Informationen aus den Modellen entnommen werden können. Dieser Vorgang wird exemplarisch am Bauwerksmodell veranschaulicht, womit die Anwendung des Multimodells bestätigt ist. Aus Sicht der Arbeit ist daher der Multimodellansatz für Brandsimulationen im Bauwesen geeignet und bietet den Vorteil, voneinander unabhängige Datenmodelle zu kombinieren. Das Variationsprinzip kann in der Arbeit konzeptionell berücksichtigt werden. Hierfür werden Attribute bzw. Parameter aufgezeigt, deren Eigenschaften ein Brandereignis maßgebend beeinflussen. Diese werden exemplarisch für die Erstellung eines Variationsmodells verwendet. Demzufolge kann die Frage, inwieweit das Variationsprinzip für FDS-Simulationen Anwendung finden kann, nur teilweise beantwortet werden. Mit der Möglichkeit der Generierung eines Variationsmodells entsteht jedoch ein Hilfsmodell, was innerhalb einer Programmentwicklung berücksichtigt werden kann. Daher ist aus Sicht der Arbeit die Implementierung des Modells möglich.

Mit Abschluss der Diplomarbeit liegt ein Multimodell für Brandsimulationen mit dem *Fire Dynamics Simulator* vor. Die theoretischen Überlegungen sind durch eine eigene Programmentwicklung verifiziert, womit auch die Grundlage für weiterführende Software-

entwicklungen geschaffen wird. In dem Zusammenhang wird festgestellt, wie komplex und zeitintensiv Programmerstellungen werden können, was sich bereits bei der Umsetzung der vorgestellten Transformationsvorschriften zeigt. Vordergründig ist dies sicherlich mit der noch fehlenden Erfahrung im Umgang derartiger Projekte begründbar. Doch gerade diese Aufgaben bergen einen signifikanten Mehrwert, da eigene Überlegungen umgesetzt, Potenziale erkannt und verwirklicht werden können. Im Nachhinein zeigt sich, wie Datenstrukturen problemlos entwickelt und in einem Multimodell umsetzbar sind. Die Frage, ob ein Multimodell für Brandsimulationen anwendbar ist, kann unter der Voraussetzung eines existierenden Datenschemas und einer eindeutigen ID, mit Beendigung dieser Arbeit umformuliert werden: Gibt es eine Anwendung, die nicht mit einem Multimodell realisiert werden kann? Derartige Erkenntnisse entstehen sicherlich erst durch eigenständige Entwicklungen von Fachmodellen, deren Verknüpfungen untereinander sowie einer konkreten Anwendung und erlauben einen abschließenden Ausblick.

7.3. Ausblick

Der Detaillierungsgrad der entwickelten Applikation bestimmt den Funktionsumfang des vorgestellten Multimodells. Durch die vollständige Implementierung der Fachmodelle sowie die Berücksichtigung nicht orthogonaler Geometrieverläufe kann der Mehrwert für die Anwendung mit dem *Fire Dynamics Simulator* erhöht werden. Hierzu sind weitere Rechenvorschriften zu entwickeln und innerhalb einzelner Programmroutinen zu realisieren. Es können beliebig viele Fach- und Linkmodelle erweitert werden, womit die Frage nach weiteren Anwendungsbereichen entsteht. In diesem Zusammenhang ist auf die Möglichkeit der Simulation von Personenströmen hinzuweisen. Damit wird ein weiterer Fachbereich aus den Ingenieurmethoden des Brandschutzes tangiert. Abschließend ist das Variationsmodell in dem Programm zu implementieren, womit die Durchführung von effizienten Brandsimulationen mit dem *Fire Dynamics Simulator* realisiert wird.

Ende.

Literatur- und Quellenverzeichnis

Literatur

[Bei10] G. Beilicke. *Bautechnischer Brandschutz: Brandlastberechnung*. BBV, Beilicke-Brandschutz-Verlag, 2010.

[GG12] K. Grewolls und G. Grewolls. *Praxiswissen Brandschutz - Simulationen: schneller Einstieg und kompaktes Wissen ; mit 8 Tabellen*. FeuerTRUTZ Verlag, 2012.

[Hos13] Dietmar Hosser. *Leitfaden Ingenieurmethoden des Brandschutzes. vfdb TB 04-01 November 2013*. 3., überarbeitete und ergänzte Auflage November 2013. Technischer Bericht. 2013.

[MB18] Josef Mayr und Lutz Battran. *Brandschutzatlas*. Band 1, Stand März 2018. FeuerTRUTZ, Verlag für Brandschutzpublikationen, 2018.

[McG+17] Kevin McGrattan u. a. *Fire Dynamics Simulator User's Guide*. NIST Special Publication 1019 Sixth Edition. Manuel. National Institute of Standards und Technology. Nov. 2017.

[Meh17] Dr.-Ing. Friedrich Mehl. *Ingenieurmethoden im Brandschutz, Einführung in die Grundlagen*. Promat GmbH, 2017.

[OL03] Herbert Oertel und Eckart Laurien. *Numerische Strömungsmechanik :Grundgleichungen, Lösungsmethoden, Softwarebeispiele /*. 2., neu bearb. Aufl. Vieweg, 2003.

[Sch+18] Raimar J. Scherer u. a. *A variation model method for real time system identification in bridge health monitoring*. Paper. Institute of Construction Informatics, TU Dresden, Germany, 2018.

[See95] Thomas Seeger. *Aspekte der Professionalisierung des Berufsfeldes Information :Beiträge zu Ausbildung und Beruf in der Informationslandschaft anläßlich des 10jährigen Bestehens des Fachbereichs Information und Dokumentation der Fachhochschule Darmstadt*. Konstanz : Universitätsverl., 1995.

[SS14] R.J. Scherer und S.E. Schapke. *Informationssysteme im Bauwesen 1: Modelle, Methoden und Prozesse*. VDI-Buch. Springer Berlin Heidelberg, 2014.

Normen, Regelwerke und Richtlinien

[ARG02] Bauministerkonferenz (ARGEBAU). *Musterbauordnung -MBO-. Fassung November 2002*. Zuletzt geändert durch Beschluss der Bauministerkonferenz vom 13.05.2016. Mustervorschrift/Mustererlass. 2002.

[Inn17] Sächsischen Staatsministeriums des Innern. *Verwaltungsvorschrift des Sächsischen Staatsministeriums des Innern zur Sächsischen Bauordnung. Sächs-ABl.SDr. 2005 Nr. 2, S. 59 Fsn-Nr.: 421-V05.1*. Fassung gültig ab: 1. September 2017. Verwaltungsvorschrift. 2017.

[Mül15] Hans Müller-Steinhagen. *Prüfungsordnung für den Diplom-Studiengang Bauingenieurwesen an der Technischen Universität Dresden*. Lesefassung 10.03.2018. Ordnung. 2015.

[Nor10] DIN Deutsches Institut für Normung. *Eurocode 1: Einwirkungen auf Tragwerke –Teil 1-2: Allgemeine Einwirkungen – Brandeinwirkungen auf Tragwerke; Deutsche Fassung EN 1991-1-2:2002 + AC:2009*. Norm. Dez. 2010.

[Sch00] VdS Schadenverhütung. *Bemessungsbrände für Brandsimulationen und Brandschutzkonzepte, VdS 2827:2000-05(01)*. Publikation. 2000.

Internet

[B118] buildingSMART. *Industry Foundation Classes Release 4 (IFC4)*. Internetquelle. Abgerufen am: 30.07.2018. URL: http://www.buildingsmart-tech.org/ifc/IFC4/final/html/.

[B218] buildingSMART. *IfcBuildingElement, Inherited definitions from supertypes*. Internetquelle. Abgerufen am: 31.07.2018. URL: http://www.buildingsmart-tech.org/ifc/IFC4/Add2/html/schema/ifcproductextension/lexical/ifcbuildingelement.htm.

