Im Rahmen des sich derzeit vollziehenden Wandels von der segmentierten, zeichnungsorientierten zur integrierten, modellbasierten Arbeitsweise bei der Planung von Bauwerken und ihrer Erstellung werden Computermodelle nicht mehr nur für die physikalische Simulation des Bauwerksverhaltens, sondern auch zur Koordination zwischen den einzelnen Planungsdisziplinen und Projektbeteiligten genutzt. Die gemeinsame Erstellung und Nutzung dieses Modells zur virtuellen Abbildung des Bauwerks und seiner Erstellungsprozesse, das sogenannten Building Information Modeling (BIM), ist dabei zentraler Bestandteil der Planung. Die Integration der Terminplanung in diese Arbeitsweise erfolgt bisher jedoch nur unzureichend, meist lediglich in der Form einer nachgelagerten 4D-Simulation zur Kommunikation der Planungsergebnisse.

Gegenstand der vorliegenden Arbeit ist die tiefere Einbettung der Terminplanung in die modellbasierte Arbeitsweise. Auf Basis einer umfassenden Analyse der Rahmenbedingungen und des Informationsbedarfs der Terminplanung werden Konzepte zur effizienten Wiederverwendung von im Modell gespeicherten Daten mit Hilfe einer Verknüpfungssprache, zum umfassenden Datenaustausch auf Basis der Industry Foundation Classes (IFC) und für das Änderungsmanagement mittels einer Versionierung auf Objektebene entwickelt. Die für die modellbasierte Terminplanung relevanten Daten und ihre Beziehungen zueinander werden dabei formal beschrieben sowie die Kompatibilität ihrer Granularität durch eine Funktionalität zur Objektteilung sichergestellt. Zur zielgenauen Extraktion von Daten werden zudem Algorithmen für räumliche Anfragen entwickelt.

ISBN: 978-3-86068-416-0

Herausgegeben von der Professur Informatik im Bauwesen, Verlag der Bauhaus-Universität Weimar

4

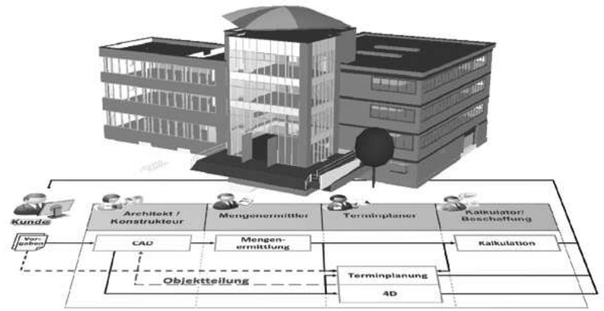
Jan Tulke Kollaborative Terminplanung auf Basis von Bauwerksinformationsmodellen

4

Informatik
in Architektur und Bauwesen

Jan Tulke

Kollaborative Terminplanung auf Basis von Bauwerksinformationsmodellen



function({Θ : filterList -> extract }, ...)

Bauhaus-Universität Weimar

Informatik
in Architektur und Bauwesen 4
Hrsg. von der Professur Informatik im Bauwesen

Jan Tulke

Kollaborative Terminplanung auf Basis von Bauwerksinformationsmodellen

Verlag der Bauhaus-Universität Weimar

Impressum

Schriftenreihe Informatik in Architektur und Bauwesen
Herausgegeben von der Professur Informatik im Bauwesen an der Bauhaus-Universität Weimar

Jan Tulke
Kollaborative Terminplanung auf Basis von Bauwerksinformationsmodellen

Umschlag: Katrin Wender

Druck: docupoint Magdeburg GmbH

Verlag der Bauhaus-Universität Weimar 2010

ISBN: 978-3-86068-416-0

Bestellungen:
verlag@uni-weimar.de
Fax: 03643/581156

Diese Publikation steht auch online zur Verfügung unter
http://e-pub.uni-weimar.de/volltexte/2010/1513

Kollaborative Terminplanung auf Basis von Bauwerksinformationsmodellen

Dissertation zur Erlangung des akademischen Grades

Doktor - Ingenieur

an der Fakultät Bauingenieurwesen
der
Bauhaus-Universität Weimar

vorgelegt von

Jan Tulke
geboren am 23. April 1978
in Berlin

Gutachter: 1. Prof. Dr.-Ing. Karl Beucke, Bauhaus-Universität Weimar

2. Prof. Dr.-Ing. Peter Racky, Technische Universität Darmstadt

3. Prof. Dr.-Ing. Markus König, Ruhr-Universität Bochum

Eingereicht am: 25. Oktober 2009

Tag der Disputation: 30. April 2010

Vorwort

Die vorliegende Arbeit entstand in der Zeit von Juni 2006 bis Oktober 2009 im Rahmen einer externen Promotion am Lehrstuhl Informatik im Bauwesen der Bauhaus-Universität Weimar. Die thematische Bearbeitung war im Zusammenhang mit meiner Projektleitertätigkeit für die HOCHTIEF AG in das vier Jahre laufende EU Forschungsprojekt „Open Information Environment for knowledge-based collaborative Processes throughout the lifecycle of a building“ (InPro) innerhalb des sechsten EU-Forschungsrahmenprogramms eingebettet. Eine vorhergehende, über dreijährige Berufstätigkeit in der Angebots- und Ausführungsplanung von Großprojekten im Tief-, Tunnel-, Wasserbau bei HOCHTIEF Consult Infrastructure und die damit verbundene Berufserfahrung sowie vorhandenen Kontakte zu praktisch tätigen Fachleuten kamen mir bei der Bearbeitung dieser Dissertation sehr zugute. Hierzu zählte insbesondere auch der direkte Zugriff auf in der Praxis gängige, kommerzielle Softwarepakete sowie auf reale Projektdaten aus dem Bereich virtuellen Bauens der HOCHTIEF ViCon GmbH, bei der ich über die gesamte Bearbeitungszeit hinweg meinen festen Arbeitsplatz hatte. Durch meine Lehrtätigkeit für die HOCHTIEF Akademie im Fach Bauinformatik war es mir darüber hinaus möglich, ein unmittelbares Gefühl für die Bedürfnisse und Anforderungen von Baupraktikern an unterstützende Softwareanwendungen zu erlangen.
Mit der modellbasierten Terminplanung kam ich erstmalig im Herbst 2005 im Rahmen eines Pilotprojektes zur Aufbereitung eines Bauzeitennachtrags mit Hilfe von 4D-Simulationen in Kontakt. Kern des Projektes war die vergleichende Visualisierung mehrere Terminplanalternativen für die Bauausführung auf Basis eines einzigen, dreidimensionalen CAD Modells. Die damalige breite Akzeptanz und hohe Begeisterung für die Projektergebnisse in Verbindung mit den zahlreich angetroffenen Schwierigkeiten und unausgereiften Softwarefunktionalitäten motivierten mich, diese Technologie, trotz der damals schon breiten Widmung in der Forschung, weiter zu verfolgen und ihre konzeptionelle Effizienz und damit Praxistauglichkeit durch neue Ansätze entscheidend zu verbessern.
Mein besonderer Dank gilt meinem Doktorvater Herrn Prof. Dr.-Ing Karl Beucke, der bzgl. einer zielgerichteten Durchführung meiner Promotion stets ein wertvoller Ansprechpartner war, sowie der HOCHTIEF AG und hier insbesondere Herrn Wolfgang Katzer für die mir entgegengebrachte persönliche, finanzielle und organisatorische Unterstützung während der gesamten Bearbeitungszeit. Weiterhin danke ich allen Mitarbeitern am Lehrstuhl, die mir stets mit Rat und Tat zu Seite standen, sowie den beiden externen Gutachtern Prof. Dr.-Ing. Peter Racky, Leiter des Institut für Baubetriebswirtschaft an der Universität Kassel, und Prof. Dr.-Ing Markus König, Leiter des Lehrstuhls für Informatik im Bauwesen an der Ruhr-Universität Bochum, für die Erstellung der Gutachten.
Bei meiner zukünftigen Frau, meiner Familie und meinen Freunden bedanke ich mich außerordentlich für das mir entgegengebrachte Verständnis für den langjährigen Zeitmangel und die häufige Trennung, verursacht durch die nicht unbedeutenden Entfernungen zwischen meinen Wohn- und Arbeitsorten Essen, Weimar, Berlin und Herbrechtingen.

Essen, im Mai 2010
Jan Tulke

Kurzfassung

Im Rahmen des sich derzeit vollziehenden Wandels von der segmentierten, zeichnungsorientierten zur integrierten, modellbasierten Arbeitsweise bei der Planung von Bauwerken und ihrer Erstellung werden Computermodelle nicht mehr nur für die physikalische Simulation des Bauwerksverhaltens, sondern auch zur Koordination zwischen den einzelnen Planungsdisziplinen und Projektbeteiligten genutzt. Die gemeinsame Erstellung und Nutzung dieses Modells zur virtuellen Abbildung des Bauwerks und seiner Erstellungsprozesse, das sog. Building Information Modeling (BIM), ist dabei zentraler Bestandteil der Planung. Die Integration der Terminplanung in diese Arbeitsweise erfolgt bisher jedoch nur unzureichend, meist lediglich in der Form einer nachgelagerten 4D-Simulation zur Kommunikation der Planungsergebnisse. Sie weist damit im Verhältnis zum entstehenden Zusatzaufwand einen zu geringen Nutzen für den Terminplaner auf.

Gegenstand der vorliegenden Arbeit ist die tiefere Einbettung der Terminplanung in die modellbasierte Arbeitsweise. Auf Basis einer umfassenden Analyse der Rahmenbedingungen und des Informationsbedarfs der Terminplanung werden Konzepte zur effizienten Wiederverwendung von im Modell gespeicherten Daten mit Hilfe einer Verknüpfungssprache, zum umfassenden Datenaustausch auf Basis der Industry Foundation Classes (IFC) und für das Änderungsmanagement mittels einer Versionierung auf Objektebene entwickelt. Die für die modellbasierte Terminplanung relevanten Daten und ihre Beziehungen zueinander werden dabei formal beschrieben sowie die Kompatibilität ihrer Granularität durch eine Funktionalität zur Objektteilung sichergestellt. Zur zielgenauen Extraktion von Daten werden zudem Algorithmen für räumliche Anfragen entwickelt.

Die vorgestellten Konzepte und ihre Anwendbarkeit werden mittels einer umfangreichen Pilotimplementierung anhand von mehreren Praxisbeispielen demonstriert und somit deren praktische Relevanz und Nutzen nachgewiesen.

Abstract

In context of the current change from a segmented, drawing based to an integrated, model based planning approach for buildings and their construction processes, computer models are no longer used for physical simulations of structural characteristics only, but also as basis for coordination between different design disciplines and stakeholders. Thereby the collaborative compilation and utilisation of such a model which serves as virtual representation, the so called Building Information Modeling (BIM), is within the core of the new planning approach. But currently, construction sequence planning is only insufficiently integration in this new approach and mostly limited to a subsequent 4D-Simulation for communication of scheduling results. Consequently the accruing additional effort for the model use cannot be fully justified by the resulting benefits.

Therefore subject of the present thesis is a deeper integration of construction sequence planning into the model based way of working. Starting from a comprehensive analysis of general conditions and information needs during scheduling, concepts of a linking language for efficient reuse of model data, complete data exchange based on Industry Foundation Classes (IFC) and change management based on object versioning are presented. Data used during scheduling and its inherent relationships are described formally and a functionality of object splitting is developed to ensure the compatibility of data granularity. For the sake of target precise data extraction, algorithms of spatial queries are presented.

All concepts are implemented in an extensive software package and their practical application is demonstrated by several examples.

Inhaltsverzeichnis

1 Einleitung

> *„Whithout a doubt, BIM has arrived, and everyone's business will be affected. We are entering the most transformative time our industry has ever experienced."*
>
> von Stephen A. Jones aus [Young u. a. 2008, S. 21]

1.1 Computermodelle im Bauwesen

Computer und zugehörige Datenmodelle werden im Bauwesen schon seit längerer Zeit eingesetzt. Tatsächlich geht sogar die Entwicklung des Computers selbst auf den Bauingenieur Konrad Zuse zurück, der mit seiner 1938 fertig gestellten, elektronisch angetriebenen, aber dennoch mechanischen Rechenmaschine Z1 die Berechnung von Statiken für den Flugzeugbau automatisieren wollte. Die 1941 fertiggestellte Nachfolgeversion Z3 war der erste voll funktionsfähige, programmgesteuerte Rechner der Welt. Er basierte bereits auf dem binären Zahlensystem. Später wurde die zunächst im Fahrzeugbau und in der Luft- und Raumfahrtindustrie entwickelte Finite Element Methode (FEM[1]) auch im Bauwesen eingeführt und fortan für die strukturmechanische Berechnung von Tragwerken eingesetzt. Das erste Lehrbuch zur FEM „The Finite Element Method in Structural and Continuum Mechanics" wurde 1967 vom Mathematiker O.C. Zienkiewicz veröffentlicht, der in der Abteilung Bauingenieurwesen der Universität von Wales in Swansea lehrte.
Mit Einführung der ersten Heimcomputer erhielt Mitte der 1980er Jahre dann auch das computerunterstützte Konstruieren (CAD[1]) Einzug in die Planungsbüros und -abteilungen der Baufirmen. Neben den physikalischen FEM Berechnungen wurde der Computer somit nun auch für planerische Tätigkeiten genutzt. Das heute noch sehr verbreitete AutoCAD erschien 1982 unter dem Betriebssystem DOS und stand auch für den Einsatz in kleineren Planungsbüros zu Verfügung. Das Vorgehen bei der Konstruktion blieb jedoch trotz der Computerunterstützung gegenüber der papierbasierten Planung zunächst unverändert. Es wurden weiterhin einfache Striche anstelle von Bauteilobjekten gezeichnet, d.h. die Semantik[1] der dargestellten geometrischen Formen wurde nicht im Computer hinterlegt und konnte somit nur vom Menschen interpretiert werden. Auch blieben die einzelnen Pläne weiterhin getrennte Dokumente, die weder untereinander noch mit anderen Planungsdokumenten verknüpft waren. Vorteile aus der Einführung des CAD-basierten Planens ergaben sich jedoch bereits aus der Erstellung sauberer Zeichnungen, der besseren Änderungsmöglichkeit und der leichteren Erstellung von Planungsalternativen. Ein guter Überblick über die Entwicklungsgeschichte des CAD und die zugehörigen Computermodelle sowie Datenaustauschstandards wird

[1] siehe Glossar

in [Eastman 1999] gegeben.
Mit der Verfügbarkeit immer leistungsfähigerer Soft- und Hardware auf der einen und immer komplexeren Planungsaufgaben auf der anderen Seite gibt es seit Anfang der 2000er Jahre das Bestreben, die Planung vollständig auf ein integriertes Computermodell zu stützen. Die bisherige Erstellung der Planung in voneinander getrennten Dokumenten, von denen jedes nur einen Teilausschnitt des gesamten Designs darstellen kann, erfordert stets die explizite Interpretation durch den Menschen. Das Gesamtmodell kann nur mental in der Vorstellung des Betrachters zusammengefügt werden. Bei hunderten bis tausenden von verschiedenen Dokumenten bei einer üblichen Projektgröße, die zudem noch mehrere Revisionsstände durchlaufen, ist es offensichtlich, dass die Koordinierung der Dokumente zur Vermeidung von Planungsfehlern stets eine der größten Herausforderungen bei der Durchführung von Bauprojekten ist. Es besteht nun die Aussicht, dass die Koordination und Kommunikation zwischen den Projektbeteiligten durch die Nutzung eines integrierten Gebäudemodells mit dreidimensionaler Darstellung entscheidend erleichtert werden kann. In anderen Industrien, wie etwa dem Flugzeug-, Schiffs-, Automobil- und Anlagenbau, werden derzeit schon solche integrierten Computermodelle mit Erfolg eingesetzt. Eine Analyse der Gründe für den verzögerten Einsatz in der Bauindustrie ist in [Jaeger u. a. 2007] zu finden. Dabei stehen keineswegs technische Aspekte im Vordergrund.
Neben der erleichterten Koordination der Planung wird durch die zusätzliche Speicherung der Semantik der verwalteten Daten im Modell die Vernetzung von Informationen aus verschiedenen Fachbereichen und Quellen sowie eine automatisierte Auswertung der Planungsdaten ermöglicht. Zusätzlich wird durch die einheitliche Datenspeicherung die Wiederverwendung von Daten erleichtert bzw. gar erst ermöglicht. Bisher mussten Daten, die zu unterschiedlichen Zwecken verwendet wurden, aufgrund der inkompatiblen Datenhaltung der einzelnen Anwendung stets von Hand neu eingegeben werden. Somit kann durch die neue Datenintegration eine höhere Planungsqualität, aber letztlich auch eine Steigerung der Produktivität erreicht werden.
Der seit Kurzem viel verwendete Begriff Building Information Model(ing) (BIM[1]) beschreibt diesen Prozess der gemeinsamen Erzeugung und Nutzung eines digitalen Gebäudemodells für die Planung, Erstellung und den Betrieb von Bauwerken. Ein solches Modell enthält die präzise dreidimensionale Geometrie sowie weitere Daten, die zur Planung, Vorfertigung und Erstellung auf der Baustelle sowie der Beschaffung von Bedeutung sind (siehe hierzu auch [Eastman u. a. 2008]). Dieses bedeutet, dass Computermodelle nicht mehr nur zur Simulation des physikalischen Verhaltens von Bauwerken, sondern insbesondere auch zur Koordinierung und Unterstützung der Planung sowie zur Erstellung und zum Betrieb von Bauwerken verwendet werden. Dokumente sind von nun an nur noch Teilauszüge als Arbeitsunterlagen für die Unterstützung der Ausführung einzelner Aufgaben und nicht mehr alleinige Basis der Planung. Es ist somit zu erwarten, dass in Zukunft jedes Gebäude zweimal erstellt werden wird. Zunächst virtuell als digitales Modell zwecks Optimierung und später ein zweites Mal auf der Baustelle nach exakten Vorgaben des Modells und daher mit hoher Effizienz. Ziel ist es, das Gros an Planungsfehlern bereits im Vorfeld mittels des Modells zu erkennen und die Ausführung soweit zu optimieren, dass während der Errichtung keine unerwarteten Zusatzkosten mehr entstehen.
In [Young u. a. 2008] wurden 302 Personen aus Architektur- und Ingenieurebüros sowie

von Baufirmen und Auftraggebern, vorwiegend aus dem US amerikanischen Raum, bezüglich des derzeitigen Einsatzes von BIM auf realen Projekten befragt. Dabei nutzten 2008 bereits ein Drittel der Personen BIM in mindestens drei von fünf ihrer Projekte. Für 2009 wird dies sogar für fast die Hälfte der Befragten erwartet.
Bereits heute existiert eine Vielzahl von Softwareprogrammen, die zur Erstellung und Bearbeitung von BIM-Daten verwendet werden können. Eine Befragung im Jahr 2007 ([Young u. a. 2007]) ergab, dass BIM-kompatible Software bei 28% aller Befragten mittlerweile die am häufigsten eingesetzte Software ist. Damit liegt diese nach 2D-CAD- (57%) und Terminplanungssoftware (39%) bereits auf dem dritten Platz. Bemerkenswert ist, dass in der Gruppe der Baufirmen und Auftraggeber Terminplanungssoftware das am häufigsten eingesetzte Werkzeug ist. Dieses zeigt bereits die besondere Bedeutung der Terminplanung für diese Gruppen.
In den Umfragen werden als Haupteinsatzbereiche für BIM-kompatible Software die Mengenermittlung, die Terminplanung und die Kostenkalkulation genannt. Doch auch die automatische Konfliktortung auf Basis geometrischer Durchdringung (Clash Detection[1]) findet immer mehr Anwendung. Die in [Young u. a. 2007] genannten, tatsächlich eingetretenen Vorteile aus dem modellbasierten Arbeiten erscheinen vielfältig und entsprechen weitestgehend den zuvor beschriebenen Erwartungen. Im Einzelnen sind dieses:

- Eine verbesserte Kommunikation zwischen den Planungsbeteiligten.
- Eine einfachere Koordinierung der Planung und kostengünstigere Fehlerbehebung.
- Eine höhere Effektivität in der Planung, da mehr Zeit für die Planung selbst und weniger für die Dokumentation aufgewendet werden kann.
- Eine bessere Einhaltung von Qualität, Kosten und Terminen durch optimierte Planung.
- Ein positiver Einfluss auf die Erlangung von Aufträgen durch höhere Transparenz der technischen Kompetenz und bessere Marketing-Möglichkeiten.

Des Weiteren wird von in den Firmen gemessenen *Return on Investment*-Raten (ROI[1]) von 100-1000% berichtet. Doch obwohl durch die Nuzung von BIM der Fall der traditionellen Barrieren zwischen den in einem Projekt beteiligten Firmen erwartet wird, welches die gemeinsame Nutzung von Daten ermöglichen würde, berichten die Baufirmen, dass sie meistens noch ihr eigenes Modell auf Basis einer bereits existierenden 2D Planung erstellen, aber dennoch bereits profitieren.
Die Gruppe der Architekten scheint den Umfragen zufolge derzeit BIM am meisten einzusetzen und es wird ein weiterer Anstieg in dieser Gruppe erwartet, was bedeutet, dass auch für andere Projektbeteiligte später im Prozess in Zukunft häufiger die Möglichkeit bestehen wird, auf bereits existierende Modelle aufzusetzen. Doch hierbei ist zu bedenken, dass nicht nur die reine Existenz eines Modells Voraussetzung für dessen Wiederverwendung ist, sondern insbesondere auch dessen Datenqualität bzgl. Inhalt und Struktur. Hierfür steht mittlerweile eine Reihe von Richtlinien zur Verfügung. Eine entsprechende Liste findet man in [Succar 2009]. Diese Richtlinien basieren in der Regel auf einem Konzept, das den Reifegrad des Modells entsprechend den zugrunde

gelegten Planungsphasen in verschiedene Level einteilt. Aus dem Reifegradlevel werden zugehörige Mindestanforderungen an den Inhalt und die Struktur des Modells abgeleitet. Dabei ist auch zu beachten, dass die verschiedenen, an der Planung beteiligten Fachplaner zum Teil unterschiedliche Sichtweisen auf die Planungsdaten haben. So unterscheidet sich z.B. ein virtuelles Gebäudemodell für die Architekturplanung von dem für die Planung der Bauausführung in Konstruktionsdetails und in der Aufteilung des Gesamtmodells in einzelne Bauabschnitte. Die Gesamtheit der für die Zusammenarbeit benötigten Daten und der Zweck ihrer Verwendung ist daher zunächst zu definieren, und nur die gemeinsam genutzten Daten sind im zentralen Modell zu verwalten. Weitere, nur für die Arbeit in einer Fachdisziplin benötigte Informationen können dagegen separat gehalten werden ([Liebich u. a. 2009]).
Jede Fachdisziplin hat zudem einen gewissen Bedarf an Eingangsinformationen. Um diese zu befriedigen, ist es notwendig, die Zusammenarbeit in einem globalen Arbeitsprozess (Workflow[1]) sorgfältig zu definieren. Nur so kann sichergestellt werden, dass die benötigten Daten zur richtigen Zeit und in der erforderlichen Detaillierung zur Verfügung stehen. Hierzu haben sich sog. Workflow-Management-Systeme, meistens in Zusammenhang mit einer Dokumentenverwaltung, etabliert [Rueppel u. Klauer 2004], [König 2006]. Auch die Validierung der Daten und damit die Sicherung der Datenqualität kann durch entsprechende Werkzeuge für die automatische Konfliktortung (Clash Detection) unterstützt werden. Detaillierte Informationen hierzu und bzgl. der Organisation der Zusammenarbeit sind in [Benning u. a. 2009] zusammengestellt.
Einen weiteren wichtigen Aspekt für die Zusammenarbeit stellt die Definition von Datenzugriffs- und -hoheitsrechten dar. Es ist zu definieren, wer welche Informationen lesen, ändern und löschen darf. Dabei werden Änderungsrechte je nach thematischem Bezug der Daten einem Akteur des Planugsprozesses zugeordnet. Jeder Akteur ist nur zum Erzeugen oder Verändern der ihm zugeordneten Daten berechtigt. Ein Lesezugriff wird jedoch auch anderen Projektbeteiligten ermöglicht.
Trotz der zahlreichen bereits existierenden Möglichkeiten und positiven Praxisbeispielen liegen derzeit bei der Definition der Zusammenarbeit und dessen datentechnischer Organisation noch große Herausforderungen. In der Praxis scheint bisher noch keine Softwareplattform zu existieren, die allen Anforderungen der Datenverwaltung und -integration sowie der Koordination der Zusammenarbeit gerecht wird. Eine detailliertere Beschreibung der Anforderungen sowie einen prinzipiellen Entwurf einer solchen Plattform findet man in [Froese 2003], [Nummelin u. a. 2007] und [Liebich u. a. 2008].
Herausforderungen liegen nach wie vor auch im Austausch von Projektdaten selbst. Aufgrund der im Vergleich zu anderen Industrien starken Fragmentierung der Bau- und Bausoftwareindustrie kommen innerhalb eines Projektes oft zahlreiche unterschiedliche Programme zum Einsatz, deren Datenformate untereinander inkompatibel sind. Der Einsatz eines brancheneinheitlichen und herstellerunabhängigen Datenformats könnte dieses Problem beheben.
Abschließend kann gesagt werden, dass die Vorteile der gemeinschaftlichen Zusammenarbeit an einem integrierten Modell mittlerweile allgemein erkannt und zum Teil auch schon nutzbar gemacht werden. Eine vollständige Umstellung auf die neue Arbeitsweise in allen Bereichen und Phasen der Projektabwicklung ist jedoch derzeit noch nicht erfolgt. Der Wandel wird auch mangels leistungsfähiger Lösungen für einige der angerissenen Themen nur schrittweise vollzogen werden und bedarf einer für jede Fachdiszi-

plin spezifischen Adaption der gewohnten Arbeitsweisen und ihrer Randbedingungen. Einen sehr wichtigen Bereich für die erfolgreiche Durchführung von Bauprojekten stellt hierbei die Terminplanung dar.

1.2 Terminplanung im Bauwesen

1.2.1 Definition der Terminplanung

Der Begriff Terminplanung beschreibt den Prozess der Erstellung eines terminierten Projektablaufplans. Dieser gliedert das Projekt in einzelne Arbeitsvorgänge und weist diesen Zeitfenster und Ressourcen (Material, Personal, Werkzeuge und Maschinen) für deren Durchführung zu. Die Erstellung eines Terminplans kann in mehrere Schritte gegliedert werden:

1. Die *Unterteilung des Gesamtprojektes in Einzelvorgänge und die Bestimmung deren gegenseitiger Abhängigkeiten.* Hierzu gehören Vogänger- und Nachfolgerbeziehungen sowie Abstandszeiten. Diese können sich aus den durchzuführenden Aufgaben, der Konstruktion des Bauwerks, der gewählten Bauverfahren, der Limitierung gemeinsam genutzter Ressourcen sowie aus räumlichen, vertraglichen oder regulativen Zwangsbedingungen ergeben.
   Zur Kennzeichnung markanter Zeitpunkte im Projektverlauf werden sog. Meilensteine festgelegt. Diese kennzeichnen das Erreichen im Voraus definierter Teilziele, die für den Erfolg des Gesamtprojektes wesentlich sind.

2. Die *Bestimmung der Dauer der Einzelvorgänge.* Hierfür existieren, wie später im Einzelnen erläutert wird, verschiedene Vorgehensweisen. Prinzipiell ist die reale Vorgangsdauer jedoch durch den Aufwand der durchzuführenden Tätigkeit sowie der dafür zur Verfügung gestellten Ressourcen und deren Verfügbarkeit und Leistung bestimmt.
   Meilensteine bezeichnen einzelne Zeitpunkte und haben daher keine Dauer.

3. Automatische *Berechnung des Gesamtterminplans* unter Berücksichtigung der unter 1 und 2 definierten Randbedingungen und der Vorgabe des Projektstart- oder -endtermins.

Kommt es im Laufe der Projektbearbeitung zu Änderungen in den zugrunde gelegten Informationen, so ist eine Überprüfung und ggf. Anpassung des Terminplans erforderlich. Hierzu zählen Änderungen am geplanten Bauwerk selbst, der Art der Bauwerkserstellung oder der Verfügbarkeit von Ressourcen.
Auch unvorhergesehene Störungen im Projektverlauf können eine Anpassung des Terminplans erforderlich machen, um deren Auswirkungen auf das Projektergebnis zu begrenzen.

1.2.2 Bedeutung der Terminplanung

Der Terminplanung kommt im Bauwesen, wie auch in anderen Branchen mit einer vorwiegend projektbezogenen Arbeitsweise, eine entscheidende Rolle zu. Mit ihr wird

im Vorfeld der zeitliche Rahmen aller Projektaktivitäten festgelegt. Später, während der Projektdurchführung, dient der Terminplan dann als Grundlage für einen Soll-Ist-Vergleich und damit als Hauptkontroll- und -steuerungswerkzeug des Projektmanagements. Zudem werden während der Terminplanung unter Beachtung der Projektrahmenbedingungen Entscheidungen über die einzusetzenden Baugeräte und Personalressourcen getroffen. Dabei erfolgt eine Entscheidung stets nach der Maßgabe einer ausgewogenen Minimierung von Projektkosten und benötigter Abwicklungszeit bei gleichzeitiger Einhaltung der höchst möglichen bzw. der geforderten Qualität. Jegliche Abweichung von einem im Rahmen des Projektes ausgewogenen Optimums führt direkt oder indirekt (Vertragsstrafen, Nachbesserungen, verlängerter Personal- und Maschineneinsatz) zu Mehrkosten bzw. bei einer Angebotsbearbeitung zum Verlust des Auftrags. Somit sind alle drei Faktoren gleichwertige Einflussgrößen für den wirtschaftlichen Erfolg eines Bauprojektes (Abbildung 1.1), in denen die einzelnen Anbieter am Markt gegeneinander konkurrieren.

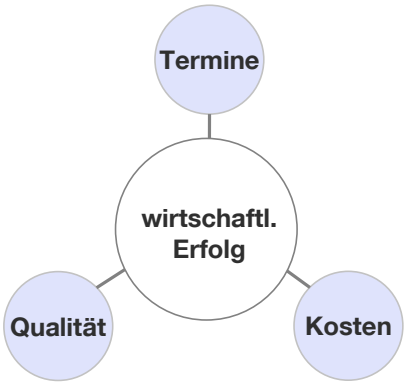


Abbildung 1.1: Komponenten des wirtschaftlichen Erfolgs eines Bauprojektes

Nach [Kochendörfer u. a. 2007] sind die Anforderungen an die Terminplanung und das Terminmanagement in den letzten Jahren stetig gewachsen. Immer kürzere Innovationszyklen innerhalb des Technologie- und Dienstleistungssektors führen dazu, dass Investoren, um sich am Immobilienmarkt behaupten zu können, Trends früh erkennen und mit einem adäquaten Angebot innerhalb kürzester Zeit beantworten müssen. Hieraus resultiert, dass inzwischen der Zeitpunkt des Markteintritts für Auftraggeber bzw. Investoren ein zentrales Investitionskriterium geworden ist und immer kürzere Realisierungsphasen erreicht werden müssen.
Doch auch für Bieter und Auftragnehmer spielt die Bauzeit eine wichtige Rolle, da Jahreszeit und Länge der Bauzeit wesentliche Kalkulationsfaktoren sind. Zudem stellen die mangelfreie Bauwerkserstellung und deren fristgerechte Fertigstellung rechtlich gleichrangige Verpflichtungen des Auftragnehmers dar ([Vygen u. a. 2002]).
Somit stellt die Terminplanung sowohl für Auftraggeber- als auch für Auftragnehmerseite, wenn auch mit anderen Schwerpunkten und in unterschiedlichen Detailstufen, ein entscheidendes Element des Planungsprozesses dar.

1.3 Ziel und Abgrenzung der Arbeit

Aufgrund der erläuterten Bedeutung der Terminplanung ist im Rahmen des sich derzeit vollziehenden Paradigmenwechsels von der segmentierten, papierbasierten zur integrierten, modellbasierten Planung auch in diesem Bereich eine Anpassung der existierenden Prozesse und zugehörigen Softwareumgebungen zu vollziehen. Ziel ist es dabei ebenfalls, von der neuen Arbeitsweise zu profitieren.
Im Gegensatz zur geometrischen Modellierung, die, getrieben durch den bereits signifikanten Einsatz von BIM in der Architekturplanung, schon durch geeignete CAD-Anwendungen unterstützt wird, ist im Bereich der Terminplanung bisher noch keine wesentliche Anpassung erfolgt. Zwar ist mit dem aufkommenden Wunsch nach Nutzung des 3D Architekturmodells für die Visualisierung der Bauablaufplanung der Markt für einen neuen Softwareprodukttyp, die sog. 4D-Simulationssoftware[2], entstanden, doch hat sich an der Arbeitsweise der Terminplanerstellung und dessen Fortschreibung im Projektverlauf nichts geändert. Vielmehr hat die alte Arbeitsweise ohne Einbeziehung von Informationen eines Bauwerksinformationsmodells nach wie vor Bestand und wurde lediglich durch einen zusätzlichen, nachgelagerten Prozess der Erstellung einer 4D-Simulation auf Grundlage eines existierenden 3D-Geometriemodells und des fertiggestellten Terminplans ergänzt.
Ziel dieser Arbeit ist es nun, die daten- und softwaretechnische Integration der Terminplanung in den kollaborativen, modellbasierten Planungsprozess von Bauprojekten herzustellen. Auf der Basis einer Analyse des Ist-Zustands wird zunächst der potentielle Nutzen der modellbasierten Arbeitsweise für den Terminplanungsprozess identifiziert und daraus werden Lösungsansätze abgeleitet. Dabei sollen etablierte und bewährte Arbeitsweisen sowie bereits im Einsatz befindliche Software nicht vollständig ersetzt, sondern zwecks erhöhter Akzeptanz in die neue Arbeitsweise integriert bzw. durch neue Konzepte und Softwarekomponenten ergänzt und damit effektiver gestaltet werden. Zudem soll eine Brücke zwischen der unternehmensweiten und der projektbezogenen Datenverarbeitung geschlagen werden.
Direkt angestrebtes Ziel und Schwerpunkt dieser Arbeit ist die Bereitstellung und einfache Nutzung von während der Terminplanung benötigten Basisinformationen direkt aus dem Bauwerksinformationsmodell. Von den drei in Kapitel 1.2.1 bereits erläuterten Schritten der Terminplanerstellung wird damit insbesondere die Ermittlung der Dauer von Einzelvorgängen (Schritt 2) unterstützt. Die Ermittlung und Erfassung von Abhängigkeiten zwischen Terminplanvorgängen (Schritt 1) ist ebenfalls wichtig für die Berechnung eines fehlerfreien, realistischen Terminplans. Dieses soll durch eine bereits *während* der Terminplanerstellung zur Verfügung stehende Visualisierung (4D-Simulation) zwecks zwischenzeitlicher Kontrolle unterstützt werden. Eine automatische Ableitung von im Terminplan zu erfassenden Vorgangsabhängigkeiten direkt aus dem Bauwerksinformationsmodell und dessen räumlich-topologischer Struktur ist hingegen nicht Ziel dieser Arbeit. In diesem Zusammenhang wird auf [Tauscher u. a. 2009], [Mikulakova u. a. 2008] und [de Vries u. Broekmaat 2007] verwiesen.
Wichtigste Voraussetzung für eine jederzeit verfügbare Visualisierung und die Bereit-

[2] In der Literatur hat sich der Begriff 4D-*Simulation* durchgesetzt, auch wenn es sich üblicherweise um eine Animation handelt. (siehe auch Glossar)

stellung von benötigten Teilinformationen aus dem Gesamtdatenbestand ist das kontinuierliche Fortschreiten der Verknüpfung von Terminplanvorgängen mit Elementen des Bauwerksmodells. Hierzu wird im Rahmen dieser Arbeit auf Basis einer Analogie zu bewährten Techniken des papierbasierten Arbeitens ein effektives und gleichzeitig flexibles Werkzeug in Form einer Verknüpfungssprache entwickelt. Da der Arbeitsschritt der Erstellung bzw. Aktualisierung von Verknüpfungen zwischen Terminplanvorgängen und Objekten des Bauwerksmodells, wie später noch erläutert wird, den Hauptarbeitsaufwand bei der Nutzung von Bauwerksinformationen für die Terminplanung darstellt, ist von einer Verbesserung in diesem Bereich eine beträchtliche Akzeptanzsteigerung der modellbasierten Arbeitsweise zu erwarten.
Die klassischen Verfahren zur Berechnung von Terminplänen (Schritt 3) sowie deren Optimierung hinsichtlich der durch Bauprozesse belegten Arbeitsräume und der Vermeidung einer ggf. verminderten Produktivität aus diesbezüglichen Konflikten zwischen Vorgängen wird im Rahmen dieser Arbeit nicht betrachtet. Ausführungen hierzu sind in [Akinci u. a. 2000], [Nigudkar 2005],[Jongeling u. a. 2005] und [Jongeling u. Olofsson 2007] zu finden.
Im Bereich des Datenaustauschs stellt die bisher fehlende Unterstützung eines einheitlichen Datenformats, das alle für die im Zusammenhang mit der Terminplanung auszutauschenden Daten unterstützt, ein Hindernis für die durchgängige modellbasierte Zusammenarbeit dar. Daher wird im Rahmen dieser Arbeit dargelegt, wie die einzelnen für die Terminplanung relevanten Daten unter den Projektbeteiligten auf Basis der Industry Foundation Classes (IFC[1]) ausgetauscht werden können. Des Weiteren wird zur Unterstützung der Reaktion auf Planungsänderungen ein Konzept für die Versionierung der Planungsdaten auf Objektebene vorgestellt. So können Änderungen in den zugrunde gelegten Planungsdaten zielgenau identifiziert und direkt auf einen evtl. Einfluss auf den Terminplan hin untersucht werden.
Insgesamt soll durch die vorliegende Arbeit die Nutzung eines digitalen Bauwerksmodells innerhalb des Terminplanungsprozess und damit ein Mehrwert gegenüber der heutigen Arbeitsweise erreicht werden. Dieses geschieht konzeptionell auf drei Arten:

1. Die *Förderung der Wiederverwendung von Daten* durch die Nutzung eines einheitlichen und herstellerunabhängigen Datenaustauschformats.

2. Die *Verringerung des erforderlichen Aufwands* für die Extraktion und Verknüpfung benötigter Daten sowie die Erstellung von 4D-Simulationen durch die Nutzung einer im Rahmen dieser Arbeit spezifizierten Verknüpfungssprache. Zusätzlich erfolgt eine Unterstützung bei der Identifikation von Änderungen in zugrunde gelegten Planungsdaten sowie deren Auswirkungen auf Basis einer Versionierung auf Objektebene.

3. Die *Generierung von zusätzlichem Nutzen* aus den erstellten Verknüpfungen zwischen Terminplanvorgängen und Objekten des Bauwerksmodells.

Letzteres bezieht sich insbesondere auf die Bereitstellung von Basisdaten für die Berechnung der Dauer von Vorgängen, hat jedoch seine Bedeutung auch über die eigentliche Terminplanung hinaus. Denn ist eine Datenverknüpfung mit dem Terminplan erst einmal erzeugt und somit eine zeitliche Reihenfolge in den Planungsdaten verankert, so ergeben sich aus der Nutzung dieser Information weitere Möglichkeiten, wie z.B.

die Optimierung des Materialtransports und des Lagermanagements („just in time"-Anlieferung), die Erstellung von Zahlungsplänen oder die automatische Fortschrittskontrolle auf der Baustelle mittels Kameraüberwachung ([Rebolj u. a. 2008]). Trotz ihrer Relevanz sind diese weiterführenden Themen jedoch nicht Gegenstand dieser Arbeit. Die Weiterverarbeitung von im Terminplanungsprozess erzeugten Informationen in nachgelagerten Prozessen, wie z.B. der Kostenkalkulation, ist ebenfalls nicht Gegenstand dieser Arbeit.

1.4 Vorgehensweise

Aufbau der Arbeit: Die vorliegende Arbeit ist in sieben Kapitel unterteilt, wobei die Kapitel 4 und 5 den Kern der Arbeit bilden.

Kapitel 2 – Prozessanalyse und Stand der Technik: Nachdem in der vorhergehenden Einleitung bereits die Motivation und Zielsetzung der Arbeit dargelegt wurden, erfolgt im nächsten Kapitel eine detailliertere Analyse der Ausgangssituation. Es werden die zur Erstellung eines Terminplans angewendeten Methoden, die zugrunde liegenden Daten sowie die Notwendigkeit deren Austauschs analysiert. Danach folgt eine Zusammenfassung des Standes der Technik sowie ein Marktüberblick. Der Fokus liegt dabei auf der modellbasierten Unterstützung des Terminplanungsprozesses. Zu den einzelnen Themen werden zudem bereits erste Schlussfolgerungen gezogen.