[B318] buildingSMART. *System | Multimodelle*. Internetquelle. Abgerufen am: 12.08.2018. URL: https://www.buildingsmart.de/kos/WNetz?art=Project.show&id=143.

[W18] WIKIPEDIA Die freie Enzyklopädie. *Numerische Strömungsmechanik*. Internetquelle. Abgerufen am: 13.07.2018. URL: https://de.wikipedia.org/wiki/Numerische_Str%C3%B6mungsmechanik.

Software

[buia] buildingSMART. *Industry Foundation Classes Release 4 (IFC4))*. buildingSMART e. V., Germany. EXPRESS-Schema. URL: http://www.buildingsmart-tech.org/specifications/ifc-releases/ifc4-release.

[buib] buildingSMART. *LinkModel*. buildingSMART e.V. XML Schema Definition (XSD). URL: https://www.buildingsmart.de/kos/WNetz?art=Project.show&id=143.

[DOM] DOM. *Document Object Model (DOM)*. Application Programming Interface. URL: https://www.w3.org/DOM/.

[Fou] Eclipse Foundation. *Eclipse Oxygen.3a (4.7.3a)*. Eclipse Foundation. development platform. URL: http://www.eclipse.org/downloads/packages/.

[Gra] Graphisoft. *Archicad 21, Studentenversion*. Graphisoft Deutschland GmbH, Germany. CAD/BIM Software. URL: http://www.gshelp.de/edu-next/200_download.html.

[JAX] JAXB. *JAXB*. 4.40. Application Programming Interface, Eclipse plugin. URL: https://javaee.github.io/jaxb-v2/.

[LKS] LKSoft. *JSDAI*. LKSoftWare GmbH, Steinweg 1 (Pilgerzell), 36093 Kuenzell, Germany. Application Programming Interface, Eclipse plugin.

[SC] NIST National Institute of Standards und Technology U.S.Department of Commerce. *Fire Dynamics Simulator (FDS) and Smokeview (SMV)*. NIST National Institute of Standards and Technology U.S.Department of Commerce, USA. simulation program. URL: https://pages.nist.gov/fds-smv/downloads.html.

[Sch+] Raimar J. Scherer u. a. *variation model*. Institute of Construction Informatics, TU Dresden, Germany. XML Schema Definition (XSD).

Anlage 4

Selbstständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsinstitution vorgelegt und ist auch noch nicht veröffentlicht worden.

Dresden, den

.....
(Unterschrift)

A. Anhang

A.1. Anhang zum FDS-Minimalbeispiel

Quelltext A.1: Vollständiger Quelltext der Eingabedatei

```
/----------------------------------------------------------------------------------------
/----------------------------- Simulationsprofil ----------------------------------------
/----------------------------------------------------------------------------------------
&HEAD FYI                  = 'Neuen Auftrag anlegen'                                    ,
      CHID                 = 'FDS-Minimalbeispiel_V1'                                   ,
      TITLE                = 'Zimmerbrand Variante 1'                                   /
&TIME FYI                  = 'Simulationszeit in Sekunden'                              ,
      T_End                =  60.0000                                                   /
&MESH FYI                  = 'Simulationsraum, 60 Maschen in x,y,z'                     ,
      ID                   = 'Netz'                                                     ,
      IJK                  =  60,60,60                                                  ,
      XB                   =  0.000,8.830,0.000,4.465,0.000,4.000                       ,
      RGB                  =  230,230,250                                               /
&VENT FYI                  = 'Simulationsraum-Xmax oeffnen'                             ,
      MB                   = 'XMAX'                                                     ,
      SURF_ID              = 'OPEN'                                                     /
&VENT FYI                  = 'Simulationsraum-Ymax Oeffnung'                            ,
      XB                   =  0.000,0.350,4.465,4.465,1.400,2.900                       ,
      SURF_ID              = 'OPEN'                                                     /
/----------------------------------------------------------------------------------------
/----------------------------- Bauwerksmodell -------------------------------------------
/----------------------------------------------------------------------------------------
&OBST FYI                  = 'Modellierung Deckenbauteile'                              ,
      ID                   = 'Kellerdecke'                                              ,
      XB                   =  0.865,7.865,0.500,3.965,0.300,0.500                       ,
      RGB                  =  105,105,105                                               /
&OBST ID                   = 'Geschossdecke'                                            ,
      XB                   =  0.865,7.865,0.500,3.965,3.500,3.600                       ,
      RGB                  =  105,105,105                                               /
/----------------------------------------------------------------------------------------
&OBST FYI                  = 'Modellierung Wandbauteile'                                ,
      ID                   = 'Aussenwand'                                               ,
      XB                   =  7.865,8.230,0.000,4.465,0.000,4.000                       ,
      RGB                  =  128,128,128                                               /
&OBST ID                   = 'Trennwand'                                                ,
      XB                   =  0.865,7.865,3.600,3.965,0.500,3.500                       ,
      RGB                  =  128,128,128                                               /
&OBST ID                   = 'Treppenraumwand'                                          ,
      XB                   =  0.500,0.865,0.000,4.465,0.000,4.000                       ,
      RGB                  =  128,128,128                                               /
&OBST ID                   = 'Innenwand'                                                ,
      XB                   =  0.865,7.865,0.500,0.600,0.500,3.500                       ,
      TRANSPARENCY         =  0.250                                                     ,
      RGB                  =  128,128,128                                               /
/----------------------------------------------------------------------------------------
&HOLE FYI                  = 'Modellierung Oeffnungen'                                  ,
      ID                   = 'Eingangstuer'                                             ,
      XB                   =  0.000,0.865,1.595,2.605,0.500,2.501                       /
&HOLE ID                   = 'Zimmertuer'                                               ,
      XB                   =  1.000,1.885,0.500,0.600,0.500,2.501                       /
&HOLE ID                   = 'Fenster'                                                  ,
      XB                   =  7.865,8.230,1.400,2.600,1.400,2.900                       /
/----------------------------------------------------------------------------------------
/----------------------------- Brandentstehung ------------------------------------------
```

```
/----------------------------------------------------------------------------------------------
&OBST FYI                  = 'Modellierung Entstehungsbrand'                                  ,
      ID                   = 'E-Geraet'                                                       ,
      XB                   =  7.200,7.700,0.700,1.200,0.500,1.000                             ,
      RGB                  =  0,0,0                                                           ,
      OUTLINE              = .TRUE.                                                           /
&VENT FYI                  = 'Bereich auf Oberflaeche'                                        ,
      XB                   =  7.300,7.600,0.800,1.100,1.000,1.000                             ,
      SURF_ID              = 'Defekt'                                                         /
&SURF FYI                  = 'Waermefreisetzungsrate'                                         ,
      ID                   = 'Defekt'                                                         ,
      RGB                  =  255,0,0                                                         ,
      HRRPUA               =  1000.000                                                        ,
      RAMP_Q               = 'Funktionsverlauf'                                               /
&RAMP ID                   = 'Funktionsverlauf', T= 5.000, F=0.000                            /
&RAMP ID                   = 'Funktionsverlauf', T=10.000, F=1.000                            /
&RAMP ID                   = 'Funktionsverlauf', T=30.000, F=1.000                            /
&RAMP ID                   = 'Funktionsverlauf', T=31.000, F=0.000                            /
&REAC FYI                  = 'Verbrennungsreaktion'                                           ,
      FUEL                 = 'PROPANE'                                                        ,
      SOOT_YIELD           =  0.010                                                           /
/----------------------------------------------------------------------------------------------
/----------------------------- Brennbare Gegenstaende -----------------------------------------
/----------------------------------------------------------------------------------------------
&OBST FYI                  = 'Modellierung Regalwand'                                         ,
      ID                   = 'Regal1'                                                         ,
      XB                   =  2.500,7.700,0.600,0.800,1.500,1.650                             ,
      SURF_ID              = 'Regaleigenschaften'                                             /
&OBST ID                   = 'Regal2'                                                         ,
      XB                   =  2.500,7.700,0.600,0.800,2.100,2.250                             ,
      SURF_ID              = 'Regaleigenschaften'                                             /
&SURF FYI                  = 'Eigenschaften'                                                  ,
      ID                   = 'Regaleigenschaften'                                             ,
      RGB                  =  41,36,33                                                        ,
      THICKNESS            =  0.050                                                           ,
      MATL_ID              = 'Holz'                                                           ,
      BURN_AWAY            = .TRUE.                                                           /
&MATL FYI                  = 'Material'                                                       ,
      ID                   = 'Holz'                                                           ,
      SPEC_ID              = 'PROPANE'                                                        ,
      DENSITY              =    40.000                                                        ,
      SPECIFIC_HEAT        =     6.500                                                        ,
      CONDUCTIVITY         =     0.050                                                        ,
      HEAT_OF_REACTION     =    10.000                                                        ,
      HEAT_OF_COMBUSTION   = 30000.000                                                        ,
      REFERENCE_TEMPERATURE =   120.000                                                       ,
      N_REACTIONS          =     1.000                                                        ,
      NU_SPEC              =     1.000                                                        /
/----------------------------------------------------------------------------------------------
&OBST FYI                  = 'Modellierung Sofa'                                              ,
      ID                   = 'Sitzteil'                                                       ,
      XB                   =  2.500,7.000,0.700,1.700,0.500,0.700                             ,
      SURF_ID              = 'Sofaeigenschaften'                                              /
&OBST ID                   = 'Kopfteil'                                                       ,
      XB                   =  2.500,7.000,0.700,1.000,0.700,1.000                             ,
      SURF_ID              = 'Sofaeigenschaften'                                              /
&SURF FYI                  = 'Eigenschaften'                                                  ,
      ID                   = 'Sofaeigenschaften'                                              ,
      RGB                  =  41,36,33                                                        ,
```

```
      THICKNESS             =  0.200                                                  ,
      MATL_ID               = 'Textil'                                                ,
      BURN_AWAY             = .TRUE.                                                  /
&MATL FYI                   = 'Material'                                              ,
      ID                    = 'Textil'                                                ,
      SPEC_ID               = 'PROPANE'                                               ,
      DENSITY               =    40.000                                               ,
      SPECIFIC_HEAT         =     1.000                                               ,
      CONDUCTIVITY          =     0.050                                               ,
      HEAT_OF_REACTION      =  1000.000                                               ,
      HEAT_OF_COMBUSTION    = 30000.000                                               ,
      REFERENCE_TEMPERATURE =   150.000                                               ,
      N_REACTIONS           =     1.000                                               ,
      NU_SPEC               =     1.000                                               /
/-----------------------------------------------------------------------------------------
&OBST FYI                   = 'Modellierung Schrank'                                  ,
      ID                    = 'Korpus'                                                ,
      XB                    =  2.000,7.200,3.100,3.600,0.500,3.100                    ,
      SURF_ID               = 'Schrankeigenschaften'                                  /
&SURF FYI                   = 'Eigenschaften'                                         ,
      ID                    = 'Schrankeigenschaften'                                  ,
      RGB                   =  41,36,33                                               ,
      THICKNESS             =  0.200                                                  ,
      MATL_ID               = 'Eiche'                                                 ,
      BURN_AWAY             = .TRUE.                                                  /
&MATL FYI                   = 'Material'                                              ,
      ID                    = 'Eiche'                                                 ,
      SPEC_ID               = 'PROPANE'                                               ,
      DENSITY               =    40.000                                               ,
      SPECIFIC_HEAT         =     1.000                                               ,
      CONDUCTIVITY          =     0.050                                               ,
      HEAT_OF_REACTION      = 10000.000                                               ,
      HEAT_OF_COMBUSTION    = 20000.000                                               ,
      REFERENCE_TEMPERATURE =   350.000                                               ,
      N_REACTIONS           =     1.000                                               ,
      NU_SPEC               =     1.000                                               /
/-----------------------------------------------------------------------------------------
/----------------------------- Auswertung ------------------------------------------------
/-----------------------------------------------------------------------------------------
&SLCF FYI                   = 'Temperaturscheibe x-Ebene'                             ,
      QUANTITY              = 'TEMPERATURE'                                           ,
      PBX                   =  7.500                                                  /
/-----------------------------------------------------------------------------------------
&SLCF FYI                   = 'Temperaturscheibe y-Ebene'                             ,
      QUANTITY              = 'TEMPERATURE'                                           ,
      PBY                   =  2.200                                                  /
/-----------------------------------------------------------------------------------------
&SLCF FYI                   = 'Temperaturscheibe z-Ebene'                             ,
      QUANTITY              = 'TEMPERATURE'                                           ,
      PBZ                   =  2.000                                                  /
/-----------------------------------------------------------------------------------------
&TAIL                                                                                 /
```