Kapitel 3 – Entwicklungsbedarf: Aus den analysierten Prozessanforderungen und den identifizierten Defiziten erfolgt in diesem Kapitel eine Ableitung des Entwicklungsbedarfs. Dieser wird thematisch gruppiert sowie priorisiert und stellt den Maßstab für den anschließenden Lösungsansatz dar.

Kapitel 4 – Lösungsansatz: Auf Basis einer Modellbildung werden in diesem Kapitel Konzepte zur Aufhebung der identifizierten Defizite entwickelt. Kern stellt die Entwicklung einer Sprache für die effiziente Erstellung von fachdisziplinsübergreifenden Verknüpfungen dar. Der Fokus liegt dabei auf den für die Terminplanung relevanten Daten. Der vorgestellte Ansatz ist jedoch allgemeingültig und auch auf andere Fachdisziplinen übertragbar.
Die Unterstützung einer kollaborativen Zusammenarbeit im Planungsprozess erfolgt zudem durch eine globale Systemarchitektur, bei der die Datenhaltung während der fachspezifischen Bearbeitung durch den Terminplaner in einem versionierten Workspace erfolgt. Das entwickelte Versionierungskonzept wird vorgestellt.
Für den Datenaustausch zwischen verschiedenen Partnern in einem Projekt wurde das Datenformat IFC gewählt. Am Ende dieses Kapitels wird daher ein Überblick über die gewählte Abbildung der auszutauschenden Daten auf die in IFC definierten Objekte gegeben.

Kapitel 5 – Umsetzung: Zur Verifizierung und Erprobung der entwickelten Ansätze wurden eine IFC-kompatible 4D-Software mit integriertem Sprachinterpreter für die spezifizierte Verknüpfungssprache sowie ein versionierter Workspace entwickelt. In diesem Kapitel werden die wesentlichen, im Rahmen dieser Arbeit entwickelten Funktionalitäten sowie Aspekte ihrer Implementierung vorgestellt. Die Integration von kom-

merzieller Terminplanungssoftware in den Datenaustausch wird beispielhaft anhand von MS Project gezeigt.

Kapitel 6 – Beispiele und Demonstration: Anhand von Beispielen wird in diesem Kapitel die Anwendbarkeit und Praxisrelevanz der entwickelten Konzepte demonstriert. Zusätzlich wird ein Überblick über ein weiterreichendes Demonstrationscenario gegeben, dessen Vorbereitung sowie die Entwicklung zusätzlicher Komponenten derzeit zusammen mit Partnern im Rahmen des EU-Forschungsprojektes InPro[3] erfolgt.

Kapitel 7 – Zusammenfassung und Ausblick: Die Arbeit schließt mit einer Zusammenfassung der gewonnenen Erkenntnisse und einem Fazit bzgl. der erreichten Integration der Terminplanung in einen kollaborativen, modellbasierten Planungsprozess. Ebenso wird ein Ausblick auf noch ausstehende Arbeiten und aktuelle Entwicklungen am Softwaremarkt gegeben.

[3] www.inpro-project.eu

2 Prozessanalyse und Stand der Technik

2.1 Prozessanalyse

Grundlage für den Entwurf von Verbesserungsmaßnahmen ist zunächst ein tieferes Verständnis des derzeitigen Terminplanungsprozesses, der darin angewandten Methoden und verarbeiteten Daten sowie der verwendeten Hilfsmittel. Daher wurden als Grundlage dieser Arbeit Interviews mit mehreren Fachleuten aus der Praxis der Terminplanung und Arbeitsvorbereitung geführt und aus den daraus gewonnenen Erkentnissen mit Hilfe der Business Process Modeling Notation (BPMN[1]) der Kernprozess der Terminplanerstellung modelliert ([Fijneman u. a. 2008]). Die Wesentlichen Ergebnisse aus dieser Prozess- und Methodenanalyse sind in den folgenden Unterkapiteln zusammengefasst. Dabei entspricht die dargestellte Sichtweise im wesentlichen der eines Generalunter- bzw. -übernehmers und betrachtet somit das volle Spektrum der Terminplanung. Am Ende jedes Unterkapitels werden bereits erste Schlussfolgerungen gezogen, die beim späteren Entwurf eines Zielszenarios zu berücksichtigen sind. Zusätzlich sei auf [Winch u. Kelsey 2005] hingewiesen, wo Resultate von in England durchgeführten Interviews veröffentlicht sind.

2.1.1 Zusammenarbeit in der Terminplanung

Im Rahmen größerer Projekte ist stets eine Vielzahl von unterschiedlichen Projektbeteiligten zur Lösung vielfältiger Teilaufgaben involviert. Deren zeitlich reibungslose Zusammenarbeit ist von der *projektorientierten Terminplanung* zu koordinieren, die auf einer übergeordneten Planungsebene eine ganzheitliche Betrachtung des Bauvorhabens und dessen Planung über mehrere Projektphasen hinweg verfolgt ([Mechnig 1997a]). Im Gegensatz hierzu wird mit der *produktionsorientierten Terminplanung* die zeitliche Planung einzelner feingegliederter Produktionsprozesse auf der operativen Planungsebene bezeichnet, die sich meist nur über einen Teil einer Projektphase erstreckt (z.B. Ausführung eines Gewerks[4]).
Die Gesamtheit der Terminplandaten besteht somit aus einem groben Gerüst, das vom Auftraggeber, dem Projektleiter oder dem Generalunter- bzw. -übernehmer zwecks Projektstrukturierung vorgegeben wird, sowie aus den einzelnen Detailabläufen der unterschiedlichen Projektphasen und Ausführungsgewerke (siehe Abbildung 2.1 auf der nächsten Seite). Letztere werden von den ausführenden Einzelunternehmen im Rahmen der ihnen zugeteilten Zeitfenster eigenverantwortlich erstellt. Diese Detailterminpläne

[4] siehe Glossar

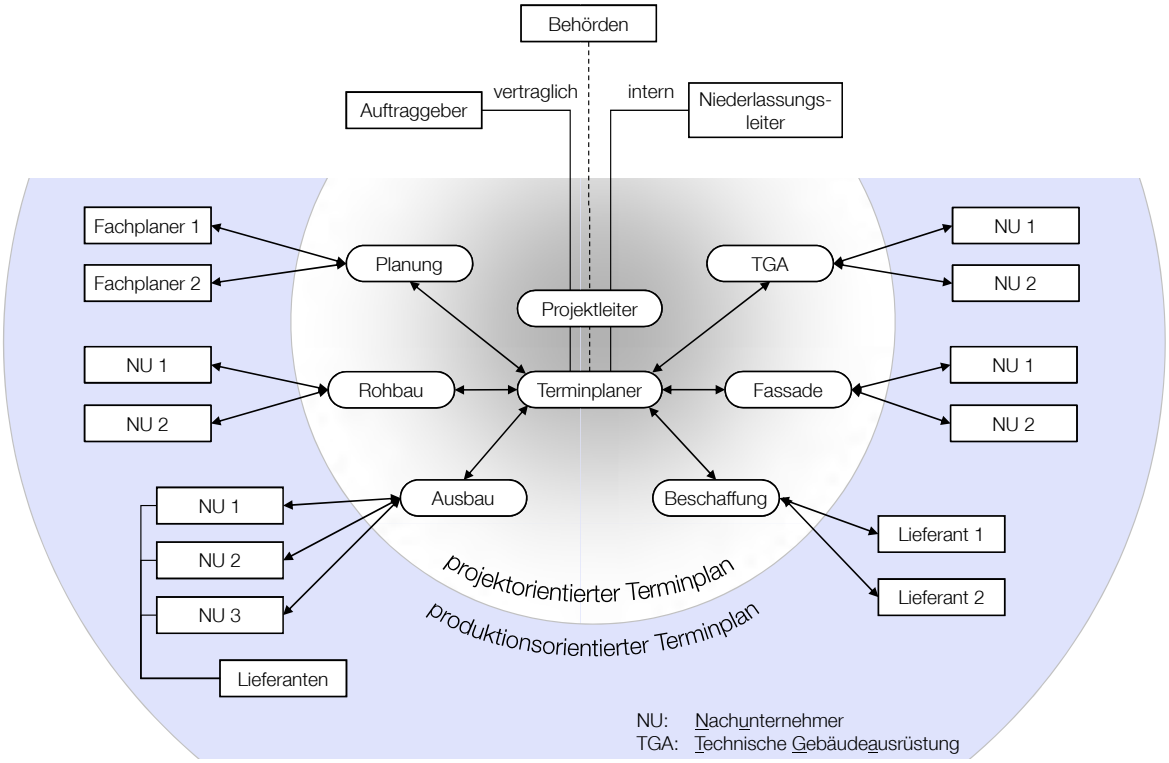


Abbildung 2.1: Austausch von Terminplanungsdaten zwecks zentraler Koordinierung eines Bauprojekts

werden der Projektleitung zur Kenntnisnahme und späteren Leistungskontrolle zur Verfügung gestellt. Mitunter kommt es auch zum Austauch von Terminplanungsdaten zwischen Einzelunternehmen. Dieses ist der Fall, wenn die betreffenden auszuführenden Tätigkeiten technologisch voneinander abhängen oder eine räumliche Trennung sicherzustellen ist.
Zur Zusammenführung und Koordinierung der einzelnen Teilterminpläne sowie zur Information aller Projektbeteiligten über zeitliche Abläufe ist im Projekt eine „Datenleitung“ aufzubauen, die eine Übermittlung von Terminplanungsdaten ermöglicht ([Mechnig 1997b]). Dabei hilft der schnelle EDV-gestützte Austausch von Terminplanungsdaten nicht nur Informationsverluste bei der Weitergabe von Terminen und Fristen zu vermeiden, sondern erhöht auch die Termintransparenz und gewährleistet so die Termintreue aller Projektbeteiligten. Gerade in der Phase der Projektrealisierung erscheint es wichtig, dass durch den nahezu unverzögerten Austausch von Terminplanungsdaten die Reaktionszeiten verkürzt werden können.
Im gleichen Zusammenhang wird in [Kochendörfer u. a. 2007] betont, dass für das Verständnis der terminlichen und technischen Abläufe eine geeignete Darstellungsform zu wählen ist, die den Projektbeteiligten alle Informationen über den Ablauf in einfachster Weise zur Verfügung stellt. Die Erfahrung zeigt demnach, dass die Akzeptanz von Terminplänen dramatisch fällt, wenn deren Inhalte ungefiltert an Projektbeteiligte ausgegeben werden. D.h., es sind stets dem Aufgabenfeld der Projektbeteiligten entsprechende Teilauszüge des Gesamtterminplans weiterzuleiten und darin enthaltene Einzelvorgänge ggf. gemäß ihrer Bedeutung farblich zu gestalten.

Schlussfolgerungen: Aus den geschilderten Aspekten ist ersichtlich, dass eine Unterstützung des Terminplanungsprozesses durch die Sicherstellung des ungehinderten Austauschs von Terminplandaten unter den Projektbeteiligten erfolgen kann. Als Ergänzung erscheint eine Filtermöglichkeit zur verwendungsgerechten Extraktion von Teilinformationen zwingend erforderlich. Zusätzlich besteht bei allen Projektbeteiligten ein Bedarf für eine einfach verständliche Visualisierung von terminlichen und technischen Abhängigkeiten. Hierbei sollte zum leichteren Verständnis eine Farbkodierung verwendet werden. Eine projektübergreifende feste Bedeutung je Farbe erscheint dabei sinnvoll und erleichtert das schnelle Verständnis.

2.1.2 Inhalt und Gliederung eines Terminplans

Wie im vorherigen Kapitel erläutert, ist das Ziel der projektorientierten Terminplanung die zentrale terminliche Koordination des Gesamtprojektes. Hieraus ergibt sich, dass alle wichtigen Phasen der Projektabwicklung im Terminplan berücksichtigt werden müssen. Im Einzelnen gehören hierzu (siehe auch Abbildung 2.1):

- die technische Planung des Bauwerks (inkl. externer Prüfung),
- die Ausschreibung und Vergabe von Nachunternehmerleistungen,
- die Beschaffung von Material und Maschinen,
- die Einrichtung der Baustelle,
- die Bauausführung selbst,
- die Bemusterung (Fassade, Ausbau usw.),
- die Bauabnahmen (Bauherr, Feuerwehr, TÜV, Nutzer) und Inbetriebnahme des Bauwerks.

Somit enthält der Gesamtterminplan eines Projektes einen relativ großen Anteil (20-50%) an Vorgängen, die keine unmittelbaren Bauprozesse darstellen, jedoch mit diesen eng verwoben und von gleichrangiger Bedeutung sind. Dabei ist die zeitliche Anordnung der einzelnen Projektphasen aus globaler Projektsicht nicht strikt sequenziell, sondern mit gewisser Überlappung zeitlich gestaffelt. So erfolgt z.B. in der Regel die detaillierte Ausführungsplanung baubegleitend mit einem Vorlauf bis zur Planfreigabe des ersten Bauabschnitts. Während der Bauausführung erfolgt dann die Fertigstellung der Planung für den nächsten Bauabschnitt. D.h., es existieren komplexe Abhängigkeiten zwischen den einzelnen Projektphasen, die erst auf Detailebene erfasst werden können. Zur Wahrung der Übersicht und des leichteren Verständnisses der Abhängigkeiten wird der Terminplan mit einer kapitelähnlichen Gliederungsstruktur versehen. Diese wird in der Regel durch eine entsprechend horizontale Einrückung der Vorgänge in der tabellarischen Ansicht verdeutlicht (Abbildung 2.2 auf der nächsten Seite). Übliche Gliederungskriterien sind unter anderen: Geschoss, Bauabschnitt, Gewerk sowie die oben aufgeführten Phasen. Um leicht einem Zweck entsprechende Teilauszüge aus dem Gesamtterminplan erstellen zu können, werden die Terminplanvorgänge zusätzlich mit Eigenschaften versehen, nach denen gefiltert werden kann. Diese Eigenschaften können

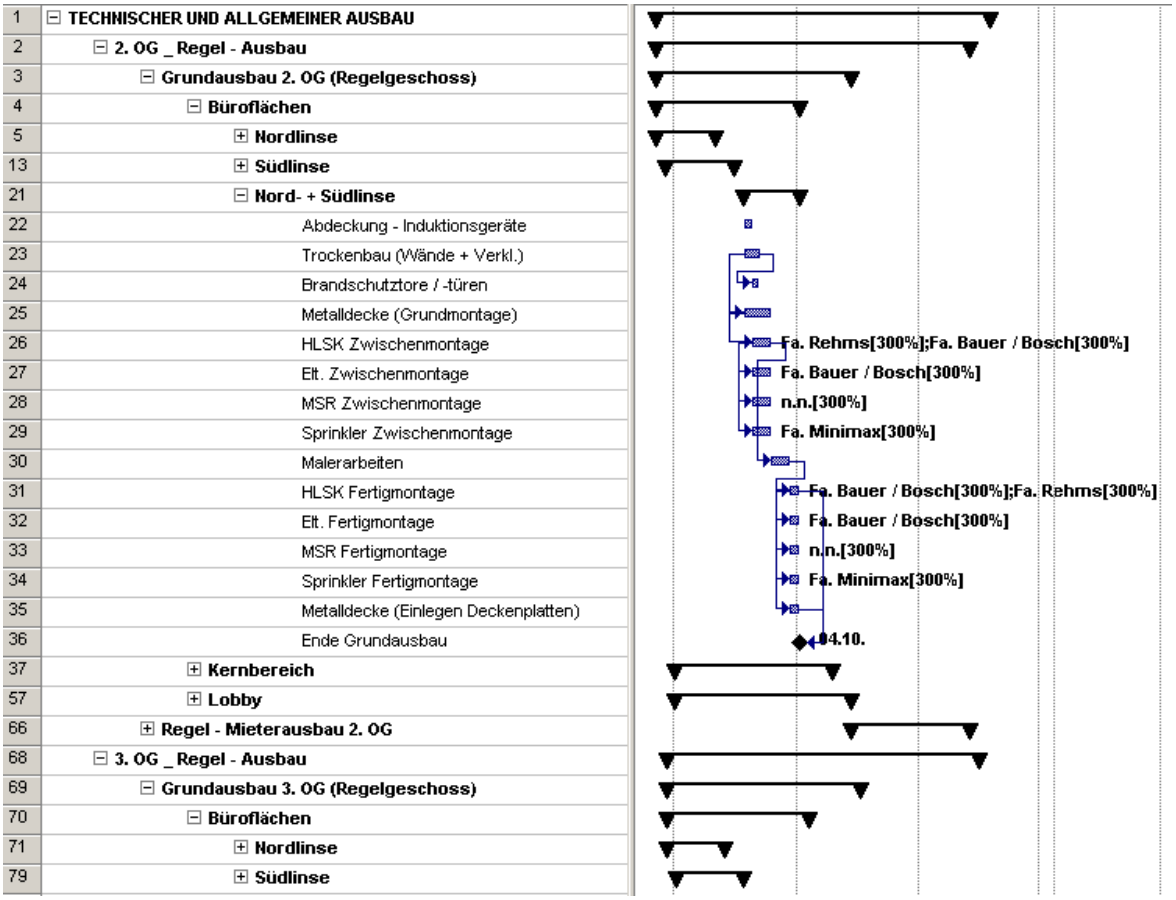


Abbildung 2.2: Typische Gliederungsstruktur eines Terminplans

eine Abbildung der Gliederungsstruktur sein oder darüber hinausgehende Informationen enthalten. Die Gliederungsstruktur und einzelnen Prozessketten eines Terminplans sind prinzipiell von der Art des Bauwerks und den Projektrandbedingungen abhängig. Bei ähnlichen Voraussetzungen wird jedoch oft die Terminplanstruktur eines alten Projektes als Grundlage für ein neues Projekt verwendet. Dieses spart Zeit und hilft sicherzustellen, dass alle wichtigen Vorgänge und Abhängigkeiten Berücksichtigung finden.

Schlussfolgerungen: Ein Gesamtterminplan enthält neben Vorgängen für die Bauwerkserstellung noch eine Vielzahl weiterer Vorgänge und Abhängigkeiten, deren Visualisierung unter Verwendung eines animierten, geometrischen Bauwerksmodells (4D-Simulation) nicht ohne Weiteres möglich ist. Somit hat eine Verknüpfung des Terminplans mit dem Bauwerksinformationsmodell nur selektiv für einen Teil aller Terminplanvorgänge zu erfolgen.
Zum Extrahieren bestimmter Vorgänge aus dem Gesamtterminplan hat sich eine Filterung auf Basis vorher gemäß der Gliederungsstruktur für jeden Vorgang vergebener Eigenschaften als nützlicher und leistungsfähiger Mechanismus etabliert. Dabei ist die verwendete Gliederungsstruktur prinzipiell vom Projekttyp abhängig, sollte aber von vorhergehenden Projekten übernommen werden können.
Beide genannten Anforderungen sind von der verwendeten Verknüpfungsmethodik zu unterstützen.

2.1.3 Methoden zur Bestimmung der Dauer von Vorgängen

Eine wesentliche Aufgabe bei der Terminplanerstellung ist die Ermittlung von angemessenen Dauer der Einzelvorgänge. Je nach aktuellem Reifegrad der Planung und den damit zur Verfügung stehenden Informationen werden hierzu verschiedene Methoden angewandt. Diese werden nachfolgend in der Reihenfolge vom detaillierten (A) zum groben (D) Planungsstand erläutert. A stellt dabei die bevorzugte Methode dar, da sie bereits auf gesicherten Planungsdaten basiert und somit die höchste Genauigkeit bietet. In frühen Planungsphasen muss jedoch aufgrund des groben Informationsstandes auf die anderen Methoden ausgewichen werden.

Methode A
Sie wird verwendet, wenn bereits eine detaillierte Planung von Bauteilen vorliegt. Es werden detaillierte Bauteilmengen ermittelt und diese mit Zeitbedarfsfaktoren für die betreffende Aufgabe multipliziert. Da im Vorfeld keine Aufschlüsselung existiert, in der Bauteilmengen den einzelnen Bauabschnitten zugeordnet sind (z.B. Betonmenge pro Betonierabschnitt einer Decke), müssen diese je Bauabschnitt manuell vom Terminplaner auf Basis von Plänen neu ermittelt werden.