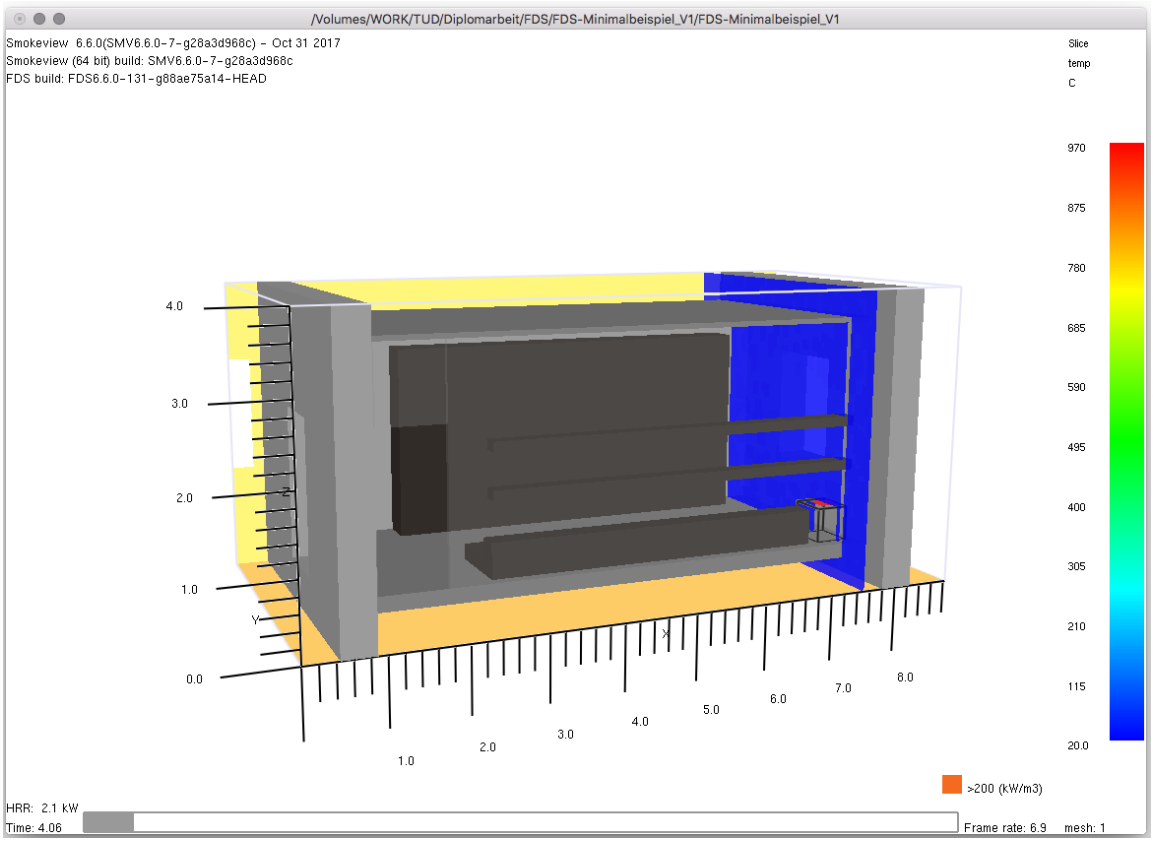

Abb. A.1.: Auswertung und Simulation nach T = 4 Sekunden.