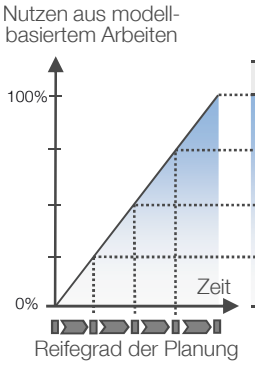
Methode B (Alternative zu C)
Diese Methode findet ihren Einsatz, wenn erst eine Entwurfsplanung vorliegt. Die Ermittlung der Vorgangsdauer erfolgt hierbei auf Basis von groben Bauwerksgeometriedaten (umbautes Raumvolumen, Geschossflächen, usw.) und zugehörigen, groben Zeitbedarfsfaktoren, die in Abhängigkeit zum Bauwerkstyp, dem Bauverfahren und dem technischen Schwierigkeitsgrad stehen.

Methode C (Alternative zu B)
Bei dieser Methode erfolgt die Ermittlung der Vorgangsdauer auf Basis der Kostenschätzung oder des Kostenbudgets über den gewerkspezifischen Lohnkostenanteil und die Anzahl der vorgesehenen Arbeitskräfte. Diese Methode wird im Wesentlichen verwendet, wenn die Planung für das betreffende Gewerk sich noch in einer sehr frühen Phase befindet und somit kaum Informationen zur Verfügung stehen. Die Aufschlüsselung des Kostenbudgets auf die Einzelgewerke wird hierbei von der Kalkulation zur Verfügung gestellt. D.h., bei dieser Methode besteht eine Umkehr der sonst üblichen Abhängigkeit zwischen Terminplanung und Kostenermittlung. Ein Risiko bei der Verwendung dieser Methode besteht darin, dass aufgrund der fehlenden Informationen technologische Erschwernisse und projektspezifische Prozessabhängigkeiten keine Berücksichtigung finden.

Methode D
Hierbei erfolgt die Festlegung der Vorgangsdauer lediglich durch Schätzung rein aus Erfahrung heraus. Dieses trifft insbesondere für Vorgänge zu, die nicht direkt die Bauwerkserstellung beschreiben (Planung, Beschaffung, Inbetriebnahme, usw.). Hierzu werden in der Regel Standardprozessketten verwendet, die lediglich gemäß den organisatorischen Rahmenbedingungen an das aktuelle Projekt anzupassen sind. Ggf. werden Zusatzinformationen, wie etwa die Anzahl der während der Planung zu erstellenden Pläne, mit in die Zeitschätzung einbezogen.

Die genannten Methoden werden auch in [Rösch u. Volkmann 1994] und teilweise in [Bielfeld 2009] beschrieben. Für die Planung der Bauausführung werden in der Regel die Methoden A-C verwendet. Die Entscheidung für eine Methode fällt dabei je Vorgang und in Abhängigkeit zu den verfügbaren Planungsdaten sowie des betrachteten Gewerks. Es ist üblich, dass innerhalb eines Terminplans alle genannten Methoden zur Anwendung kommen. Die Methoden B und C kommen dabei bei grobem Informationsstand alternativ zum Einsatz und können zur gegenseitigen Kontrolle verwendet werden. Einige Gewerke wie etwa der Rohbau werden bei grobem Informationsstand bevorzugt nach Methode B berechnet, andere wie Innenausbau- oder Fassadenarbeiten nach Methode C. Dieses ist darin begründet, dass eine für die Terminplanung relevante höherwertigere und damit technisch aufwändigere Bauweise im Innenausbau sich stets auch in den Kosten widerspiegelt, nicht jedoch in der Bauwerksgeometrie.
Die einzelnen zur Berechnung der Dauer von Vorgängen verwendeten Formeln und Faktoren stellen ein gewisses Know-how der jeweiligen Firma dar und werden Außenstehenden bzw. anderen Projektbeteiligten nicht zur Verfügung gestellt. In der Regel exisitieren in den Baufirmen hierfür umfangreiche Datensammlungen. Einen Einblick in die Struktur solcher Datensammlungen erhält man in [Bauwirtschaft 1982] und [Bernsdorff-Diehl-Klein 1998]. Hierin wird eine Zusammenstellung von Zeitbedarfswerten zur projektorientierten Terminplanung durch Auftraggeber bzw. Projektsteuerer angegeben. Es ist anzumerken, dass für Aufwandswerte in Tabellensammlungen stets Bereiche und nicht exakte Werte angegeben sind. Der verwendete Wert wird vom Terminplaner individuell je nach Projektrahmenbedingungen ausgewählt und ggf. mit zusätzlichen Erhöhungs- bzw. Abminderungsfaktoren versehen. Dabei wird stets versucht, die betrachteten Einflussfaktoren möglichst getrennt abzubilden, um sie jederzeit nachjustieren zu können. Diese Flexibilität zum Anpassen an Projektbedingungen wird von in der Praxis tätigen Personen als zwingend erforderlich empfunden. Eine ähnliche Vorgehensweise zur Berücksichtigung von Einzelfaktoren bei der Berechnung von Aufwandswerten wird in [Akbas 2003] verfolgt.

Schlussfolgerungen: Im Rahmen der Ermittlung der Dauer von Vorgängen werden verschiedene Eingangsinformationen aus der Planung und Kalkulation benötigt. Dieser Informationsbedarf kann potentiell aus einem gemeinsam genutzten Bauwerksinformationsmodell gedeckt werden. Dabei ergibt sich mit steigendem Detaillierungsgrad der benötigten Informationen und infolge sichergestellter Datenkonsistenz ein erhöhter Nutzen aus dem gemeinsamen, modellbasierten Arbeiten (Abbildung 2.3). Werden die benötigten Eingangsinformationen nicht oder in ungeeigneter Form von anderen Projektbeteiligten bereitgestellt, so erfolgt eine Neuermittlung (Mengenermittlung) bzw. es werden Annahmen getroffen. Dieses bedeutet einen erhöhten Aufwand, kann zu Dateninkonsistenz führen und sollte daher vermieden werden.
Aus gleichem Grund sind unternehmensweit einheitliche Musterprozessketten, Berechnungsformeln und Aufwandswerte zu verwenden. Um eine Pflege dieses Firmenfachwissens zu ermöglichen, ist eine zentrale Verwaltung eines solchen Kataloges erforderlich. Eine individuelle Anpassung der Vorlagen und Tabellenwerte an die speziellen Projektgegebenheiten muss jedoch stets möglich sein.



Methode	basierend auf	Ausgangsinformationen
A	Detailmengen	Bauteilmengen, detaillierte Aufwandswerte
B	Gebäudegeometrie	Gebäude- bzw. Raumkubaturen und -flächen, grobe Aufwandswerte
C	Kostenbudget	Kosten je Gewerk, Faktor für Personalanteil
D	Erfahrung	Projektbeschreibung, Visualisierungen

Abbildung 2.3: Methoden zur Bestimmung der Vorgangsdauer, benötigte Ausgangsinformationen und zugehöriger möglicher Nutzen aus modellbasiertem Arbeiten

2.1.4 Detaillierungsstufen eines Terminplans

Wie bereits in Kapitel 2.1.1 erläutert, wird der Gesamtterminplan aus dem projektorientierten Terminplan und dessen Verfeinerungen, den produktionsorientierten Terminplänen verschiedenster Projektbeteiligter, zusammengesetzt. Im Laufe eines Projektes durchläuft der Terminplan dabei parallel zum Reifegrad der technischen Planung verschiedene Detailstufen, die durch unterschiedliche Projektbeteiligte erstellt werden. Es können in etwa die in Tabelle 2.1 auf der nächsten Seite aufgeführten sieben Detailstufen unterschieden werden. Erwähnenswert ist, dass in einem Gesamtterminplan je nach verfügbaren Informationen zu einem Zeitpunkt für verschiedene Teilprozesse unterschiedliche Detailstufen vorliegen können. Beim Übergang zu einem feineren Detaillierungsgrad wird der vorherige Terminplan als Rahmen bzw. als Grundlage für die gegenseitige Kontrolle verwendet. D.h. er verbleibt als Summenvorgang im Terminplan erhalten und dient als Schnittstelle bei der Detaillierung durch andere Projektpartner. Diese benötigen für die Detaillierung, neben dem ihnen übermittelten Rahmenterminplan, aufgrund der Verwendung der gleichen Terminplanungsmethode (Methode A) ebenfalls die durch die vorhergehende Terminplanung bereits auf die vorgesehenen Bauabschnitte aufgeteilten Mengeninformationen.
Neben einer fortlaufenden Detaillierung des Terminplans infolge wachsenden Reifegrades der Planung muss auch auf Planungsänderungen reagiert werden. Hierbei besteht die Herausforderung zunächst darin, die neuen Daten an alle an der Terminplanerstellung Beteiligten zu kommunizieren und darin enthaltene Änderungen zu identifizieren. Letzteres erfolgt heutzutage meistens manuell und erfordert hohen Aufwand. Erst dann kann eine Überprüfung auf etwaige Einflüsse auf den Terminplan und ggf. dessen Anpassung erfolgen.
Daraus ergibt sich, dass alle an der Terminplanung Beteiligten bzw. Interessierten stets zeitnah mit aktuellen Informationen zu versorgen bzw. über Planungs- und Terminänderungen zu informieren sind. Dabei sind jedoch private Daten, wie etwa Berechnungsformeln und Zeitbedarfswerte, vor dem Zugriff durch andere Projektbeteiligte zu schützen. Lediglich (im Projekt) öffentliche Daten, wie etwa die als Berechnungsergebnis erhaltene Dauer von Vorgängen, ist zur Verfügung zu stellen.
Prinzipiell kann gesagt werden, dass es wünschenswert ist, im Terminplan bereits vor

Stufe	Ersteller	Verwendung	Methode
1	Terminplaner	Erster grober Phasenterminplan auf einem A3 Plan, ca. 25-30 Vorgänge für die Ausführung und ca. 5 Vorgänge für die Planung der Planung	B/C und D
2	Terminplaner	Vertragsterminplan, Detailstufe auf Geschossebene (z.B ein Vorgang für de Erstellung aller Stützen des 2.OG)	A und B/C
3	Terminplaner	Ausführungsterminplan	A
4-5	Bauleiter	Definition der Vorgaben für Nachunternehmer	A
6-7	Polier	eigenverantwortlicher Wochenterminplan	A

Tabelle 2.1: Detailstufen der Terminplanung

Baubeginn eine möglichst hohe Detaillierungstiefe zu erreichen. Allerdings bedingt dies ein erhöhtes Mitwirken des Auftraggebers, der oftmals in frühen Phasen bewusst keine endgültigen Festlegungen vornehmen möchte. Damit ist z.B. eine hohe Detaillierung der Terminplanung für den Innenausbau zu frühen Zeitpunkten oftmals nicht sinnvoll, da die Bemusterung und Festlegung der Raumausstattung erst später erfolgt.

Schlussfolgerungen: Neben den Terminplandaten sind insbesondere auch die auf Bauabschnitte aufgeteilten Mengendaten anderen Projektbeteiligten zur Verfügung zu stellen. Zudem hat zur zielgenauen Reaktion auf Änderungen eine Kommunikation bzw. Identifikation dieser Änderungen im Planungsdatenbestand auf Detailebene zu erfolgen. Bei der Bereitstellung von Planungsdaten ist zusätzlich stets eine Trennung zwischen privaten und (innerhalb des Projekts) öffentlichen Daten zu beachten.

2.1.5 Festlegung von Bauverfahren und Ressourcen

Das zur Ausführung kommende Bauverfahren beeinflusst maßgeblich die Bauzeit sowie die einzusetzenden Ressourcen (Maschinen und Personal). Andererseits ist die Wahl des Bauverfahrens von weiteren Kriterien abhängig. Diese sind in der Reihenfolge ihrer Priorität:

1. die technische Anforderungen des Bauwerks (z.B. verformungsarmer Verbau),
2. die Projekt- und Vertragsrahmenbedingungen (z.B. Platzverhältnisse, erlaubte Bauverfahren),
3. die entstehenden Kosten,
4. die resultierende Bauzeit.

Daraus ergibt sich, dass das verwendete Bauverfahren in der Regel nicht von der Terminplanung, sondern der technischen Planung bestimmt wird. Nur wenn mehrere Bauverfahren unter Betrachtung der vorrangigen Kriterien gleichwertig sind, wird das verwendete Verfahren auf Basis der benötigten Bauzeit festgelegt (z.B. Entscheidung zwischen

Fertigteil- oder Ortbetonbauweise). Hierbei ist stets die Wechselwirkung zwischen Bauzeit und Kosten zu beachten. Die eingesetzten Ressourcen werden hingegen stets durch die Terminplanung festgelegt. Dieses geschieht meistens auf Seite der ausführenden Firmen nach Vorgabe der Gesamtleistung (Gesamtaufwand in Arbeitsstunden) und des zugehörigen Zeitfensters durch die Projektleitung bzw. den Generalunternehmer.

Schlussfolgerungen: Da die Festlegung des zur Ausführung kommenden Bauverfahrens in der Regel außerhalb des Verantwortungsbereichs des Terminplaners erfolgt, ist dieses als Eingangsinformation zu kommunizieren bzw. die Wahl innerhalb der Terminplanung freizugeben bzw. anzufragen.
Gegenüber den ausführenden Firmen (Nachunternehmern) sind neben den Terminplanrahmen- und Modelldaten auch die sich aus den zu erbringenden Arbeiten ergebenden Leistungsanforderungen zu kommunizieren. Diese sind Grundlage für die Ressourcenwahl.

2.1.6 4D-Simulation

Mit der aufkommenden Verfügbarkeit von dreidimensionalen Modellen aus der Architekturplanung wurde der Terminplanungsprozess durch die nachgelagerte Erstellung von 4D-Simulationen ergänzt. Hierbei wird das existierende dreidimensionale Geometriemodell mit dem fertiggestellten Terminplan verknüpft. Nach der zusätzlichen Definition von Visualisierungsparametern (z.B. Farbe) kann die Bauwerkserstellung schrittweise visualisiert werden. Dabei werden Bauprozesse durch das Ein- und Ausblenden bzw. Einfärben von Bauteilen sichtbar gemacht. Für die Visualisierung von Aus- oder Umbaumaßnahmen werden zusätzlich die Arbeiten umschließende Raumobjekte verwendet. Der wesentliche Nutzen dieses Vorgehens besteht in der erleichterten, intuitiven Vermittlung von ansonsten auf Basis von Balkenplänen für Baulaien, aber auch für Fachleute schwer verständlichen Bauabläufen. Zudem hilft diese Technik, räumliche Abhängigkeiten im Bauprozess besser zu erkennen und bietet Baufirmen die Möglichkeit, ihr technisches Prozess-Know-how marketingwirksam einzusetzen. Die 4D-Simulation wird jedoch erst nach Fertigstellung des Terminplans zur nachträglichen Kontrolle bzw. zur Kommunikation der terminlichen Abläufe eingesetzt. Aus einem Bauwerksmodell wird hierzu lediglich die geometrische Darstellung genutzt. Eine weitergehende Nutzung von im Modell enthaltenen Informationen speziell zur Unterstützung der Terminplanerstellung erfolgt jedoch bisher nicht. Der in der Praxis noch zögerliche Einsatz von 4D-Simulationen weist auf ein oftmals zu geringes Nutzen-Aufwand-Verhältnis einer solch separaten Modellnutzung hin.
Die Ausbildung dieser Separierung von Terminplanerstellung und Modellnutzung hängt jedoch nicht zuletzt damit zusammen, dass für die Terminplanung im Bauwesen i.A. keine branchenspezifische Software, sondern allgemeingültige Projektmanagementsoftware verwendet wird. Diese kann aufgrund der durch die Hersteller angestrebten Anwendbarkeit auf jegliche Art von Projekten keine spezifische Unterstützung für die modellbasierte Terminplanung im Bauwesen und die damit erforderliche Verarbeitung von Bauwerksinformationsmodellen bieten. Andererseits sind diese Softwarepakete mit der

Zeit zu sehr mächtigen Anwendungen gereift und ermöglichen in der Regel sogar eine unternehmensweite Projekt- und Ressourcenauswertung. Somit ist die Schwelle für die Akzeptanz komplett neuer, speziell auf die modellbasierte Terminplanung im Bauwesen zugeschnittener Softwarelösungen sehr hoch, was wahrscheinlich den Hauptgrund für die Konzentration neuer Lösungen auf den Bereich der nachgelagerten Modellnutzung in Form einer 4D-Simulation darstellt. In [Young u. a. 2008] werden zudem die bereits in der Vergangenheit in den Firmen getätigten hohen Investitionskosten in Projektmanagementsoftware als möglicher Grund für eine zögerliche Neuausrichtung genannt.
Betrachtet man, wie heutzutage üblich, den Einsatz von 4D-Simulationen isoliert von der Terminplanerstellung, so existieren mehrere Hindernisse für den alltäglichen Einsatz im Projektgeschäft:

- fehlender bzw. zu geringer Nutzen während der Terminplanerstellung,
- ein zu hoher Aufwand für die Verknüpfung von Terminplanvorgängen und Geometrieobjekten des Bauwerksmodells sowie dessen Aktualisierung bei Planungsänderungen,
- das Erfordernis einer feineren als sonst üblichen Granularität für Terminplan und Bauwerksmodell zur Erstellung aussagekräftiger 4D-Simulationen,
- unzureichende Visualisierungsmöglichkeiten,
- die Notwendigkeit des Beherrschens mehrerer komplexer Softwarepakete.

Die einzelnen Punkte werden nachfolgend näher erläutert.

Verknüpfungsaufwand

Als Grundvoraussetzung für eine 4D-Simulation müssen die Vorgänge des Terminplans mit Objekten des Bauwerksmodells verknüpft werden. Hierzu existieren derzeit folgende Ansätze:

1. Manuelle Verknüpfung von Terminplanvorgängen mit je einer Menge von Objekten des Bauwerksmodells. Die Auswahl der zugehörigen Objekte erfolgt dabei einzeln in einer tabellarischen Darstellung des Terminplans und einer dreidimensionalen oder baumartigen, alphanumerischen Ansicht des Bauwerksmodells. Eine Variante dieses Ansatzes stellt die vorangehende manuelle Zuweisung einer gleichlautenden Kennzeichnung zu Terminplanvorgängen und zugehörigen Bauteilobjekten in den jeweiligen Fachanwendungen dar. Die anschließende Verknüpfung kann in diesem Fall automatisch erfolgen.
2. Zuweisung von Terminen oder Bestimmung einer Erstellungsreihenfolge von Bauteilen in der CAD-Anwendung und anschließend automatische Erzeugung des Terminplans aus dem CAD-Programm.
3. Erzeugung einer gleichartigen Baumstruktur des Bauwerksmodells (produkt break down structure) und des Terminplans (work break down structure). Anschließend erfolgt eine Abbildung der beiden Bäume aufeinander.
4. Nutzung von Datenbankanfragen zur Selektion von Bauteilen auf Basis des Objekttyps und vergebener Attribute.

5. Interaktive Erstellung des Terminplans in einer 3D-Umgebung mit gleichzeitiger Spezifikation der Visualisierung für 4D.

Die erste Methode ist am weitesten verbreitet und ist die Standardmethode seit dem Erscheinen der ersten 4D-Softwarepakete. Allerdings entsteht bei dieser Methode ein sehr hoher Aufwand, da alle zu verknüpfenden Objekte manuell selektiert und zugeordnet werden müssen. Der Aufwand ist somit direkt proportional zu der Anzahl der zu verknüpfenden Terminplanvorgänge und der Anzahl der zugehörigen Objekte des Bauwerksmodells. In [Tsai u. a. 2008b] wurde die Dauer für die Erstellung einer 4D-Simulation mit ca. 5000 verknüpften CAD-Objekten analysiert. Von 192 ausgewerteten Stunden wurden 71% für die Lokalisierung der zu verknüpfenden Objekte im 3D-Modell verwendet. Der Rest betraf das eigentliche Erstellen der Verknüpfung. Eine automatische Verknüpfung auf Basis vorher in den Fachanwendungen (Terminplanungs- und CAD-Software) vergebener gleichlautender Kennzeichnungen stellt dabei keine Erleichterung dar, da hierdurch der Aufwand nicht verringert, sondern nur verlagert wird. Somit muss bereits bei der Bearbeitung im CAD-Programm die Kenntnis der späteren Verknüpfung vorliegen, oder es wird eine nachträgliche Anpassung erforderlich. Dieses erfordert eine enge Koordination zwischen CAD-Konstrukteur und Terminplaner oder die direkte Bedienung der CAD-Software durch den Terminplaner, was aufgrund der Komplexität solcher Programme unpraktikabel ist. Zudem ergeben sich bei der automatischen Verknüpfung auf Basis gleicher Bezeichner Probleme, wenn einige (nicht alle) der zu einem Vorgang zuzuordnenden Geometrieobjekte auch einem anderen Vorgang zugeordnet werden sollen. Hierzu müssen den Bauteilobjekten weitere Kennzeichner zugeordnet werden und dieses bei der automatischen Verknüpfung mit berücksichtigt werden. Bei Änderungen im Bauwerksmodell müssen die Verknüpfungen manuell überprüft und ggf. ergänzt bzw. entfernt werden, da die Logik der Verknüpfung, d.h. die Überlegung, welche Art von Objekten einem Vorgang zuzuordnen sind, nicht im Computer gespeichert wird.
Die zweite Methode wird z.B. von dem von Autodesk Consult entwickeleten MS Project Export[5] aus Revit verfolgt. Hierbei entfällt der Arbeitsschritt der manuellen Verknüpfung. Bei Erstellung des Terminplans aus dem CAD-System heraus werden gleichzeitig die Verknüpfungen erstellt. Problematisch an dieser Methode ist jedoch, dass für jedes existierende CAD-Objekt ein oder mehrere Vorgänge erzeugt werden. Dieses führt zu einer Vielzahl von Vorgängen, die, je nach benötigter Detailstufe des Terminplans, im Nachhinein zusammengefasst werden müssen. Zudem ist das Einhalten einer vorgegebenen oder erwünschten Terminplangliederung nur schwer zu erreichen. Nicht unmittelbar mit der Bauwerkserstellung zusammenhängende Vorgänge wie Planung, Abnahme usw. werden überhaupt nicht erzeugt. Daraus ergibt sich ein hoher Aufwand für die Umgestaltung und Ergänzung des automatisch erzeugten Terminplans. Im Bauwerksmodell erfolgte Planungsänderungen unter Beibehaltung einer vorher angepassten Gliederungsstruktur im Terminplan einzuarbeiten erweist sich zudem als problematisch. Gleiches gilt für den Übergang zwischen mehreren Detailstufen in der Terminplanung. Auch ist die Vergabe von Terminen oder einer Reihenfolge im CAD-Modell aus den gleichen wie oben genannten Gründen ungeeignet (Koordinationsaufwand bzw. CAD-Kenntnisse

[5] http://images.autodesk.com/adsk/files/bim_project_planning_feb07_1_.pdf

beim Terminplaner erforderlich).
Bei der dritten Methode, die in [Aalami u. a. 1998], [Dawood u. a. 2003], [Kang u. a. 2005] und [Dawood u. a. 2005]) verfolgt wird, wird eine weitgehende Automatisierung der Verknüpfungserstellung erreicht, jedoch müssen als Voraussetzung sowohl das Bauwerksmodell als auch der Terminplan nach einer vorgegebenen Gliederungsstruktur erstellt werden, um eine Abbildung aufeinander zu ermöglichen. Dieses stellt eine hinderliche Zwangsbedingung dar, die zum einen bei einer arbeitsteiligen Projektbearbeitung nur schwer sicherzustellen ist und andererseits keine flexible Anpassung an Projektanforderungen bzgl. Modell- und Terminplanstrukturierung ermöglicht.
Die vierte Methode bewirkt, insbesondere bei einer großen Anzahl von Objekten im Bauwerksmodell, eine deutliche Reduzierung des Aufwands für die Selektion von einem Vorgang zuzuordnenden Bauteilen. Durch die Angabe einer Reihe von Filterkriterien bzgl. Objekttyp und Objekteigenschaften wie z.B. Materialbezeichnung oder Geschosszügehörigkeit werden die zu verknüpfenden Bauteilobjekte ausgewählt und anschließend dem betrachteten Terminplanvorgang zugeordnet. Dieses Verfahren wird z.B. von Autodesk Navisworks Jetstream durch die sogenannte „Elementsuche“ unterstützt. Das Modul „DataLink Tools“[6] ermöglicht zudem das Anbinden zusätzlicher Objektinformationen über eine Datenbank. Eine Speicherung der Filterregeln als sog. „Suchgruppe“ erleichtert dabei eine Aktualisierung der Verknüpfungen bei Änderungen im Bauwerksmodell. Die Filterregeln aller definierten Suchgruppen können auch als XML-Datei exportiert werden und somit in anderen Projekten wiederverwendet werden. Problematisch bei dieser Methode erweist sich, dass eine derartige Filterung ohne Berücksichtigung der räumlichen Lage nicht genügend zielgenau ist. Selbst bei Abbildung der Geschossstruktur auf Attribute wird durch die Auswahlmöglichkeiten keine genügende Zielgenauigkeit erreicht. So ist es z.B. nicht möglich, alle Betonstützen auszuwählen, die sich in einem gewissen örtlich abgegrenztem Bereich eines Geschosses befinden. In diesem Fall muss wie bei der ersten Methode auf die Vergabe von zusätzlichen Attributen für die auszuwählenden Objekte zurückgegriffen werden. Sollen Bauteile wie in der heute üblichen Arbeitsweise durch ihre Lage innerhalb eines Achsrasters angesprochen werden (z.B. zwischen Achse A-H) und soll dieses flexibel möglich sein, um mehrere alternative Aufteilungen in Bauabschnitte vergleichen zu können, so ist der Weg über Zusatzattribute sehr aufwändig und unflexibel.
In [Zhou u. a. 2009] und [Waly u. Thabet 2002] wird von einem weiteren Ansatz berichtet (fünfte Methode), bei dem eine interaktive Definition von Terminplan und 4D-Simulation am virtuellen Bauwerksmodell erfolgt. Dabei wird das aus der Planung bereits existierende Bauwerksmodell vom Anwender interaktiv in einer 3D-Umgebung Bauteil für Bauteil neu zusammengesetzt und damit virtuell gebaut. Bei jedem Schritt wird dem Terminplan ein neuer Vorgang hinzugefügt und die entsprechenden Informationen wie Start, Dauer, Vorgänger usw. eingegeben. Die Verknüpfung zwischen Terminplan und Bauwerksmodell entsteht so Schritt für Schritt nebenbei mit. Eine 4D-Simulation ist somit jederzeit möglich. Diese Methode bezweckt bereits die Integration der interaktiven Modellnutzung in den Prozess der Terminplanerstellung. Das hilft, Abhängigkeiten frühzeitig zu erkennen und somit Fehler im Terminplan zu vermeiden.

[6] www.profox.com/pdf/datalinktools_broschur.pdf

Der entstehende Aufwand für Erstellung und Aktualisierung dürfte jedoch dem der erstgenannten Methode entsprechen.
Zusätzlich zur Erstellung der Verknüpfung zwischen Bauwerksmodell und Terminplan müssen für eine 4D-Simulation pro Vorgang noch Visualisierungsparameter (z.B. Farbe, Transparenz) für die Darstellung aktiver Vorgänge angegeben werden. Dieses stellt einen Zusatzaufwand dar, der durch die Nutzung von Vorlagen (sog. Templates) je Vorgangstyp verringert werden kann.

Schlussfolgerungen: Der hohe Aufwand für die Verknüpfung von Terminplan und Bauwerksmodell wurde bereits als ein Haupthinderungsgrund für den täglichen Einsatz von 4D-Simulationen erkannt. Es haben sich daher verschiedene Lösungsansätze ausgebildet. Keiner davon erscheint optimal. Die wichtigsten durch einen Verknüpfungsmechanismus zu erfüllenden Kriterien sind:

- geringer Aufwand bei Erstellung und Aktualisierung der Verknüpfungen,
- flexible Verknüpfungsmöglichkeit ohne notwendige Modellvorbereitung oder -anpassungen in anderen Fachapplikationen und dadurch vollständig in den Prozess der Terminplanerstellung integrierbar,
- keine Zwangsbedingungen für die Gliederungsstruktur des Terminplans.

Von den existierenden Verknüpfungsansätzen erscheint die Nutzung von Filterregeln für die Auswahl betreffender Objekte am effektivsten. Gleichzeitig stellt sie keine Anforderungen an die Gliederungsstruktur des Terminplans. Allerdings ist die Zielgenauigkeit aufgrund fehlender räumlicher Filterkriterien noch ungenügend.
Die Spezifikation von Visualisierungsparametern stellt einen Zusatzaufwand dar, der minimiert werden sollte.

Detaillierungsbedarf

Zweck einer 4D-Simulation ist im Wesentlichen auch die visuelle Kontrolle des erstellten Terminplans dahingehend, dass keine wesentlichen räumlichen und bautechnischen Abhängigkeiten übersehen worden sind. Hierzu ist eine entsprechende Detaillierung der 4D-Simulation erforderlich. Bei zu groben Terminplänen erhält man eine wenig aussagekräftige Visualisierung des Bauablaufs, da in der Regel sehr viele Prozesse gleichzeitig ablaufen, die eine geringere räumliche Trennung aufweisen als durch die 4D-Simulation erfassbar. So entsteht der Effekt, dass bei lokaler Betrachtung voneinander abhängige Bauteile gleichzeitig als in der Erstellung befindlich visualisiert werden, obwohl dieses real nicht der Fall ist. Werden z.B., wie für einen groben Terminplan üblich, für die Erstellung des Rohbaus eines Hochbaugeschosses nur drei Vorgänge verwendet, die das Erstellen der beiden Geschossdecken sowie aller dazwischen befindlichen Stützen abbilden, so kann es sein, dass die drei Vorgänge im Terminplan nicht streng sequentiell, sondern mit einer gewissen zeitlichen Überlappung abgebildet werden. Dieses hängt damit zusammen, dass die Erstellung in der realen Ausführung in mehreren Betonierabschnitten erfolgt und die obere Decke im ersten Betonierabschnitt evtl. schon betoniert wird, bevor die untere Decke im letzten Betonierabschnitt ausgeschalt ist. Diese räumlich-

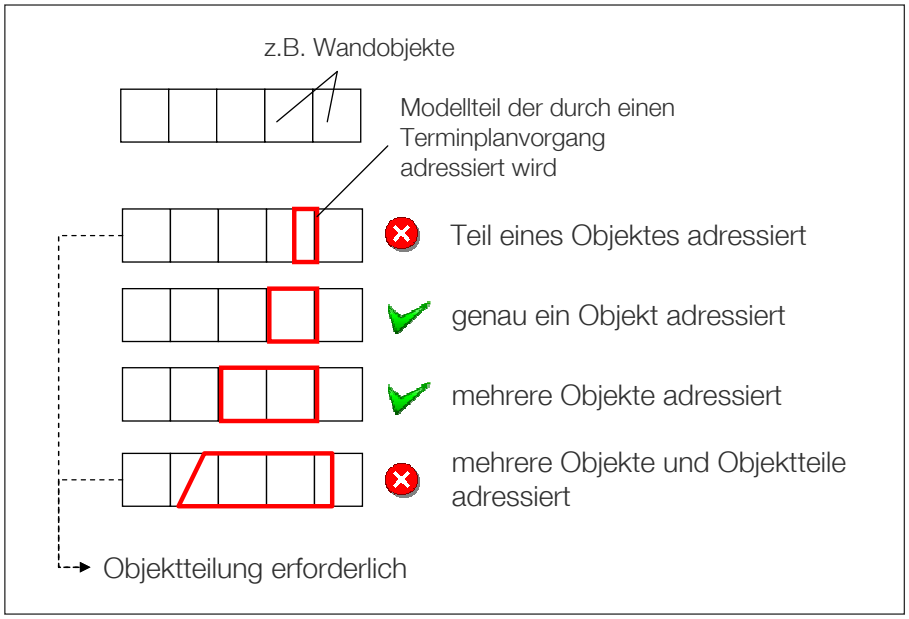


Abbildung 2.4: Notwendigkeit von Objektteilung je nach Ausgangsgranularität und Anfrageziel

zeitlichen Abhängigkeiten werden vom Terminplaner bei der Erstellung des groben Terminplans mental und durch die Vorgabe von Abstandszeiten zwischen dem Beginn zweier aufeinander folgender Vorgänge berücksichtigt (z.B. zwischen der Erstellung der unteren Geschossdecke und der Erstellung der Stützen). In einer 4D-Simulation, in der aktive Vorgänge schlicht durch Einfärben verknüpfter Bauteile visualisiert werden, führt dieses dazu, dass zu einem gewissen Zeitpunkt im ersten Betonierabschnitt sowohl die untere Geschossdecke als auch die darauf stehenden Stützen und die obere Geschossdecke als im Bau befindlich visualisiert werden. Dieses ist verwirrend und führt bei komplizierteren Fällen zur Identifikation von „Phantomfehlern“ in der Terminplanung. Um so etwas zu vermeiden und eine visuell plausible 4D-Simulation zu erhalten, muss der Terminplan in einer höheren als sonst üblichen Detaillierung erstellt werden, in der Betonierabschnitte explizit durch eigene Vorgänge abgebildet sind. Ähnliches wird auch in [Zamzow 2008, S.51] berichtet.
Dieses führt jedoch oft zu einem anderen Problem. Wird z.B. die Erstellung einer Geschossdecke im Terminplan in mehrere Bauabschnitte unterteilt, so werden für die Visualisierung der Bauabschnitte entsprechende Objekte im Bauwerksmodell benötigt. Diese sind aber in der Regel nicht vorhanden, da die Erstellung des Bauwerksmodells ohne Kenntnisse des Terminplans und nach Maßgabe einer möglichst effektiven Modellerstellung erfolgt ist. D.h., die Geschossdecke wird zuvor als ein Objekt oder eine Menge von Objekten in einer von den Bauabschnitten abweichenden Teilung modelliert. Diese Inkompatibilität der Objektgranularität des Bauwerksmodells mit der Detaillierung des Terminplans macht eine Anpassung des Bauwerksmodells erforderlich. Dieses ist immer dann erforderlich, wenn Teile eines vorhandenen Objektes mit einem Vorgang verknüpft werden sollen (Abbildung 2.4). Da heute verfügbare 4D-Simulationssoftware keine entsprechende Funktionalität bietet, muss in diesem Fall das betreffende Bauteilobjekt in der CAD-Anwendung entsprechend den Erfordernissen des Terminplans

in kleinere Objekte aufgeteilt werden. Das wiederum erfordert Koordinationsaufwand zwischen CAD-Konstrukteur und Terminplaner und stellt somit ein Hindernis im Terminplanungsprozess dar. Erfordert der Terminplan eine gröbere Objektgranularität als im Bauwerksmodell vorhanden, so kann dieses auf einfache Weise durch Gruppieren der vorhandenen Objekte erfolgen, ggf. müssen jedoch Teile von Bauteilobjekten in die Gruppierung mit einbezogen werden. Auch in diesem Fall ist somit eine Unterteilung von vorhandenen Bauteilobjekten erforderlich. Als besonders problematisch erweist sich die Anpassung der Objektgranularität des Bauwerksmodells, wenn mehrere Terminplanvarianten mit dem gleichen Modell verknüpft werden sollen. Dieses ist z.B. der Fall, wenn zur Erläuterung eines Bauzeitennachtrags verschiedene Terminpläne, die unterschiedliche Einteilungen in Bauabschnitte aufweisen, am gleichen Modell visualisiert werden sollen ([Robert 2008]). Im dort aufgeführten, konkreten Fall handelte es sich um den Vertragsterminplan, den infolge äußerer Einflüsse prognostizierten Terminablauf und den mit höherem Ressourceneinsatz realisierten Terminplan. In diesem Fall muss die Objektgranularität des Bauwerksmodells mit allen drei zu verknüpfenden Terminplänen kompatibel sein. Der Koordinierungs- und Anpassungsaufwand steigt hierdurch enorm an. Auch in [Koo u. Fischer 2003] wird die Notwendigkeit der Erstellung und Visualisierung mehrerer Terminplanvarianten betont.

Schlussfolgerungen: Um räumliche Abhängigkeiten plausibel visualisieren zu können, sollten Bauabschnitte explizit durch einzelne Vorgänge im Terminplan erfasst werden, anstatt implizit durch zeitlich versetzte, Bauabschnitte übergreifende Vorgänge. Die Angleichung des Detaillevels von Terminplan und Bauwerksmodell erfordert zudem häufig eine Teilung von Bauteilobjekten, was ein signifikantes Hindernis im Terminplanungsprozess darstellt.

Visualisierungsmöglichkeiten

Eine weitere Schwäche existierender 4D-Simulationssoftware sind die existierenden Visualisierungsmöglichkeiten. Üblich in allen existierenden Paketen sind die Visualisierung aktiver Vorgänge durch das Ändern der Darstellung von Bauteilobjekten zu den diskreten Zeitpunkten des Starts und Endes eines Vorgangs. Neben dem Ein- und Ausschalten von Objekten werden zudem die Farbe und Transparenz der Objekte geändert. Um dem Betrachter das Verständnis des visualisierten Bauablaufs zu erleichtern, sollte vom Ersteller der 4D-Simulation ein einheitliches, aussagekräftiges und gut erkennbares Farbschema verwendet werden. Nähere Untersuchungen hierzu wurden in [Chang 2009] vorgenommen. Die Standardisierung eines solchen Farbschemas würde sicherstellen, dass gleiche Vorgänge in verschiedenen Projekten auf gleiche Weise visualisiert werden und somit ein heute notwendiges, projektspezifisches Eindenken entfällt. Eine Klassifikation von Vorgängen unter anderem bzgl. ihrer Visualisierung wird in [Koo u. Fischer 2003] vorgenommen.
Doch in einigen Fällen sind die existierenden Visualisierungsmöglichkeiten nicht ausreichend. So ergibt sich z.B. bei der Visualisierung einer Gleit- oder Kletterschalungsbauweise das Problem, dass das kontinuierliche Wachsen des Bauwerks nur durch diskrete Bauabschnitte gezeigt werden kann, was zudem einen hohen Aufwand für Objektteilung