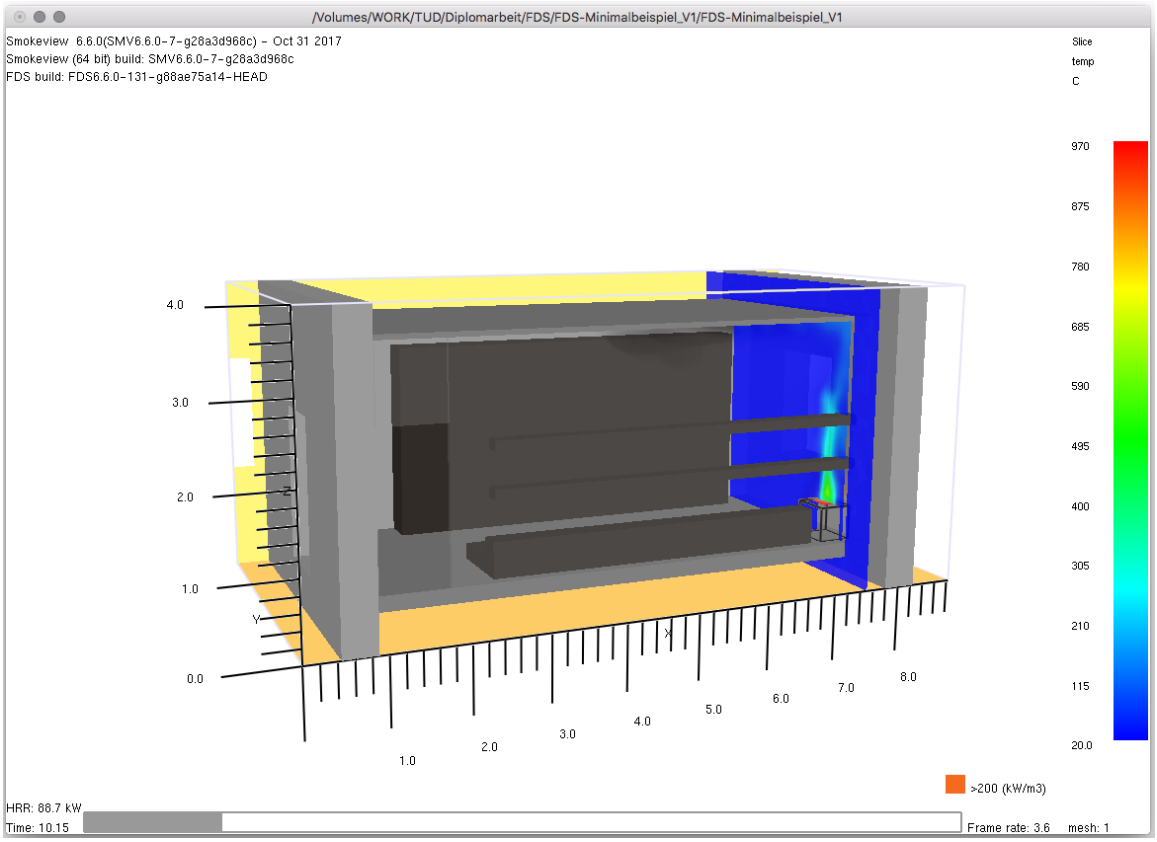

Abb. A.2.: Auswertung und Simulation nach T = 10 Sekunden.

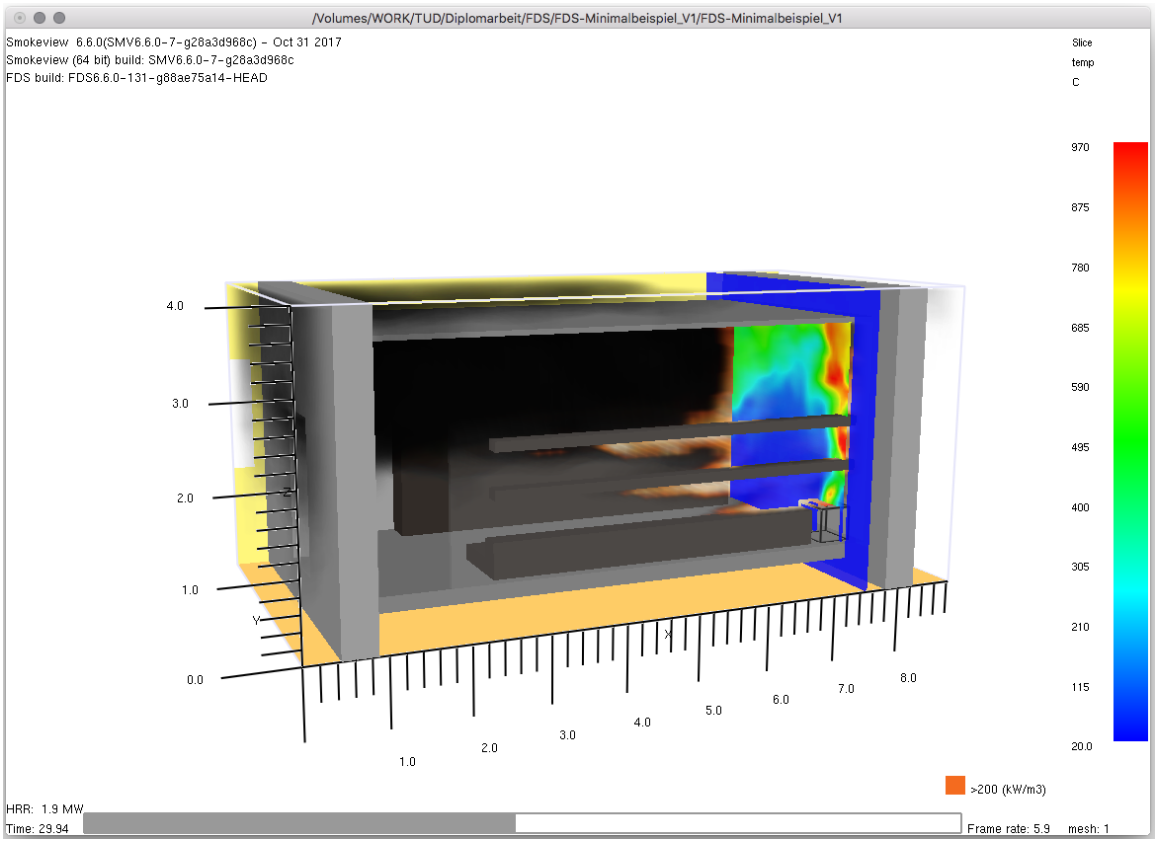

Abb. A.3.: Auswertung und Simulation nach T = 30 Sekunden.

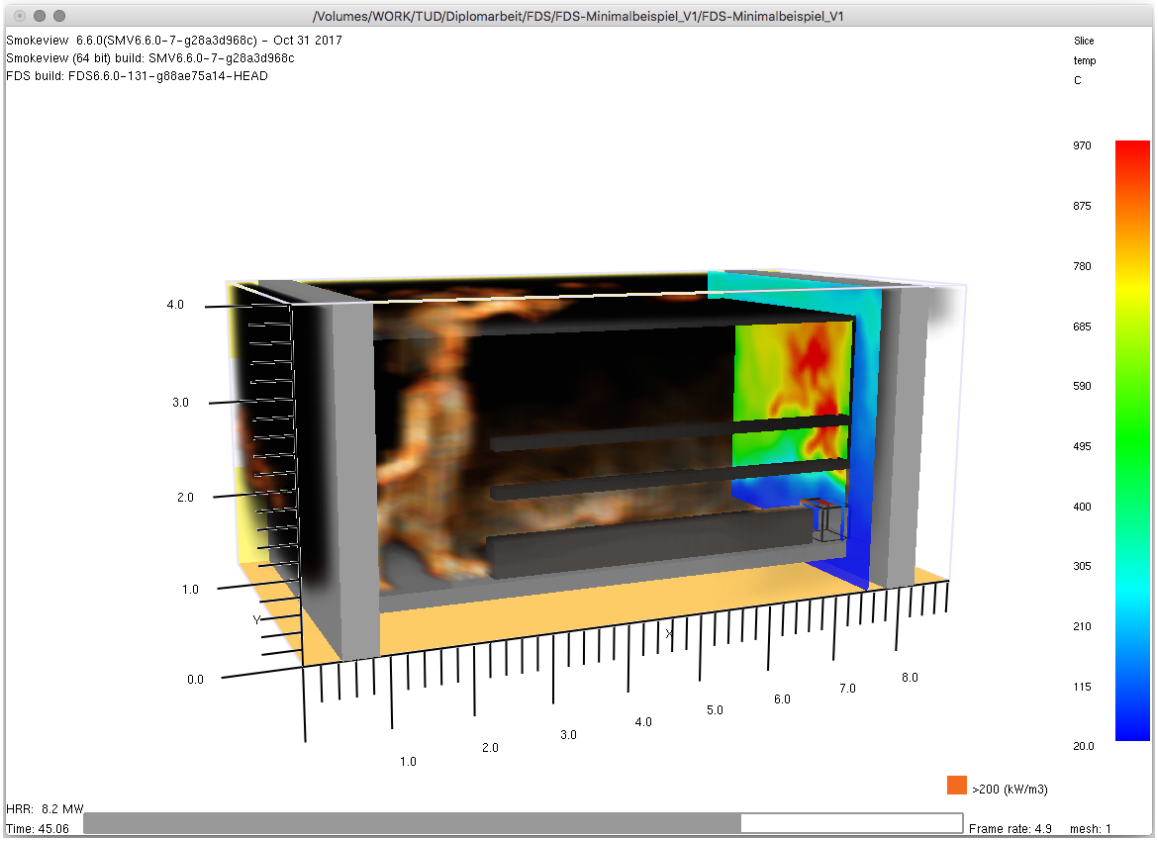

Abb. A.4.: Auswertung und Simulation nach T = 45 Sekunden.

Abb. A.5.: Brandentstehung nach T = 7 Sekunden.

Abb. A.6.: Brandentstehung nach T = 10 Sekunden.

Abb. A.7.: Brandentstehung nach T = 15 Sekunden.

Abb. A.8.: Brandentstehung nach T = 20 Sekunden.

A.2. Anhang zu den Industry Foundation Classes

Quelltext A.2: Darstellung der Gebäudeelemente durch die IFC

```
#471    = IFCSLAB('0iFuOXwidzIhaetDff_ZCU',#12,'Decke-001',$,$,#143,#465,
          '2C3F8621-EAC9-FD4AB928-DCDA69FA331E',.FLOOR.);

#657    = IFCWALLSTANDARDCASE('0qZwOvTu2lI8JQNIqy4piW',#12,'Au\X2\00DF\X0\enwand-001',
          $,$,#609,#652,'348FA639-7780-AF48-84DA-5D2D3C133B20',$);

#2080   = IFCWINDOW('3sKT00B3$hIfZZQNagws7k',#12,'Fenster-001',$,$,#762,#2076,'F651D000-
          2C3F-EB4A-98E3-69792AEB61EE',1.5,1.2,$,$,$);

#2360   = IFCWINDOW('0$JCI5IpacIe2Dmc_6N3ge',#12,'Fenster-002',$,$,#2342,#2356,'3F4CC485-
          4B39-264A-808D-C26F865C3AA8',1.5,1.2,$,$,$);

#2247   = IFCWALL('1fYCD$XnKhJeI9hiJq13Sm',#12,'Au\X2\00DF\X0\enwand-002',$,$,#2153,#2242,
          '6988C37F-8715-2B4E-8489-AEC4F4043730',$);

#2519   = IFCWALL('0Gidd$L_VHHe6UBzVVzylL',#12,'Innenwand-002',$,$,#2425,#2514,
          '10B279FF-57E7-D146-819E-2FD7DFF7CBD5',$);

#7622   = IFCWALL('3OBd3RJ3JFJR_MuJi 03uO',#12,'Innenwand-001',$,$,#7478,#7617,'D82E70DB-
          4C34-CF4D-BF96-E13B3E003E18',$);

#7399   = IFCDOOR('3HVEvcYIQ7JfesX2xIeMzD',#12,'T\X2\00FC\X0\r-002',$,$,#2611, #7395,
          'D17CEE66-8926-874E-9A36-842ED2A16F4D',2.01,0.885,$,$,$);

#12499  = IFCDOOR('3bnrXGmssuJ9pw82oHx2zu',#12,'T\X2\00FC\X0\r-001,$$,#7717,#12495,
          'E5C75850-C36D-B84C-9CFA-202C91EC2F78',2.01,0.885,$,$,$);

#13004  = IFCWALLSTANDARDCASE('1eTCwoamvkJPqjhvH1AY3V',#12,'Au\X2\00DF\X0\enwand-004',
          $,$,#12960,#12999,'6874CEB2-930E-6E4D-9D2D-AF94412A20DF',$);

#13099  = IFCWALLSTANDARDCASE('2FoZcgvEKQIvfB7xghWBFV',#12,'Au\X2\00DF\X0\enwand-003',
          $,$,#13053,#13094,'8FCA39AA-E4E5-1A4B-9A4B-1FBAAB80B3DF',$);

#23455  = IFCFURNISHINGELEMENT('2hWTgkIaY0JB8j4RUlX LU',#12,'HiFi-Anlage',$,$,#13148,#23451,
          'AB81DAAE-4A48-804C-B22D-11B7AF87E55E');

#39901  = IFCFURNISHINGELEMENT('0jSKBmipwvIuqyplvk2hXT',#12,'Designer Couch',$,$,#23510,
          #39897,'2D7142F0-B33E-B94B-8D3C-CEFE6E0AB85D');

#40266  = IFCSLAB('1bOkeNMipiHQU2BZbUPAmo',#12,'Decke-002',$,$,#40228,#40262,'6562EA17-
          5ACC-EC45-A782-2E395E64AC32',.FLOOR.);

#40038  = IFCSPACE('1tV0ztwINiHgG7hrZD7239',#12,'R-002',$,$,#39948,#40033,'B\X2\00FC\X0\ro',
          .ELEMENT.,$,$);

#40157  = IFCBUILDINGELEMENTPROXY('29_bY9$ZbNGefVVWnuRZrK',#12,'Brandlast',$,$,#40083,
          #40153,'89FA5889-FE39-5742-8A5F-7E0C786E3D54',.NOTDEFINED.);
```

Formeln A.1.: Transformationsvorschriften für die IfcDirection (1, 0, 0).