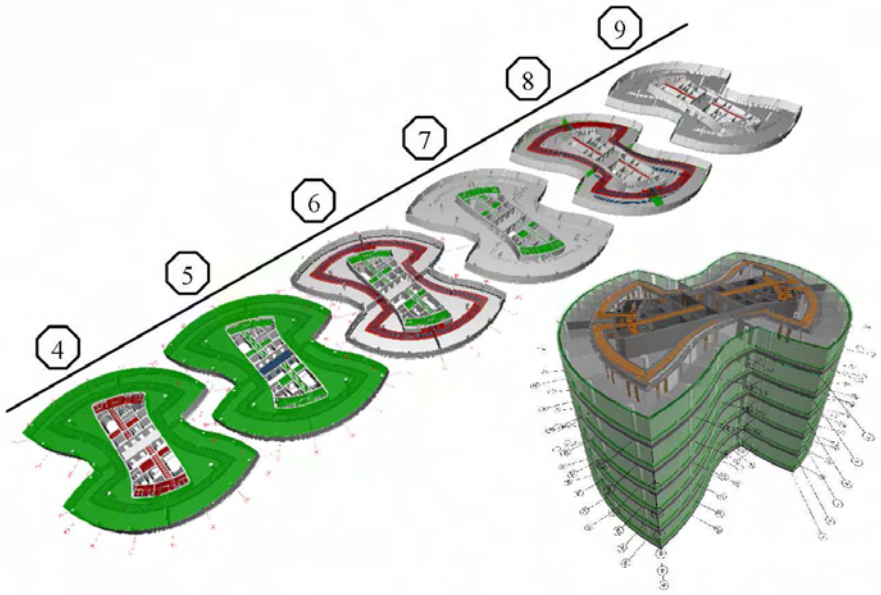


Abbildung 2.5: Explosionsansicht zur Beobachtung von Ausbauprozessen im Gebäudeinneren

und -verknüpfung bedeutet. Ein kontinuierliches Anwachsen durch die Skalierung von Bauteilobjekten oder das zeitabhängige Kappen durch dem betreffenden Bauteil zugeordneten Schnittebenen ist heutzutage nicht möglich. Das gleiche Konzept könnte auch dazu verwendet werden, Arbeitsrichtungen innerhalb von Bauabschnitten zu visualisieren. In [Akbas 2003] werden hierzu Vektorpfeile und Farbverläufe verwendet. Diese Möglichkeiten existieren jedoch in gängigen 4D-Softwarepaketen nicht. Auch das kontinuierliche Bewegen von Objekten wie z.B. Baugeräten ist heutzutage nicht möglich. Dieses kann nur über das An- und Ausschalten von mehreren Kopien des gleichen Objektes innerhalb des Modells realisiert werden. Insgesamt wäre somit für eine präzisere Visualisierung in einer 4D-Simulation die Möglichkeit zum kontinuierlichen Verändern der Objektdarstellung bzw. seiner Eigenschaften erforderlich.
Bei Baugeräten ergibt sich zudem das Problem, dass sie mit dem Bauwerksmodell modelliert und geladen werden müssen. Eine Bibliothek von Baugeräten, die zur Überprüfung von Platzverhältnissen und zur Darstellung der Anordnung der Baustelleneinrichtung einfach ausgewählt und hinzugeladen werden, wird bislang von den Softwareherstellern nicht ausgeliefert. Zudem muss der Terminplan zum An- und Abschalten der Visualisierung einzelner Baugeräte mit zusätzlichen „künstlichen“ Vorgängen versehen werden. Diese Steuerinformation sollte von der Terminplanung separat gehalten werden.
Ein schwerwiegendes Problem ist auch die Visualisierung von Ausbauprozessen, die innerhalb der Gebäudehülle verborgen sind und daher nach deren Fertigstellung in einer Außenansicht des Bauwerks nicht mehr beobachtet werden können. Hier fehlt es an einer Visualisierungsmöglichkeit, die es erlaubt, Ausbauprozesse im Verlauf durch das Gebäude hindurch verfolgen zu können. Dabei ist es erforderlich, gleichzeitig den Überblick über möglicherweise kollidierende Arbeitsprozesse im gleichen oder benachbarten Geschoss zu behalten. Hierzu wäre eine Explosionsansicht, die alle Geschosse treppenartig nebeneinander versetzt darstellt, hilfreich (Abbildung 2.5). Dieses wird

auch in [Russell u. a. 2009] berichtet. Geschosse, in denen gerade keine Bauaktivitäten stattfinden, könnten zur besseren Übersicht automatisch unsichtbar geschaltet werden. Eine solche Ansicht würde das zeitraubende manuelle Anpassen der Ansicht zur Verfolgung von Vorgängen im Gebäudeinnern überflüssig machen.
Zum Vergleich von verschiedenen Terminplanvarianten wäre auch eine zeitsynchrone Darstellung von mindestens zwei 4D-Simulationen erforderlich. Dabei sollte auch die Navigation synchronisiert erfolgen, damit das Modell inspiziert werden kann, ohne den direkten Vergleich der Bauablaufvarianten zu verlieren. Beides ist heutzutage in 4D-Programmen in der Regel nicht möglich. Wie schon erläutert, hat die Visualisierung verschiedener Terminpläne am gleichen Bauwerksmodell zudem einen erheblichen Einfluss auf die benötigte Objektgranularität und den erforderlichen Verknüpfungsaufwand.
Auch die Visualisierung von weiteren verknüpften Informationen ist sinnvoll. Hierzu gehören zeitlich synchronisierte, frei konfigurierbare Diagrammdarstellungen von Material-, Personal- und Maschineneinsatz, sowie die Entwicklung der Kosten über die Zeit aber auch die Signalisierung von verknüpften Dokumenten wie Ausführungsdetails oder ergänzende textuelle Beschreibungen im Modell. Auch eine im Modell eingeblendete, symbolische Zeitleiste mit grober Phaseneinteilung sowie markierten Meilensteinen aller momentan visualisierten Planungsalternativen würde zur Übersicht beitragen. Einige dieser Ansätze fanden bereits ihre Umsetzung. Ein Komplettpaket existiert jedoch bislang nicht.

Schlussfolgerungen: Bei den derzeitigen 4D-Simulationspaketen existiert eine Reihe von Defiziten in der Visualisierung, die eine effektive Erstellung und Nutzung von 4D-Simulationen verhindern bzw. ihren Nutzen vermindern. Insbesondere der zeitsynchrone Variantenvergleich von Bauabläufen und die Beobachtung von Ausbauvorgängen sind erforderliche Funktionalitäten.

Integration und Nutzen

Wie bereits eingangs erwähnt, findet die Erstellung einer 4D-Simulation üblicherweise nach Fertigstellung des Terminplans statt. Dieses hängt auch damit zusammen, dass hierfür eine spezielle Software verwendet wird, die nicht in übliche Terminplanungsprogramme integriert ist. Der Datenaustausch zwischen beiden Anwendungen ist nicht interaktiv, sondern erfolgt über den Umweg des (dateibasierten) Datenex- und -imports von der Terminplanungssoftware zum 4D-Simulationspaket. Ein Austausch in die andere Richtung ist in der Regel nicht möglich. So können notwendige Terminplanänderungen, die während der Validierung in der 4D-Umgebung direkt vorgenommen werden, nicht automatisch in die Terminplanungssoftware zurückgespielt werden. Werden die Änderungen in der Terminplanungsanwendung vorgenommen, so können diese nur über einen erneuten Datenex- und -import in die 4D-Simulation übernommen werden.
Aus Sicht des Terminplaners besteht die Anforderung, alle mit dem Terminplan zusammenhängenden Eingaben und Änderungen in der für diese Fachdisziplin weitaus mächtigeren Standardterminplanungssoftware vorzunehmen und eine zweite Anwendung oder ein Plug-in nur für die Interaktion mit dem Bauwerksmodell zu verwenden. Letzteres betrifft die Erstellung der Verknüpfungen mit den Bauteilobjekten, die Spezifikation

der weiteren Parameter sowie die Durchführung der 4D-Simulation. Darüber hinaus könnte die modellverarbeitende Anwendung den Terminplaner mit Daten versorgen, die er für die Berechnung der Dauer von Einzelvorgängen benötigt (siehe Kapitel 2.1.3 auf Seite 15). In Abhängigkeit vom aktuellen Detaillevel der Planung sind dieses die groben Geometrieinformationen, Kostenschätzungen/-budgets oder Detailmengen. Die benötigten Werte könnte der Terminplaner mittels Filter aus dem Modell extrahieren und zur weiteren Verarbeitung an die Terminplanungssoftware übergeben. Dort wird die Dauer von Vorgängen dann mittels Aufwandswerten und weiteren Berechnungsfaktoren ermittelt. Ideal wäre somit die parallele Nutzung beider Softwaretypen, ggf. auf zwei getrennten Computerbildschirmen nebeneinander, unterstützt durch einen interaktiven, bidirektionalen Datenaustausch. Hieraus entstehen einerseits ein Nutzen für die Terminplanung, durch Datenversorgung mit den ansonsten manuell ermittelten Ausgangsdaten und andererseits ein echter Mehrwert durch die Möglichkeit zur Terminplanerstellung begleitenden 4D-Simulation.

Schlussfolgerungen: Mangelnde Interaktionsmöglichkeit zwischen gängigen Terminplanungsanwendungen und 4D-Simulationssoftware sind ein Grund für den von der Terminplanerstellung separierten Einsatz der 4D-Technik. Somit beschränkt sich der heutige Nutzen auf die nachträgliche Validierung und Kommunikation des fertiggestellten Terminplans. Diesem steht ein beträchtlicher Erstellungsaufwand gegenüber. Eine wesentliche Verbesserung des Aufwand-Nutzen-Verhältnisses könnte durch eine interaktive, bidirektionale Schnittstelle und die *gegenseitige* Versorgung mit Planungsdaten erfolgen.

2.1.7 Hilfsmittel der Terminplanung

Während der Teminplanerstellung kommen, wie schon teilweise erwähnt, einige von den eigentlichen Projektdaten unabhängige Hilfsmittel zum Einsatz. Im Einzelnen sind dieses:

- Eine Datensammlung für Aufwandswerte. Diese ist strukturiert nach Gebäudetyp und -komplexität, Gewerken, Tätigkeiten und Kolonnenzusammenstellung.
- Ein Verzeichnis von Musterprozessketten für die Planung, Abnahme, Inbetriebnahme, usw.
- Checklisten zur Qualitätssicherung bzgl. Mindestanforderungen an Inhalt und Struktur des Terminplans.

Die genannten Hilfsmittel liegen heutzutage in Form von Dokumenten vor und werden als solche übermittelt. Zusätzlich werden Daten von vorhergegangenen Projekten als Grundlage bzw. zum Vergleich verwendet.

Schlussfolgerungen: Eine zentrale, datenbankbasierte Verwaltung von projektunabhängigen Datensammlungen würde die Pflege und Kommunikation erleichtern. Checks

zur Qualitätssicherung könnten, zumindest bzgl. der formalen Struktur, automatisch durch Software durchgeführt werden.

2.2 Marktüberblick

Für die Terminplanung im Bauwesen werden im Wesentlichen zwei Softwaretypen eingesetzt: Terminplanungs- bzw. Projektmanagementsoftware und 4D-Simulationssoftware. Während sich in der erstgenannten Anwendungsgruppe unter einer Vielzahl von Anbietern im Wesentlichen nur vier Software Programme am Markt etabliert haben, ist in der erst kürzlich mit dem Aufkommen des modellbasierten Arbeitens neu entstandenen zweiten Anwendungsgruppe noch keine endgültige Präferenz in der Bauindustrie zu erkennen. Einen Überblick über die wichtigsten Terminplanungsprogramme gibt Tabelle 2.2. Dabei sind zwei Terminplanungsmethoden zu unterscheiden: die übliche *Critical Path Method* (CPM)[7] unter Verwendung von Balkendiagrammen und die Methode namens *Line of Balance* (LoB)[7], bei der die Darstellung in verallgemeinerten Weg-Zeit-Diagrammen erfolgt. Zusätzlich zu den vier Marktführerprodukten wurde OpenProj in die Tabelle aufgenommen. Hierbei handelt es sich um eine nahezu vollständige Alternative zu MS Project, allerdings ist die Software als „open source“ verfügbar.

Firma	Internetseite	Produktname	Verfahren[7]
Asta	www.teamplan.co.uk	Teamplan	CPM
Microsoft	officemicrosoft.com	MS Project	CPM
Oracle	www.primavera.com	Primavera P6	CPM
VICO Software	www.vicosoftware.com	ViCo Control	LoB
Serena	openproj.org	OpenProj	CPM, open source

Tabelle 2.2: Überblick über die meist verbreiteten Terminplanungsprogramme

Tabelle 2.3 und 2.4 auf der nächsten Seite geben einen Überblick über kommerziell verfügbare 4D-Simulationsprogramme und Prototypen aus der Literatur. Die weiteste Verbreitung scheinen hiervon derzeit Autodesk Navisworks und Synchro Project Constructor erreicht zu haben. Aufgrund der sehr starken Dynamik im Softwareanbietermarkt kann dieser Überblick jedoch keinen Anspruch auf Vollständigkeit erheben. Auch wurde im Rahmen dieser Arbeit aufgrund des schnell wachsenden Umfangs der einzelnen Programmfunktionalitäten auf eine detaillierte Gegenüberstellung der einzelnen Softwarepakete verzichtet. Eine Beschreibung bzw. einen Vergleich einiger der genannten und weiterer Programme findet man in [Heesom u. Mahdjoubi 2004], [Leen-Seok u. a. 2004], [Sheppard 2004], [Porkka u. Kähkönen 2007] und [Eastman u. a. 2008]. Eine sehr umfangreiche Liste von Softwarepaketen insbesondere zur Terminplanung findet man auch unter www.planningengineers.org/software/. Hauptunterscheidungsmerkmale der 4D-Simulationssoftwarepakete sind die unterstützten CAD- und Terminplanungspro-

[7] siehe Glossar und [Uher 2003]

gramme, die Visualisierungsmöglichkeiten und die Existenz einer Datenbankanbindung.

Firma	Internetseite	Produktname
Autodesk	www.autodesk.com	Revit Architecture, Navisworks Simulate
Balfour Technologies	www.bal4.com	fourDscape
Bentley	www.bentley.com,	ProjectWise Navigator
Bentley	www.commonpointinc.com	ConstructSim
Building Explorer	www.buildingexplorer.com	Building Explorer
Ceco Interactive Design	www.ceco.se	Ceco Viewer
D-studio	www.d-studio.be	xD Virtual Builder
iD-ProPlan	www.idproplan.co.uk	4D-CCIR
Gehry Technologies	www.gehrytechnologies.com	Digital Project
Innovaya	www.innovaya.com	Visual Simulation
Intergraph	www.intergraph.com	SmartPlant Review
Synchro	www.synchroltd.com	Project Constructor
Tekla	www.tekla.com	Tekla Structures
VICO Software	www.vicosoftware.com	ViCo 5D Presenter

Tabelle 2.3: Marktüberblick 4D-Simulationssoftware

Organisation	Anwendungsname	Literaturquelle
University of Salford	web-based 4D/IFC data base	[Tanyer u. Aouad 2005]
University of Teeside	4D/VR information base	[Dawood u. a. 2003], [Dawood u. a. 2005]
VTT	Visual Product Chronology	[Kähkönen u. Leinonen 2003], [Porkka u. Kähkönen 2007]
University of Wolverhampton	4DX	[Zhou u. a. 2009]
Carnegie Mellon University	SiDaCoS	[Jan Reinhardt 2004]
University of British Columbia	REPCON	[Staub-French u. a. 2008]

Tabelle 2.4: Weitere 4D-Simulationssoftware in der Literatur

2.3 Weitere Literatur

Bezüglich der Verwendung von Computermodellen im Zusammenhang mit der Terminplanung existiert bereits eine Vielzahl von Veröffentlichungen. Sie beschränken sich jedoch meistens auf das Thema 4D-Simulation oder betrachten das Integrieren von CAD-, Kosten- und Terminplandaten als einen Prozess, der von einem einzelnen Akteur durchgeführt wird. D.h., ein kollaborativer Ansatz findet in der Literatur kaum Betrachtung. Nachfolgend sei jedoch noch auf einige im Zusammenhang mit dieser Arbeit stehende Literaturquellen hingewiesen.
In [Christiansson u. a. 2002] werden ein Überblick über Forschungsprojekte und Literatur im Bereich modellbasiertes Bauprozessmanagement gegeben sowie ein grober Systementwurf vorgestellt. In [Tanyer u. Aouad 2005] sind die Entwicklung der 4D-Simulationstechnik sowie deren Nutzen und existierende Probleme dargestellt. Es werden dabei der hohe Aufwand für die Erstellung einer 4D-Simulation sowie deren Anpassung und die oft inkompatible Objektgranularität von Terminplan und CAD-Modell als wesentliche Probleme identifiziert. Es werden diesbezüglich jedoch keine Lösungsansätze präsentiert. Zudem wird auf das Fehlen von IFC-kompatibler Software für die Terminplanung und Mengenermittlung hingewiesen. Darüber hinaus wird die Entwicklung und der Einsatz eines Softwarepakets beschrieben, das dem im Rahmen der vorliegenden Arbeit entwickeltem System ähnelt. Dabei wird eine einzelne Projektdatenbank verwendet und damit kein physische Trennung von privaten und öffentlichen Daten vorgenommen. Eine Versionierung wird ebenfalls nicht betrachtet.
Eine Reihe von Veröffentlichungen an der Stanford Universität befasst sich ebenfalls mit der Integration von Terminplan, Kosten- und CAD-Daten ([Staub-French u. a. 2002a], [Staub-French u. a. 2002b], [Staub-French 2004]). Jedoch steht dabei die Kostenermittlung im Vordergrund und nicht der Nutzen für die Terminplanung. In einem späteren Report ([Märki u. a. 2007]) wird hingegen die Nutzung von Mengen als Basis für die Berechnung der Dauer von Vorgängen erwähnt. Eine formale Beschreibung von Abhängigkeiten zwischen Ressourcen, Terminplan und Kosten findet man in [Chen 2008].
Einen weitergehenden Literaturüberblick zum Thema 4D-Simulation enthält [Zamzow 2008]. Ein Buch zu diesem Thema ist unter [Issa u. a. 2003] erschienen. Der Nutzen dieser Technik wird zudem in [Allen u. a. 2005], [Koche 2006] und [Goldstein 2001] betrachtet. Die bereits erläuterten Probleme des hohen Verknüpfungsaufwands, der unzureichenden Visualisierungsmöglichkeiten und die notwendigen Modellanpassungen (Objektteilung) werden auch in [Staub-French u. Fischer 2001] berichtet.
Zum Thema Projektmanagement im Allgemeinen sowie übliche Terminplanungsmethoden sind die Werke [GPM 2003], [Schexnayder u. Mayo 2004], [Schelle u. a. 2006] und [Kerzner 2009] zu empfehlen. Detailliertere Ausführungen zur Terminplanberechnung findet man in [Hofstadler 2007] und [Hinze 2004]. Einen Algorithmus für die Terminplanberechnung nach der CPM-Methode findet man in [Adeli u. Karim 2001] und unter www.codeproject.com/KB/recipes/CriticalPathMethod.aspx.

3 Entwicklungsbedarf

Aus den zuvor beschriebenen Anforderungen und existierenden Defiziten werden in diesem Kapitel Schwerpunkte gesetzt und daraus der im Weiteren verfolgte Entwicklungsbedarf abgeleitet sowie weiter detailliert.
Der Verwendung von Bauwerksinformationsmodellen für die Terminplanung wird Potential zur Verbesserung gegenüber der heutigen Arbeitsweise zugeschrieben. Allerdings weist die bisherige Form der Nutzung, meistens in Form einer nachgelagerten 4D-Simulation, noch sehr großen Arbeitsaufwand und vergleichsweise geringen Mehrwert für die Terminplaner selbst auf. Diesbezüglich werden zwei prinzipielle Möglichkeiten zur Verbesserung gesehen:

(a) Generierung zusätzlichen Nutzens,

(b) Verringerung des anfallenden Aufwands.

Die *Generierung zusätzlichen Nutzens* kann dadurch erreicht werden, dass ein Zugriff auf ein Bauwerksinformationsmodell, außer zur Verknüpfung mit den Geometrieobjekten zwecks Visualisierung, auch zur Extraktion der Eingangsinformationen für die Berechnung der Dauer von Vorgängen erfolgt. Die Bereitstellung dieser ohnehin benötigten und ansonsten manuell ermittelten bzw. übertragenen Informationen bildet den Anreiz für eine Modellnutzung schon während der Terminplanerstellung.
Die Einbeziehung von Kosten- und Mengeninformationen wurde bereits in der Literatur und auch durch Softwarehersteller verfolgt. Allerdings werden in der Regel weder der mit der Projektlaufzeit anwachsende Reifegrad der Planung und der sich damit ändernde Informationsstand noch die in mittleren und großen Bauprojekten übliche Entkopplung von Teilprozessen und ihre Ausführung durch unterschiedliche Akteure berücksichtigt. Auch die Fragmentierung der Bau- und Bausoftwareindustrie, die zu einer von Projekt zu Projekt wechselnden Zusammensetzung von Projektpartnern und einer damit verbundenen Nutzung heterogener Softwareanwendungen zur gemeinsamen Projektbearbeitung führt, wird kaum berücksichtigt. Statt dessen wird stets eine umfassende Datenbearbeitung durch einen Akteur oder innerhalb einer Anwendung bzw. Produktfamilie eines Herstellers betrachtet. Im Rahmen dieser Arbeit soll daher ein konzeptioneller Entwurf für die Zusammenarbeit auf Basis eines Bauwerksinformationsmodells unter Berücksichtigung der genannten Rahmenbedingungen entwickelt werden.
Zur *Verringerung des anfallenden Aufwands* für die Datenextraktion und die Verknüpfung mit Objekten des Bauwerksinformationsmodells ist ein leistungsfähiger Mechanismus zu entwerfen, der die benötigten Informationen verschiedener Fachdisziplinen zielgenau ansprechen kann. Dabei ist die oft benötigte Anpassung der Objektgranularität der Informationen im Bauwerksmodell weitestgehend zu automatisieren. Zur Erleichterung der Reaktion auf Planungsänderungen und zur Verwaltung mehrerer Planungsva-

rianten und -historien ist zudem ein Konzept zur lokalen, selektiven Versionierung von Planungsdaten zu entwickeln.