$$
\begin{aligned}
X_{1,fds} &= x_0 + x_1 \\
X_{2,fds} &= x_0 + x_1 + x_{dim} \\
Y_{1,fds} &= y_0 + y_1 \\
Y_{2,fds} &= y_0 + y_1 + y_{dim} \\
Z_{1,fds} &= z_0 + z_1 + z_{elevation} \\
Z_{2,fds} &= z_0 + z_1 + z_{elevation} + z_{dim}
\end{aligned}
$$

$$
\begin{aligned}
X_{1,fds} &= x_{0,wall} + x_{1,wall} + x_{1,opening} + x_2 \\
X_{2,fds} &= x_{0,wall} + x_{1,wall} + x_{1,opening} + x_2 + x_{dim,opening} \\
Y_{1,fds} &= y_{0,wall} + y_{1,wall} \\
Y_{2,fds} &= y_{0,wall} + y_{1,wall} + y_{dim,wall} \\
Z_{1,fds} &= z_{0,wall} + z_{1,wall} + z_{1,opening} + z_2 + z_{elevation} \\
Z_{2,fds} &= z_{0,wall} + z_{1,wall} + z_{1,opening} + z_2 + z_{elevation} + z_{dim,opening}
\end{aligned}
$$

Formeln A.2.: Transformationsvorschriften für die IfcDirection (0, 1, 0).

$$
\begin{aligned}
X_{1,fds} &= x_0 - y_1 \\
X_{2,fds} &= x_0 - y_1 - y_{dim} \\
Y_{1,fds} &= y_0 + x_1 \\
Y_{2,fds} &= y_0 + x_1 + x_{dim} \\
Z_{1,fds} &= z_0 + z_1 + z_{elevation} \\
Z_{2,fds} &= z_0 + z_1 + z_{elevation} + z_{dim}
\end{aligned}
$$

$$
\begin{aligned}
X_{1,fds} &= x_{0,wall} - y_{1,wall} \\
X_{2,fds} &= x_{0,wall} - y_{1,wall} - y_{dim,wall} \\
Y_{1,fds} &= y_{0,wall} + x_{1,wall} + x_{1,opening} + x_2 \\
Y_{2,fds} &= y_{0,wall} + x_{1,wall} + x_{1,opening} + x_2 + x_{dim,opening} \\
Z_{1,fds} &= z_{0,wall} + z_{1,wall} + z_{1,opening} + z_2 + z_{elevation} \\
Z_{2,fds} &= z_{0,wall} + z_{1,wall} + z_{1,opening} + z_2 + z_{elevation} + z_{dim,opening}
\end{aligned}
$$

Formeln A.3.: Transformationsvorschriften für die IfcDirection (-1, 0, 0).

$$
\begin{aligned}
X_{1,fds} &= x_0 - x_1 \\
X_{2,fds} &= x_0 - x_1 - x_{dim} \\
Y_{1,fds} &= y_0 - y_1 \\
Y_{2,fds} &= y_0 - y_1 - y_{dim} \\
Z_{1,fds} &= z_0 + z_1 + z_{elevation} \\
Z_{2,fds} &= z_0 + z_1 + z_{elevation} + z_{dim}
\end{aligned}
$$

$$
\begin{aligned}
X_{1,fds} &= x_{0,wall} - x_{1,wall} - x_{1,opening} - x_2 \\
X_{2,fds} &= x_{0,wall} - x_{1,wall} - x_{1,opening} - x_2 - x_{dim,opening} \\
Y_{1,fds} &= y_{0,wall} - y_{1,wall} \\
Y_{2,fds} &= y_{0,wall} - y_{1,wall} - y_{dim,wall} \\
Z_{1,fds} &= z_{0,wall} + z_{1,wall} + z_{1,opening} + z_2 + z_{elevation} \\
Z_{2,fds} &= z_{0,wall} + z_{1,wall} + z_{1,opening} + z_2 + z_{elevation} + z_{dim,opening}
\end{aligned}
$$

Formeln A.4.: Transformationsvorschriften für die IfcDirection (0, -1, 0).

$$
\begin{aligned}
X_{1,fds} &= x_0 + y_1 \\
X_{2,fds} &= x_0 + y_1 + y_{dim} \\
Y_{1,fds} &= y_0 - x_1 \\
Y_{2,fds} &= y_0 - x_1 - x_{dim} \\
Z_{1,fds} &= z_0 + z_1 + z_{elevation} \\
Z_{2,fds} &= z_0 + z_1 + z_{elevation} + z_{dim}
\end{aligned}
$$

$$
\begin{aligned}
X_{1,fds} &= x_{0,wall} + y_{1,wall} \\
X_{2,fds} &= x_{0,wall} + y_{1,wall} + y_{dim,wall} \\
Y_{1,fds} &= y_{0,wall} - x_{1,wall} - x_{1,opening} - x_2 \\
Y_{2,fds} &= y_{0,wall} - x_{1,wall} - x_{1,opening} - x_2 - x_{dim,opening} \\
Z_{1,fds} &= z_{0,wall} + z_{1,wall} + z_{1,opening} + z_2 + z_{elevation} \\
Z_{2,fds} &= z_{0,wall} + z_{1,wall} + z_{1,opening} + z_2 + z_{elevation} + z_{dim,opening}
\end{aligned}
$$

A.3. Anhang zum Multimodell mit dem Variationsprinzip

Quelltext A.3: EXPRESS Schema für das Fachmodell fire protection concept

```
-------------------------------------------------------------------------------------------
SCHEMA fire_protection_concept;                                                          --
-------------------------------------------------------------------------------------------
-- ENTITAETEN  ----------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
ENTITY      Fire_Protection_Infrastructure;                                              --
            referred_to:                                        class_of_building;       --
            factual_feature:                OPTIONAL            special_building;        --
            quantity_of:                    LIST [1:?] OF       Unit_Of_Use;             --
            consists_of:                    LIST [1:?] OF       Component_Request;       --
END_ENTITY;                                                                              --
                                                                                         --
ENTITY      Unit_Of_Use;                                                                 --
            is_used_as:                                         utilization;             --
            gross_floor_area:                                   INTEGER;                 --
END_ENTITY;                                                                              --
                                                                                         --
ENTITY      Component_Request                                                            --
ABSTRACT SUPERTYPE OF (ONEOF (Room_Closing_Component_Request,                            --
                              Load_Bearing_Component_Request,                            --
                              RC_LB_Component_Request));                                 --
            id:                                                 STRING;                  --
            fyi:                                OPTIONAL        STRING;                  --
            owns:                                               technical_designation;   --
            referred_din:                                       din_4102_designation;    --
            characterised_by:                                   rgb;                     --
END_ENTITY;                                                                              --
                                                                                         --
ENTITY      rgb;                                                                         --
            r_value:                                            rgb_value;               --
            g_value:                                            rgb_value;               --
            b_value:                                            rgb_value;               --
END_ENTITY;                                                                              --
                                                                                         --
ENTITY      RC_LB_Component_Request             SUBTYPE OF     (Component_Request);      --
            relates_to:                                         component_group1;        --
            referred_din_en:                                    din_en_13501_2_rei;      --
END_ENTITY;                                                                              --
                                                                                         --
ENTITY      Room_Closing_Component_Request      SUBTYPE OF     (Component_Request);      --
            relates_to:                                         component_group1;        --
            referred_din_en:                                    din_en_13501_2_ei;       --
END_ENTITY;                                                                              --
                                                                                         --
ENTITY      Load_Bearing_Component_Request      SUBTYPE OF     (Component_Request);      --
            relates_to:                                         component_group2;        --
            referred_din_en:                                    din_en_13501_2_r;        --
END_ENTITY;                                                                              --
-------------------------------------------------------------------------------------------
-- TYPEN ----------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
TYPE   class_of_building                      = ENUMERATION OF (GK_1a,GK_1b,             --
                                                                GK_2,                    --
                                                                GK_3,                    --
```

```
                                                    GK_4,                  --
                                                    GK_5);                 --
END_TYPE;                                                                  --
                                                                           --
TYPE   special_building                  = ENUMERATION OF (Nr_1_Hochhauser,--
                                                    Nr_2_Hoehe_30m,        --
                                                    Nr_3_Gebaude_1600m2,   --
                                                    Nr_4_Verkaufstaetten,  --
                                                    Nr_5_weitere);         --
END_TYPE;                                                                  --
                                                                           --
TYPE   utilization                       = ENUMERATION OF (Wohnung,        --
                                                    Buero,                 --
                                                    Praxis,                --
                                                    weitere);              --
END_TYPE;                                                                  --
                                                                           --
TYPE   technical_designation             = ENUMERATION OF (feuerhemmend,   --
                                                    hochfeuerhemmend,      --
                                                    feuerbestaendig);      --
END_TYPE;                                                                  --
                                                                           --
TYPE   din_4102_designation              = ENUMERATION OF (F_30B,F30A,     --
                                                    F_60AB,F60A,           --
                                                    F_90AB,F90A);          --
END_TYPE;                                                                  --
                                                                           --
TYPE   component_group1                  = ENUMERATION OF (Decke,Wand);    --
END_TYPE;                                                                  --
                                                                           --
TYPE   component_group2                  = ENUMERATION OF (Stuetze,Unterzug);--
END_TYPE;                                                                  --
                                                                           --
TYPE   din_en_13501_2_rei                = ENUMERATION OF (REI_30,REI_60,REI_90);--
END_TYPE;                                                                  --
                                                                           --
TYPE   din_en_13501_2_ei                 = ENUMERATION OF (EI_30,EI_60,EI_90);--
END_TYPE;                                                                  --
                                                                           --
TYPE   din_en_13501_2_r                  = ENUMERATION OF (R_30,R_60,R_90);--
END_TYPE;                                                                  --
TYPE rgb_value                           = INTEGER;                        --
   WHERE                                   WR1:           {0 = SELF = 255 };--
END_TYPE;                                                                  --
-----------------------------------------------------------------------------------------
END_SCHEMA;                                                                --
-----------------------------------------------------------------------------------------
```

Quelltext A.4: EXPRESS Schema für das Fachmodell initial fire

```
-------------------------------------------------------------------------------------------
SCHEMA initial_fire;                                                                      --
-------------------------------------------------------------------------------------------
-- ENTITAETEN  ----------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
ENTITY      Fire_Event                                                                    --
ABSTRACT SUPERTYPE OF (ONEOF (Load_Controlled_Fire,                                       --
                              Fire_Experiment));                                          --
            id:                                              STRING;                      --
            fyi:                                OPTIONAL     STRING;                      --
            heat_release_rate_per_area:                      REAL;                        --
            fire_development_time:                           INTEGER;                     --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      Load_Controlled_Fire                SUBTYPE OF   (Fire_Event );               --
            refers_to:                                       utilization;                 --
            fire_load_density:                               INTEGER;                     --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      Fire_Experiment                     SUBTYPE OF   (Fire_Event );               --
            refers_to:                          LIST [1:?] OF  Fire_Load;                 --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      Fire_Load;                                                                    --
            description:                                      STRING;                     --
            source_information:                               STRING;                     --
            calorific_value:                                  REAL;                       --
            quantity:                                         REAL;                       --
            owns:                                             unit_of_quantity;           --
            has:                                              mass_factor;                --
            classified_into:                                  category;                   --
END_ENTITY;                                                                               --
-------------------------------------------------------------------------------------------
-- TYPEN ----------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
TYPE        utilization                          = ENUMERATION OF (Wohngebaude,           --
                                                                   Buerogebauede,         --
                                                                   Krankenhaus_Zimmer,    --
                                                                   Hotel_Zimmer,          --
                                                                   Bibliothek,            --
                                                                   Schule_Klassenzimmer,  --
                                                                   Verkaufstaette,        --
                                                                   Versammlungsstaette);  --
END_TYPE;                                                                                 --
                                                                                          --
TYPE        unit_of_quantity                   = ENUMERATION OF  (kg, m,Stck);            --
END_TYPE;                                                                                 --
TYPE        mass_factor                        = REAL;                                    --
WHERE                                            WR1:           {0. = SELF = 1. };        --
END_TYPE;                                                                                 --
                                                                                          --
TYPE        category                           = ENUMERATION OF (stoff_gegenstand,        --
                                                                 plaste_elaste,           --
                                                                 moebel_einrichtung,      --
                                                                 kabel_leitung);          --
END_TYPE;                                                                                 --
END_SCHEMA;                                                                               --
```

Quelltext A.5: EXPRESS Schema für das Fachmodell material

```
-------------------------------------------------------------------------------------------
SCHEMA material;                                                                          --
-------------------------------------------------------------------------------------------
-- ENTITAETEN  ----------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
ENTITY      Material                                                                      --
SUPERTYPE OF                                                    (Pyrolysis_Material);     --
            id:                                                  STRING;                  --
            fyi:                                    OPTIONAL     STRING;                  --
            general_parameter:                                   density;                 --
            temperature_behaviour:                               behaviour_select;        --
            optional_parameter:                     OPTIONAL     emissivity;              --
END_ENTITY;                                                                               --
                                                                                          --
                                                                                          --
ENTITY      Pyrolysis_Material                      SUBTYPE OF   (Material);              --
            has_a_n_reactions:                                   n_reactions;             --
            has_a_spec_id:                                       spec_id;                 --
            has_a_nu_spec:                                       nu_spec;                 --
            has_a_reference_temperature:                         reference_temperature;   --
            has_a_heat_of_reaction:                              heat_of_reaction;        --
            has_a_heat_of_combustion:                            heat_of_combustion;      --
                                                                                          --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      Temperature_Independent_Material;                                             --
            has_a_specific_heat:                                 specific_heat;           --
            has_a_conductivity:                                  conductivity;            --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      Temperature_Dependent_Material;                                               --
            specific_heat_ramp_id:                               STRING;                  --
            conductivity_ramp_id:                                STRING;                  --
            has_a_specific_heat_ramp:               LIST [1:?] OF Specific_Heat_Ramp;     --
            has_a_conductivity_ramp:                LIST [1:?] OF Conductivity_Ramp;      --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      Specific_Heat_Ramp;                                                           --
            temperature:                                         t;                       --
            functional:                                          f;                       --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      Conductivity_Ramp;                                                            --
            temperature:                                         t;                       --
            functional:                                          f;                       --
END_ENTITY;                                                                               --
-------------------------------------------------------------------------------------------
-- TYPEN ----------------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
TYPE        behaviour_select                        = SELECT (Temperature_Independent_Material,
                                                              Temperature_Dependent_Material);
END_TYPE;                                                                                 --
                                                                                          --
TYPE        density                                 = REAL;                               --
END_TYPE;                                                                                 --
                                                                                          --
TYPE        specific_heat                           = REAL;                               --
```

```
END_TYPE;                                                                                --
                                                                                         --
TYPE        conductivity                        = REAL;                                  --
END_TYPE;                                                                                --
                                                                                         --
TYPE        emissivity                          = REAL;                                  --
END_TYPE;                                                                                --
                                                                                         --
TYPE        t                                   = REAL;                                  --
END_TYPE;                                                                                --
                                                                                         --
TYPE        f                                   = REAL;                                  --
END_TYPE;                                                                                --
                                                                                         --
TYPE        n_reactions                         = INTEGER;                               --
END_TYPE;                                                                                --
                                                                                         --
TYPE        spec_id                             = ENUMERATION OF (PROPANE, weitere);     --
END_TYPE;                                                                                --
                                                                                         --
TYPE        nu_spec                             = REAL;                                  --
WHERE                                             WR1:           {0. = SELF = 1.};       --
END_TYPE;                                                                                --
                                                                                         --
TYPE        reference_temperature               = REAL;                                  --
END_TYPE;                                                                                --
                                                                                         --
TYPE        heat_of_reaction                    = REAL;                                  --
END_TYPE;                                                                                --
                                                                                         --
TYPE        heat_of_combustion                  = REAL;                                  --
END_TYPE;                                                                                --
-----------------------------------------------------------------------------------------
END_SCHEMA;                                                                              --
-----------------------------------------------------------------------------------------
```