3.1 Modellbasierte Zusammenarbeit im Projekt

Die Planung eines Bauprojektes verläuft in mehreren Iterationszyklen mit anwachsendem Detail- und Reifegrad nach Kundenvorgaben und -anforderungen. Während der Terminplanung und der Erstellung von 4D-Simulationen werden dabei, wie in den Kapiteln 2.1.3 und 2.1.6 bereits erläutert, verschiedene Ausgangsinformationen benötigt. In der bisherigen Arbeitsweise werden geometrische und alphanumerische Daten aus Dokumenten (zweidimensionale Pläne, Tabellen usw.) manuell gewonnen, interpretiert und in die Terminplanungssoftware übertragen. In der modellbasierten Arbeitsweise können diese Informationen im Laufe des Projektes durch unterschiedliche Akteure nachfolgenden Prozessen im Modell zur Verfügung gestellt werden. Der Gesamtplanungsprozess ist daher so zu gestalten, dass benötigte Informationen im Rahmen der Zusammenarbeit zur richtigen Zeit verfügbar sind. Eine inhaltliche Entkopplung und klare Definition von Schnittstellen zwischen den einzelnen Teilprozessen stellt dabei sicher, dass ein den Bearbeitungsfluss störender Koordinierungsaufwand minimiert wird. Die *Wiederverwendung von Daten* stellt ein Grundprinzip der modellbasierten Zusammenarbeit dar. Dabei stellen Mengen neben der Bauwerksgeometrie (grobe Geometriedaten, sog. Basismengen sowie detaillierte Mengen aus dem Leistungsverzeichnis[8] (LV)) einen zentralen Teil des Informationsbestandes dar, der neben der Terminplanung auch in weiteren Teilprozessen wie z.B. der Kalkulation, Beschaffung und Abrechnung benötigt wird. Nach der bisherigen Arbeitsweise werden die Mengen in jedem Teilprozess separat manuell ermittelt. Dieses stellt einen vermeidbaren Mehraufwand dar und kann zu Dateninkonsistenz zwischen den einzelnen Teilprozessen führen. Zudem wird bei der manuellen Ermittlung der örtliche und direkte Bezug zu den Bauteilen nicht im Modell hinterlegt. Dieser ist jedoch gerade für die Terminplanung von Interesse, die die Mengen gemäß der Einteilung des Bauwerks in Bauabschnitte benötigt. Ein wesentlicher Schritt in der Gestaltung des neuen Prozesses der Zusammenarbeit ist somit die Einführung einer zentralen, modellbasierten Mengenermittlung sowie die Bereitstellung der dabei erzeugten Mengeninformationen für nachfolgende Prozesse. Der räumliche Bezug von Mengen ist dabei zu erhalten und dient einer zielgenauen Extraktion von Teilmengen für die Weiterverarbeitung in der Terminplanung. Abbildung 3.1 auf der nächsten Seite zeigt das Zielszenario. Die Wiederverwendung von Daten der Kostenschätzung/-kalkulation für die Terminplanung ist auf ähnliche Weise sicherzustellen. Im Lösungsansatz ist neben einer Beschreibung der für die Terminplanung relevanten Akteure und der von ihnen erzeugten oder genutzten Daten auch eine geeignete informationstechnische Infrastruktur zu entwerfen, die die aufgezeigte Zusammenarbeit zwischen Projektbeteiligten ermöglicht.
Auch in [Tsai u. a. 2008a] wird unter Betrachtung der einzelnen Akteure ein Prozess zur Nutzung von 4D-Simulationen in Baufirmen vorgestellt. Eine Nutzung von Mengeninformationen in der hier vorgestellten Weise findet jedoch keine Berücksichtigung.

[8] siehe Glossar

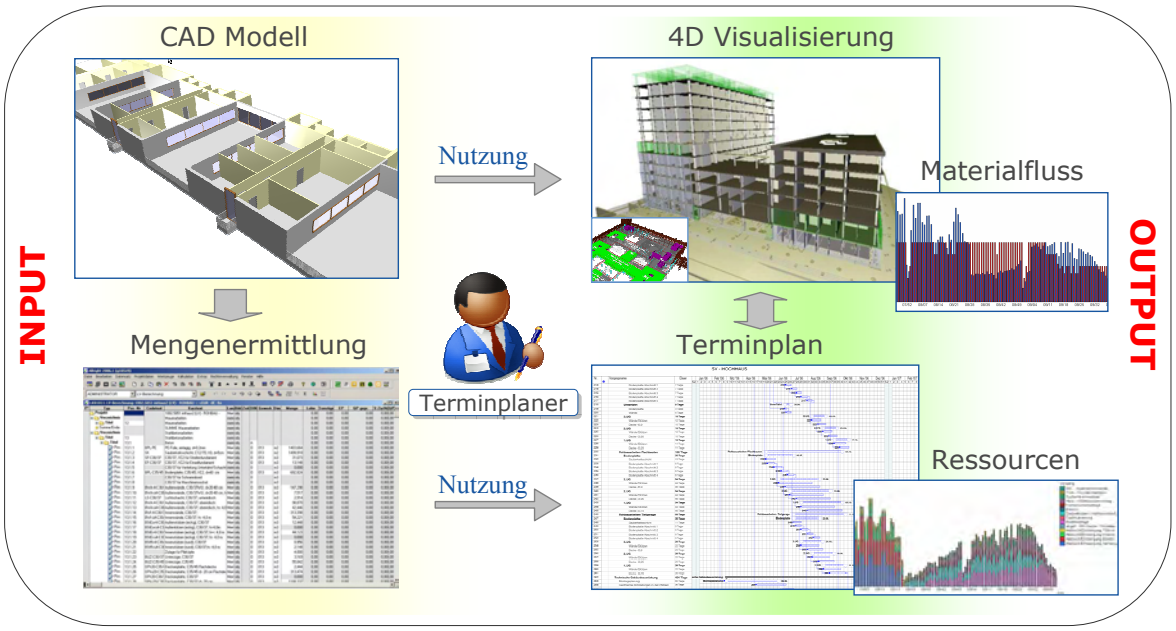


Abbildung 3.1: Wiederverwendung von im Modell gespeicherten Geometrie- und Mengeninformationen

3.2 Datenaustausch mit Projektpartnern

Grundvoraussetzung für die Wiederverwendung von Daten im Planungsprozess ist der ungehinderte Datenaustausch zwischen den beteiligten Projektpartnern in unterschiedlichen Firmen und Organisationen. Angestrebt wird ein einheitlicher Datenaustausch, der es ermöglicht, die von anderen Akteuren während oder nach der Planung bzw. Erstellung des Bauwerks benötigten Informationen zur Verfügung zu stellen. Darüber hinausgehende, im Laufe des Projektes erzeugte Daten, die nur für einzelne Akteure von Bedeutung sind, brauchen nicht ausgetauscht werden. Dieses gilt insbesondere für private, schützenswerte Informationen. Die Speicherung nicht auszutauschender Daten kann lokal und auf individuelle Art erfolgen. Die Speicherung auszutauschender Daten sollte hingegen zentral in einem einheitlichen Format erfolgen. Dabei besteht die Herausforderung darin, dass die Integration der gemeinsam genutzten Daten in ein Bauwerksinformationsmodell nur möglich ist, wenn durch das gewählte Datenmodell alle relevanten Informationen unterstützt werden. Für die im Rahmen der Terminplanung relevanten Bereiche Kalkulation, Architektur- und Terminplanung existiert jedoch traditionell eine strikte Trennung und damit jeweils eigene Datenmodelle und -formate. Dieses stellt ein Hindernis für die Zusammenarbeit dar.
Eine fachübergreifende Datenintegration und eine damit verbundene einheitliche Datenaustauschmöglichkeit bieten die Industry Foundation Classes (IFC). Hierbei handelt es sich um ein in EXPRESS[9] definiertes, objektorientiertes Datenmodell. Zum Austausch der Daten können spezielle STEP[10]- oder XML[11]-Dateien verwendet wer-

[9] Eine unter ISO 10303-11 standardisierte Datenmodellierungssprache
[10] STEP Physical File gemäß ISO 10303-21
[11] STEP-XML gemäß ISO 10303-28

den. Das IFC-Datenmodell ist bereits unter ISO/PAS[12] 16739 registriert und auf dem Weg, ein internationaler ISO-Standard zu werden. Die Spezifikation und Dokumentation dieses Datenmodells ist in verschiedenen Versionen frei verfügbar. Zusätzlich bietet es den Vorteil, dass es nicht von einem oder wenigen Softwareherstellern kontrolliert wird. Somit bieten sich allen Interessengruppen im Bauwesen die gleichen Nutzungs- und Weiterentwicklungsmöglichkeiten.
Derzeit gewinnt das IFC-Datenmodell auch in der Praxis beträchtlich an Bedeutung:

- Die meisten namhaften Bausoftwarehersteller liefern mit ihren Softwareprodukten IFC-Im- und Exportschnittstellen bereits mit aus.
- Das Bundesamt für Bauwesen und Raumordnung in Deutschland hat im Dezember 2006 den Datenaustausch mittels IFC in seiner "Baufachlichen Richtlinie Gebäudebestandsdokumentation"(BFR GBestand) für neu zu errichtende bzw. umzubauende Hochbauten mit aufgenommen.
- Die Bayerische Staatliche Bauverwaltung setzt seit 2007 verstärkt auf die IFC basierte Datenverarbeitung ([Pries 2007]).
- Die Baubehörde in Singapur (BCA) nutzt bereits ein IFC basiertes, elektronisches Planprüfsystem für Bauanträge ([buildingSMART 2003]).
- Verwaltungsbehörden öffentlicher Gebäude in den USA, Norwegen, Finnland und Dänemark haben eine Absichtserklärung zur Umsetzung der Nutzung von IFC in allen größeren Projekten innerhalb der nächsten zwei Jahre unterzeichnet ([GSA 2008]).
- Im international am weitesten verbreiteten Dokumentenaustauschformat PDF ist seit Version 9 die Einbettung und damit auch die (geschützte) Verbreitung, Betrachtung und Kommentierung von IFC-Daten möglich[13,14].
- In vielen weiteren Ländern gibt es nationale Gruppen (Chapters), die sich an der Entwicklung und der Erprobung von IFC beteiligen. Neben den bereits genannten sind dieses z.B.[15]: Österreich[16], Schweiz, Frankreich, England, Kanada, Australien und Japan.

Aufgrund der genannten Vorteile und der zunehmenden Verbreitung in der Bauindustrie erscheinen die IFC, insbesondere für den Datenaustausch über Unternehmensgrenzen hinweg, als geeignete Basis für die modellbasierte Zusammenarbeit. Die Verwendung eines solchen, offenen und standardisierten Formats für den Datenaustausch würde sowohl die immer wieder neue Zusammensetzung von Projektpartnern als auch deren individuelle Softwarewahl ohne negativen Einfluss auf die Zusammenarbeit ermöglichen. Daher ist die Abbildung der für die modellbasierte Terminplanung relevanten Daten auf die in IFC definierten Objekte zu untersuchen bzw. zu spezifizieren.

[12] Publicly Available Specification. Hierbei handelt es sich noch nicht um eine Norm, d.h. es ist noch kein Konsenz zwischen allen Interessensgruppen erreicht.
[13] http://www.adobe.com/go/kb403526
[14] http://www.adobe.com/designcenter/video_workshop/?id=vid0323
[15] http://cig.bre.co.uk/iai_uk/iai/page3.htm
[16] Eine Analyse von IFC2x3 hinsichtlich der Verwendbarkeit im Rahmen der österreichischen Hochbaurichtlinien findet man in [Stoiber 2007]

3.3 Schnittstelle zwischen projekt- und unternehmensbezogener Datenverarbeitung

Im Bereich der Terminplanung existieren prinzipiell zwei Sichtweisen. Die den Anforderungen des Projektes entsprechende Terminplanung entspricht der *Projektsicht*. Ihr Ziel ist der optimale Ablauf des Einzelprojektes. Dabei werden Eingangsinformationen über das spezielle Bauwerk und die Projektrahmenbedingungen benötigt. Als Ergebnis entsteht eine zeitliche Abfolge der geplanten Prozesse sowie eine Aufstellung über Höhe und zeitliche Verteilung der einzusetzenden Ressourcen. Die Aggregation letzterer Informationen über alle Projekte eines Unternehmens ermöglicht die Disposition von Personal und Maschinen sowie die Steuerung einer zentralen Beschaffung. Dieses ist Gegenstand der *Unternehmenssicht*, die die projektübergreifende Optimierung der Geschäftsabläufe eines Unternehmens verfolgt.
Über eine Vernetzung der Terminplaninformationen aller Projekte über einen Server[17] ist bereits heute eine unternehmensweite Auswertung und Disposition von Ressourcen möglich. In der Projektbearbeitung auf Basis eines gemeinsam genutzten Bauwerksinformationsmodells werden zudem die projektbezogenen Daten zusammengeführt. Dieses Modell ist Quelle für die im Rahmen der Terminplanung benötigten Eingangsinformationen. Somit bilden Terminplanungs- und 4D-Simulationssoftware zusammen die Schnittstelle zwischen der projektbezogenen und der unternehmensbezogenen Datenverarbeitung in der Terminplanung. Ein *gegenseitiger* Datenaustausch auf Basis einer bidirektionalen Schnittstelle, der einerseits die benötigen Eingangsinformationen aus dem Modell übermitteln und andererseits die Termindaten infolge Verknüpfung durch den Anwender im Bauwerksmodell verankern kann, würde die Brücke zwischen beiden Datenhaltungen schlagen. Auch der nach Kapitel 2.1.1 erforderliche Austausch von Terminplandaten zwecks Zusammenführung und Koordinierung von projektorientierten und produktionsorientierten Einzelterminplänen könnte über diese Schnittstelle erfolgen.
Heutige 4D-Simulationssoftware bietet in der Regel jedoch weder eine bidirektionale Schnittstelle zur Terminplanungssoftware noch verarbeitet sie alle für die Terminplanung relevanten Modellinformationen. Mengen- und Kostendaten, die von anderen Projektbeteiligten im Laufe des Planungsprozesses aus anderen Anwendungen bereitgestellt werden, können in 4D-Simulationssoftware nicht verarbeitet werden. Hier ist auf der Basis existierender Technologien eine Verbesserung zu erzielen. Dabei ist darauf zu achten, dass eine Trennung zwischen öffentlichen und privaten Daten an der Grenze zwischen Unternehmens- und Projektbereich erfolgt.

3.4 Versionierung

Im Projektverlauf werden Planungsdaten detailliert und iterativ angepasst sowie verschiedene Varianten betrachtet. Im weiteren Verlauf sind durch andere Planungspartner Änderungen in Basisinformationen auf Objektebene zu erkennen und bzgl. ihrer Auswirkung zu bewerten, um ggf. in deren Verantwortungsbereich erforderliche Modellanpas-

[17] z.B MS Project Server oder Primavera P6

sungen bzw. Erweiterungen vorzunehmen. Zusätzlich ist eine detaillierte Dokumentation von Änderungen am Modell während des Planungsverlaufs auf Objektebene erforderlich. Sie ermöglicht eine spätere Analyse von Planungsschritten und -entscheidungen, was nicht zuletzt auch in Streit- und Nachtragsfragen von großer Bedeutung ist.
Obwohl bereits Ansätze ([Abdalla u. Powell 1991], [Hall u. a. 1991], [Firmenich 2002], [Beer 2005], [Wang u. a. 2007], [Koch 2009]) und erste Softwarepakete ([Sari 2006],[Oracle 2007]) existieren, die das automatische Aufzeichnen, Verwalten und Austauschen einer Änderungshistorie auf Objektebene in Ingenieuranwendungen unterstützen, so bieten die heutzutage für die Terminplanung und 4D-Simualtion verwendeten Softwarepakete keine entsprechenden Funktionalitäten. Das Erkennen fremder sowie das Verwalten eigener Änderungen erfolgt vielmehr manuell und auf Dokumentenebene unter erheblichem Arbeits- und Speicheraufwand. Eine Kennzeichung von Änderungen beim Austausch erfolgt selten und unzureichend.
Es ist daher ein generischer Ansatz für die zumindest lokale Verbesserung dieser Situation zu entwickeln, bei dem ein automatischer Vergleich von unterschiedlichen Planungsständen und eine Verringerung des Speicheraufwands bei der Verwaltung der eigenen Datenhistorie erreicht wird. Dabei ist es nicht erforderlich, den gesamten Datenbestand zu versionieren. Eine Konzentration auf das Erkennen und Verwalten von im Rahmen des betrachteten Anwendungsbereichs (hier der modellbasierten Terminplanung) relevanten Änderungen ist ausreichend. Das Problem des Zusammenführens von konkurrierenden, d.h. gleichzeitig von verschiedenen Projektbeteiligten am gleichen Datenbestand vorgenommenen Änderungen kann durch eine geeignete Workflow-Definition und die Aufteilung von Datenänderungsrechten an Modelldaten weitestgehend vermieden werden. Es wird daher im Rahmen dieser Arbeit nicht weiter betrachtet und diesbezüglich auf die oben angegebene Literatur verwiesen.
Zum Ermitteln formaler und semantischer Unterschiede zweier Datenbestände existieren verschiedene Verfahren ([Weise 2006], [Vad 2007]). Die Nutzung eines objektorientierten Datenmodells, in dem die einzelnen Objekte persistent identifiziert sind, ist jedoch die Voraussetzung, um solch eine Analyse überhaupt bzw. effizient durchführen zu können. Im Datenmodell der IFC ist das für alle Objekte sichergestellt, die sich von IfcRoot ableiten.
Sind Änderungen in den Basisinformationen bekannt, können Auswirkungen auf die darauf aufbauenden Daten analysiert bzw. diese aktualisiert werden. In [Hanff 2003] wird ein Konzept für die automatische Aktualisierung von Planungsdaten beschrieben. In der Praxis erfolgt dieser Schritt in der Regel manuell. Das ist zu einem gewissen Maß auch erwünscht, um weitreichende Auswirkungen nachvollziehbar und beherrschbar zu halten. Zudem können Änderungen in Basisdaten unter Umständen eine Änderung in der Plangungsstrategie erfordern, die menschliches Beurteilungsvermögen erfordert. Der erforderliche manuelle Aufwand für die Durchführung von Aktualisierungen sollte jedoch durch einen kontrollierten, automatischen Mechanismus minimiert werden. Dieses gilt insbesondere für die Aktualisierung der Verknüpfungen zwischen Terminplan und Bauwerksmodell.

3.5 Verknüpfungsmethodik

Die Erstellung der Verknüpfung von Vorgängen des Terminplans mit Informationen des Bauwerksmodells stellt den Hauptarbeitsschritt bei der Wiederverwendung von Modellinformationen im Rahmen der Terminplanung dar. Wie in Kapitel 2.1.6 erläutert, weisen jedoch die bisher existierenden Methoden gewisse Nachteile auf. Im Einzelnen wurden folgende Kritikpunkte identifiziert:

- hoher Aufwand für die Erstellung der Verknüpfungen,
- ähnlich hoher Aufwand bei der Aktualisierung der Verknüpfungen,
- Erfordernis der Bedienung eines CAD-Programms zur Durchführung notwendiger Modellanpassungen bzw. hoher Koordinierungsaufwand mit anderen Projektpartnern zur
  - Eingabe zusätzlicher Informationen als Grundlage einer automatischen Verknüpfung oder zum Generieren eines Terminplans,
  - Herstellung einer kompatiblen Granularität von Objekten des Bauwerksmodells mit den Vorgängen im Terminplan,
- unzureichende Zielgenauigkeit bei der attributbasierten Objektauswahl,
- Schwierigkeit des Einhaltens einer vorgegebenen Gliederungsstruktur für den Terminplan sowie der Eingliederung von nicht mit dem Bauwerksmodell in Verbindung stehenden Vorgängen (bei automatischer Terminplangenerierung aus dem Modell).

Zudem konnten aus der Prozessanalyse folgende Anforderungen für die Verknüpfungsmethodik abgeleitet werden, die bisher kaum Berücksichtigung finden:

- Neben den Geometrieobjekten sind auch Mengen oder Kosten mit Terminplanvorgängen zu verknüpfen, um neben der Visualisierung weiteren Mehrwert aus der Verknüpfungserstellung zu generieren. Der örtliche Bezug von Mengeninformationen kann dabei für die Zuordnung zu Bauabschnitten genutzt werden.
- Die Verknüpfung mehrerer Terminpläne mit dem gleichen Bauwerksmodell sollte unterstützt werden. Das wesentliche Problem besteht dabei in der Sicherstellung einer mit allen Terminplänen kompatiblen Objektgranularität des Bauwerksmodells.
- Unabhängig von der Erstellung von Verknüpfungen ist eine Filterung des Gesamtdatenbestands auf eine dem Verwendungszweck entsprechende Teildatenmenge erforderlich. Dieses erfolgt z.B. auch zur Erstellung von Teilterminplänen oder zur Auswahl konkreter Werte aus einer Aufwandswertedatenbank.