Quelltext A.6: EXPRESS Schema für ein Linkmodell

```
------------------------------------------------------------------------------------------
SCHEMA linkmodel;                                                                       --
------------------------------------------------------------------------------------------
-- ENTITAETEN  -----------------------------------------------------------------------------
------------------------------------------------------------------------------------------
ENTITY      Linkmodel;                                                                  --
            name:                                 OPTIONAL       STRING;                --
            description:                          OPTIONAL       STRING;                --
            has_a_number_of:                      LIST [1:?] OF  Link;                  --
END_ENTITY;                                                                             --
                                                                                        --
ENTITY Link;                                                                            --
            consists_of:                          LIST [2:?] OF  Referenze;             --
END_ENTITY;                                                                             --
                                                                                        --
ENTITY      Referenze;                                                                  --
            references_to:                                       data_model;            --
            contains:                                            identifier;            --
END_ENTITY;                                                                             --
------------------------------------------------------------------------------------------
-- TYPEN -----------------------------------------------------------------------------------
------------------------------------------------------------------------------------------
TYPE        data_model                                         = STRING;                --
END_TYPE;                                                                               --
                                                                                        --
TYPE        identifier                                         = STRING;                --
END_TYPE;                                                                               --
------------------------------------------------------------------------------------------
END_SCHEMA;                                                                             --
------------------------------------------------------------------------------------------
```

Quelltext A.7: EXPRESS Schema für das FDS-Simulationsmodell

```
-------------------------------------------------------------------------------------------
SCHEMA simulation_model;                                                                  --
-------------------------------------------------------------------------------------------
-- ENTITAETEN  ----------------------------------------------------------------------------
-------------------------------------------------------------------------------------------
ENTITY      Root                                 ABSTRACT SUPERTYPE;                      --
          identification:                                      id;                        --
          description:                          OPTIONAL       fyi;                       --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      FDS_Simulation_Model                 SUBTYPE OF     (root);                   --
            refers_to:                                           Simulation_Profile;      --
            owns:                                LIST [1:?] OF   Building_Element;        --
            refers_to_a:                                         Fire_Development;        --
            has:                                 LIST [1:?] OF   Flammable_Objects;       --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      Simulation_Profile;                                                           --
            mission:                                             HEAD_Instruction;        --
            maximum_time:                                        TIME_Instruction;        --
            space:                                               MESH_Instruction;        --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      HEAD_Instruction;                                                             --
            chid:                                                STRING;                  --
            title:                                               STRING;                  --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      TIME_Instruction;                                                             --
            t_end:                                               REAL;                    --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      MESH_Instruction                     SUBTYPE OF     (root);                   --
            subdivision:                                         ijk;                     --
            geometry:                                            XB;                      --
            representation:                                      rgb;                     --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      Building_Element;                                                             --
            selection:                                           element_select;          --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      OBST_Instruction                     SUBTYPE OF     (root);                   --
            selection:                                           attributes_select;       --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      Object_Attributes;                                                            --
            geometry:                                            xb;                      --
            representation1:                                     rgb;                     --
            representation2:                                     transparency;            --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      HOLE_Instruction                     SUBTYPE OF     (root);                   --
            geometry:                                            xb;                      --
END_ENTITY;                                                                               --
                                                                                          --
ENTITY      Flammable_Object_Attributes;                                                  --
```

```
            geometry:                                       xb;                     --
            features:                                       surf_id;                --
END_ENTITY;                                                                         --
                                                                                    --
ENTITY      Fire_Development;                                                       --
            refers_to_a:                                    OBST_Instruction;       --
            especially_for:                                 VENT_Instruction;       --
            properties:                                     SURF_Instruction;       --
END_ENTITY;                                                                         --
                                                                                    --
ENTITY      VENT_Instruction;                                                       --
            geometry:                                       XB;                     --
            features:                                       surf_id;                --
END_ENTITY;                                                                         --
                                                                                    --
ENTITY      SURF_Instruction                SUBTYPE OF      (root);                 --
            selection:                                      properties_select;      --
END_ENTITY;                                                                         --
                                                                                    --
ENTITY      Fire_Development_Properties;                                            --
            representation:                                 rgb;                    --
            hhrpua:                                         REAL;                   --
            fuel:                                           STRING;                 --
            soot_yield:                                     REAL;                   --
END_ENTITY;                                                                         --
                                                                                    --
ENTITY      Flammable_Object_Properties;                                            --
            representation:                                 rgb;                    --
            thickness:                                      REAL;                   --
            features:                                       matl_id;                --
            burn_away:                                      BOOLEAN;                --
END_ENTITY;                                                                         --
                                                                                    --
ENTITY      Flammable_Objects;                                                      --
            refers_to_a:                                    OBST_Instruction;       --
            properties:                                     SURF_Instruction;       --
            owns:                                           MATL_Instruction;       --
END_ENTITY;                                                                         --
                                                                                    --
ENTITY      MATL_Instruction                SUBTYPE OF      (root);                 --
            spec_id:                                        STRING;                 --
            density:                                        REAL;                   --
            specific_heat:                                  REAL;                   --
            conductivity:                                   REAL;                   --
            heat_of_reaction:                               REAL;                   --
            heat_of_combustion:                             REAL;                   --
            reference_temperature:                          REAL;                   --
            n_reations:                                     INTEGER;                --
            nu_spec:                                        REAL;                   --
END_ENTITY;                                                                         --
-------------------------------------------------------------------------------------
-- TYPEN ----------------------------------------------------------------------------
-------------------------------------------------------------------------------------
TYPE        attributes_select                 = SELECT    (Object_Attributes,       --
                                                           Flammable_Object_Attributes);
END_TYPE;                                                                           --
                                                                                    --
TYPE        element_select                    = SELECT    (OBST_Instruction,        --
                                                           HOLE_Instruction);       --
```

```
END_TYPE;                                                                                --
                                                                                         --
TYPE        properties_select                    = SELECT     (Fire_Development_Properties,
                                                               Flammable_Object_Properties);
END_TYPE;                                                                                --
                                                                                         --
TYPE        ijk                                  = LIST [3:3] OF INTEGER;               --
END_TYPE;                                                                                --
                                                                                         --
TYPE        matl_id                              =              STRING;                 --
END_TYPE;                                                                                --
                                                                                         --
                                                                                         --
TYPE        surf_id                              =              STRING;                 --
END_TYPE;                                                                                --
                                                                                         --
TYPE        xb                                   = LIST [6:6] OF REAL;                  --
END_TYPE;                                                                                --
                                                                                         --
TYPE        transparency                         =               REAL;                  --
WHERE                                              WR1:          {0. = SELF = 1.};      --
END_TYPE;                                                                                --
                                                                                         --
TYPE        rgb                                  = LIST [3:3] OF INTEGER;               --
WHERE                                              WR1:          {1 = SELF = 255};      --
END_TYPE;                                                                                --
                                                                                         --
TYPE        id                                                 = STRING;                --
END_TYPE;                                                                                --
                                                                                         --
TYPE        fyi                                                = STRING;                --
END_TYPE;                                                                                --
-----------------------------------------------------------------------------------------
END_SCHEMA;                                                                              --
-----------------------------------------------------------------------------------------
```

B. Compact Disk