Von den existierenden Verknüpfungsmethoden erscheint die Formulierung von attributbasierten Regeln (bisher in Form von Datenbankanfragen) zur Bestimmung der mit einem Terminplanvorgang zu verknüpfenden Objekten am ehesten den genannten Anforderungen gerecht werden zu können. Je nach Anzahl der benötigten Informationen aus dem Bauwerksmodell können auf diese Weise pro Terminplanvorgang ein oder mehrere Regeln definiert werden, die die zu verknüpfenden Objekte bestimmen. Werden

für einen Terminplanvorgang keine Informationen benötigt, so ist auch keine Regel zu definieren. Die Gliederungsstruktur des Terminplans wird durch die Nutzung von Verknüpfungsregeln nicht beeinflusst. Zudem wird durch diese Methode eine Verringerung des Aufwands für die Erstellung von Verknüpfungen und deren Aktualisierung erreicht. Das ist der Fall, da die zu verknüpfenden Objekte nicht einzeln ausgewählt werden müssen, sondern dieses durch Auswertung der durch den Anwender definierten Regel automatisch geschieht. D.h., es wird auf Basis der Regel eine Teilmenge der im Bauwerksmodell enthaltenen Objektmenge berechnet. Diese Objekte sind zu verknüpfen. Bei der Reaktion auf Änderungen im Bauwerksmodell müssen lediglich die Regeln neu interpretiert werden. Dabei können folgende Fälle auftreten:

- Es wurden aus dem Bauwerksmodell Objekte entfernt, die mit Terminplanvorgängen verknüpft waren. Die Verknüpfungen werden automatisch entfernt.
- Es sind neue Objekte im Bauwerksmodell enthalten, die den Kriterien einer Verknüpfungsregel entsprechen. Die neuen Objekte werden automatisch mit verknüpft.
- Es sind neue Objekte im Bauwerksmodell enthalten, die von keiner Verknüpfungsregel erfasst werden. In diesem Fall ist die Anpassung bzw. Neuerstellung von Terminplanvorgängen und Verknüpfungsregeln erforderlich.

In den ersten beiden Fällen muss lediglich eine Überprüfung durch den Terminplaner erfolgen, ob die sich neu ergebende Zuordnung ohne weitere Anpassungen sinnvoll ist oder ob eine Umplanung der Bauabschnitte oder gewählten Kapazitäten bzgl. Personal, Gerät und Material erforderlich ist.
Verbleibende Kritikpunkte der regelbasierten Verknüpfungserstellung sind die ungenügende Zielgenauigkeit aufgrund der ausschließlich alphanumerischen Filterung ohne Berücksichtigung der räumlichen Lage sowie die Notwendigkeit zur vorhergehenden Anpassung der Objektgranularität im CAD-System. Ohne letzteres ist die Addressierung eines Teils des Bauwerksmodells auf die Grenzen filterbarer Teilmengen von Objekten der vorgegebenen Objektgranularität beschränkt. Dieses ist für die in der Terminplanung erfolgende Einteilung des Bauwerks in den technischen Erfordernissen entsprechende Bauabschnitte unzureichend.
Um einen leistungsfähigen Mechanismus zur Verfügung zu stellen, soll der existierende Ansatz der regelbasierten Verknüpfung in Form einer Verknüpfungs- bzw. Auswertungssprache erweitert werden. Diese Sprache soll im Wesentlichen drei Funktionalitäten vereinen (siehe Abbildung 3.2 auf der nächsten Seite):

- attributbasierte Filterung von Objekten,
- räumliche Anfragefunktionalität zur Verbesserung der Zielgenauigkeit,
- automatische Anpassung der Objektgranularität, wenn dieses im Kontext der räumlichen Anfrage erforderlich ist.

Mit diesen drei Funktionalitäten soll es dem Terminplaner ermöglicht werden, ohne Berücksichtigung von Zwangsbedingungen einen beliebigen Teil des Bauwerksmodells adressieren zu können. Somit wird auch eine Verknüpfung mit mehreren Terminplänen ohne weiteres unterstützt. Zusätzlich ist die Sprache so zu gestalten, dass für den jeweils angesprochenen Bauwerksteil unterschiedliche Informationen angefordert werden

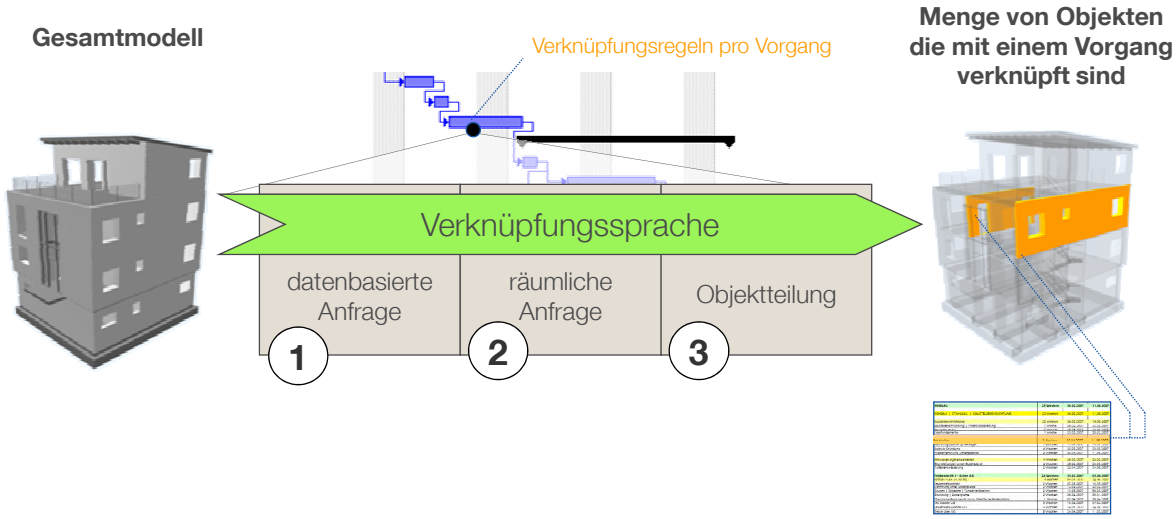


Abbildung 3.2: Prinzip der Verknüpfungssprache

können (Geometrie, Attribute, Kosten, Mengen, usw.). Weiterhin sind für die Verarbeitung dieser Informationen geeignete Operationen bereitzustellen. Hierzu gehören z.B. das Auswählen, Extrahieren oder Verknüpfen von Objekten sowie das Addieren von Mengeninformationen oder das Zählen adressierter Objekte. Zudem ist dem Terminplaner eine geeignete Oberfläche zur vorgangsbezogenen Definition von Regeln zur Verfügung zu stellen.

3.6 Visualisierung

Im Bereich der Visualisierungsmöglichkeiten besteht noch vielfältiger Entwicklungsbedarf, um die Erstellung und das Verständnis von 4D-Simulationen zu erleichtern. In Kapitel 2.1.6 wurde hierauf bereits eingegangen und entsprechende Verbesserungsvorschläge vorgestellt. Dringend erforderlich erscheint ein Konzept, das es ermöglicht, Ausbauvorgänge beobachten zu können. Auch die Realisierung einer über die Zeit kontinuierlichen Anpassung der Objektdarstellung (Farbverlauf, Skalierung, Schnitteben, Bewegung) und dessen effizienter Definition durch den Anwender, eventuell aufbauend auf die im weiteren vorgestellte Verknüpfungssprache, scheint Raum für die Entwicklung einer generalisierten Methodik zu bieten. Ebenso wäre eine darauf aufbauende Kollisionsprüfung bewegter Objekte ([Lai u. Kang 2009], [Tantisevi u. Akinci 2009]) denkbar.
Im weiteren Verlauf dieser Arbeit wird jedoch auf das Thema Visualisierung nicht weiter eingegangen.

4 Lösungsansatz

Nachdem in den vorhergehenden Kapiteln die Anforderungen und bisher existierenden Hindernisse für eine modellbasierte Terminplanung entsprechend der Definition aus Kapitel 1.2 diskutiert wurden, werden nachfolgend die einzelnen Komponenten einer durchgängigen Lösung vorgestellt. In einem Systementwurf wird das Zusammenspiel der beteiligten Softwarepakete betrachtet und anschließend der dadurch unterstützte Soll-Prozess und seine Akteure beschrieben. Nach der anschließenden datentechnischen Modellbildung, die die Grundlage für die spätere Umsetzung bildet, erfolgt eine detaillierte Beschreibung der verarbeiteten Daten. Darauf aufbauend werden eine Sprache und die zugehörigen Algorithmen für die Extraktion bzw. Verknüpfung von Modelldaten mit Terminplanvorgängen spezifiziert. Abschließend wird ein Konzept für die Versionierung von Planungsdaten zur Unterstützung des Anwenders beim iterativen Durchlauf des Planungsprozesses vorgestellt.

4.1 Systementwurf

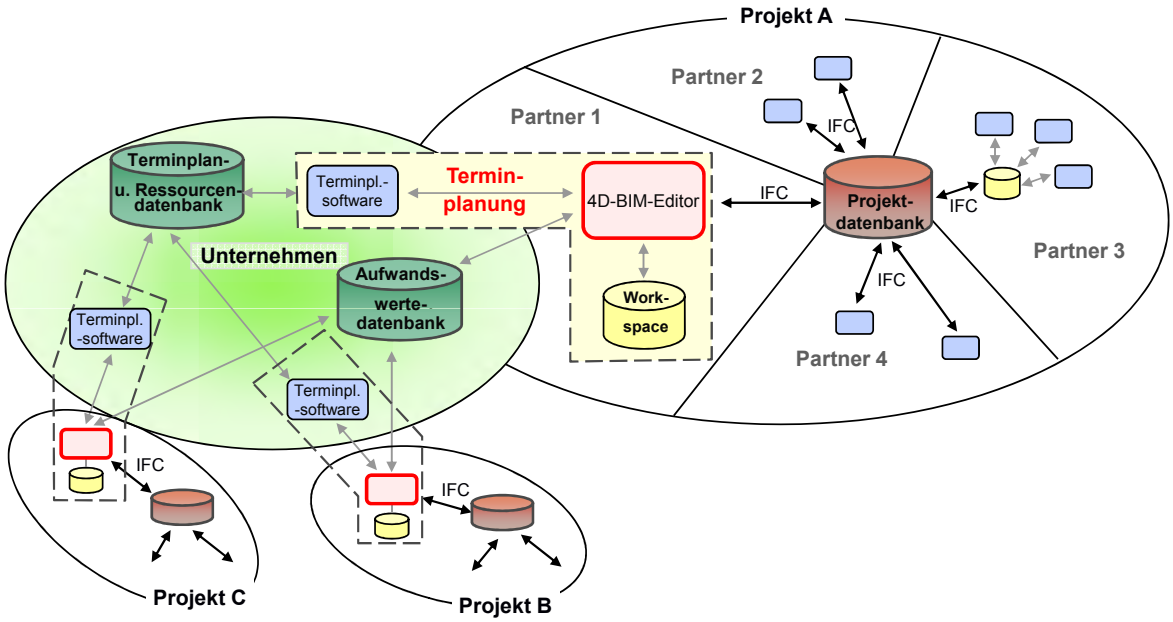


Abbildung 4.1: Softwareinfrastruktur; Terminplanungsumgebung als Brücke zwischen Projekt- und Unternehmenssoftwarelandschaft

Die modellbasierte Terminplanung erstreckt sich über einen unternehmensbezogenen und einen projektbezogenen Einflussbereich (siehe Abbildung 4.1). Die zentrale Be-

reitstellung und der Austausch von strukturierten Daten[18] erfolgt mittels mehrerer Datenserver. Die verwendeten Server können anhand folgender Kriterien unterschieden werden:

- Einflussbereich, in dem sich der Server physisch befindet (Projektbereich/Unternehmensbereich).
- Zeitdauer, für die Zugriff auf den Server besteht.
- Art der hinterlegten Daten:
  - private und/oder (im Projekt) öffentliche Daten,
  - volle oder verminderte Detaillierung von Daten,
  - Daten eines oder mehrerer Projekte,
  - Daten einer oder mehrerer Fachdisziplinen.
- Personenkreis, der Zugriff auf die hinterlegten Daten erhält.
- Art des Datenaustauschs mit anderen Softwarepaketen (proprietär oder auf Basis eines standardisierten, neutralen Datenformats).

Projektbezogener Einflussbereich: Im einzelnen Projekt erfolgt der Datenaustausch über einen Modellserver. Dieser vereint die für die Zusammenarbeit benötigten Daten aller Fachdomains und ist somit die einzige Stelle, an der ein Gesamtmodell des Bauwerks vorliegt. Der Datenbestand liegt jedoch nicht in vollem Detailumfang vor, sondern weist nur die im Rahmen der Zusammenarbeit öffentlichen Informationen auf. Darüber hinausgehende Detailinformationen der Planung verbleiben in der Datenhaltung der einzelnen Fachplaner. Hierbei handelt es sich um private Daten. Um auf Planungsänderungen reagieren zu können, ist die Integrität privater und öffentlicher Daten zu wahren.
Sämtliche an der Planung, Ausführung und dem Betrieb beteiligte Unternehmen erhalten über die Zeit ihrer Tätigkeit im Projekt einen ihrer Rolle entsprechenden Zugriff auf die Daten des Modellservers. Jedes Unternehmen verwendet zur Bearbeitung seine bevorzugte Software. Um eine Kompatibilität mit sämtlichen Softwarepaketen verschiedener Projektpartner zu erreichen, erfolgt der Datenaustausch mit dem Modellserver auf Basis eines neutralen, standardisierten Datenformats. Der Datenaustausch zwischen verschiedenen Fachanwendungen innerhalb eines Unternehmens kann hingegen individuell erfolgen, da es sich hier um eine stabile Infrastruktur handelt.
Der Modellserver ist über den gesamten Lebenszyklus hinweg Grundlage der auf das spezielle Bauwerk bezogenen Datenverarbeitung. Er sollte daher stets unter der direkten Verantwortung des Bauwerkseigentümers stehen. Die enthaltenen Daten werden ihm von den Fachplanern zur freien Nutzung übergeben.

Unternehmensbezogener Einflussbereich: Im Rahmen der Terminplanung werden in einem Unternehmen zwei zentrale Datenhaltungen vorgenommen. Der Anschluss

[18] Weiterhin kommen Systeme für Dokumentenmanagement, Referenzenmanagement und Workflowmanagement zum Einsatz, die im Rahmen dieser Arbeit nicht weiter betrachtet werden. Siehe hierzu [Liebich u. a. 2008]

der Terminplanungssoftware an einen Datenserver dient der projektübergreifenden Auswertung und Disposition von Ressourcen sowie der Zusammenarbeit mehrerer Mitarbeiter eines Unternehmens am selben Projekt. Verwaltet werden auf diesem Server ausschließlich alphanumerische Daten der Termin- und Ressourcenplanung aller Projekte eines Unternehmens. Die Daten werden über die jeweilige Projektlaufzeit hinaus in vollem Detailumfang vorgehalten, d.h. inkl. detaillierter Berechnungsformeln. Musterterminpläne werden ebenfalls auf diesem Server hinterlegt. Dadurch wird ein Zugriff auf Vorlagen und Vergleichsdaten ermöglicht. Eine Verwaltung von Modelldaten erfolgt nicht. Lediglich die aus dem Modell extrahierten Basisinformationen, die Teil der Berechnungsformeln für die Dauer von Vorgängen sind, werden mit verwaltet. Eine zweite zentrale Datenverwaltung dient der Bereitstellung unternehmenseinheitlicher Aufwandswerte in Katalogform. Beide Datenhaltungen werden von mehreren Projekten gleichzeitig genutzt. Sie enthalten private Daten und sind daher nur Mitarbeitern des Unternehmens zugänglich. Der Datenaustausch kann, abgestimmt auf die im Unternehmen eingesetzte Software, proprietär erfolgen.

Schnittstelle zwischen Projekt- und Unternehmensbereich: Während der Projektbearbeitung greift der Terminplaner auf alle drei zuvor genannten Datenserver zu. Er nutzt unternehmensinterne Vorlagen und Informationen über die Verfügbarkeit von Ressourcen sowie Aufwandswerte und Berechnungsformeln für die Terminplanerstellung. Zusätzlich erstellt er für die Extraktion von Mengen bzw. Kosten und die Erstellung von 4D-Simulationen Verknüpfungen mit dem Bauwerksmodell des spezifischen Projektes. Das dabei entstehende integrierte Modell enthält für die vom Terminplaner bearbeiteten Vorgänge alle erforderlichen Einzelfaktoren und Verknüpfungen mit dem Modell. Es ist somit Grundlage für Anpassungen infolge Planungsänderungen. Aufgrund der enthaltenen privaten Berechnungsgrundlagen ist es jedoch nicht öffentlich. Es bedarf daher einer projektbezogenen Speicherung im Einflussbereich des Unternehmens, geschützt vor dem Zugriff durch externe Partner. Zur Unterstützung des Änderungsmanagements ist diese Datenhaltung zu versionieren. Durch das Vorhalten der Historie der Plangungsdaten erfolgt zugleich eine für Streitfragen ohnehin erforderliche Dokumentation des Planungsgeschehens auf Unternehmensseite. Für diese Speicherung und Versionierung wird ein sog. Workspace verwendet. Dieser enthält die für die Terminplanung relevanten Daten mehrerer Fachdisziplinen eines Projektes. Eine Auswertung über mehrere Projekte ist nicht erforderlich und wird vom Workspace nicht unterstützt. Ein Zugriff auf die Daten ist nur bis Projektabschluss erforderlich. Im Workspace können mehrere interne Zwischenstände abgespeichert werden, bevor eine Version auf dem Projektserver veröffentlicht wird.
Der Terminplaner arbeitet parallel, ggf. auf zwei Computerbildschirmen, mit zwei Softwarepaketen: der Terminplanungssoftware und einem 4D-BIM-Editor, die miteinander über eine interaktive, bidirektionale Schnittstelle Daten austauschen (Abbildung 4.2 auf der nächsten Seite). In der Terminplanungssoftware erfolgt die Terminplanerstellung sowie der Zugriff auf die Terminplan- und Ressourcendatenbank. Im 4D-BIM-Editor erfolgt die Verarbeitung von Modelldaten und die Erstellung von Verknüpfungen zwischen Terminplan und Modell, sowie die Durchführung von 4D-Simulationen und die Analyse weiterer Informationen, wie z.B. des Materialverbrauchs über die Zeit. Da der 4D-BIM-Editor alle erforderlichen Daten verarbeitet, erfolgt hierüber auch die Kommunikation mit dem Workspace. Die Anbindung der Aufwandswertedatenbank kann

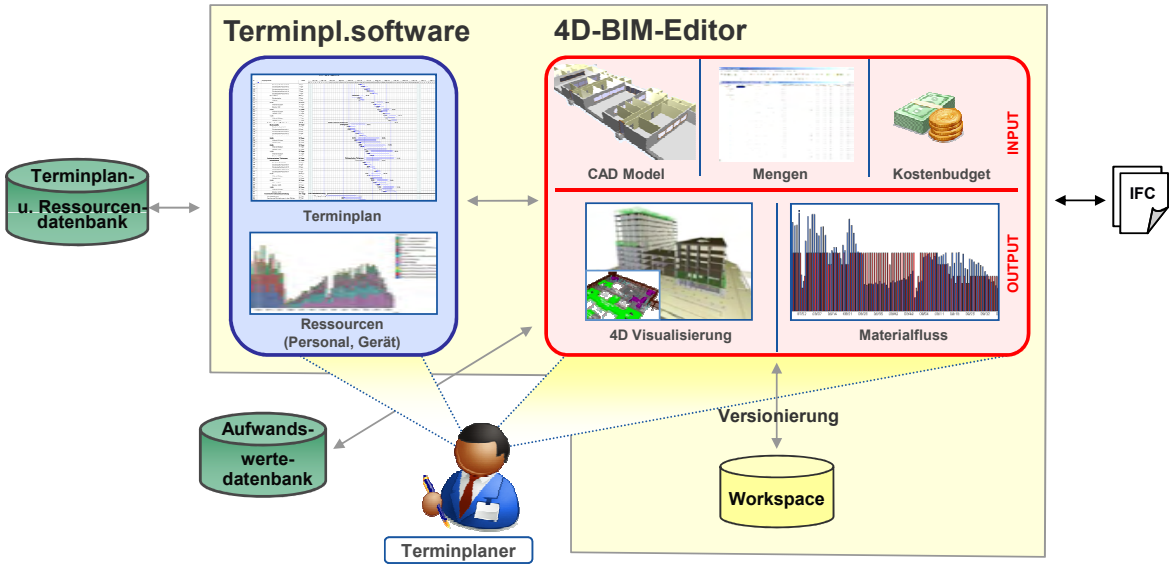


Abbildung 4.2: Lokale Softwareinfrastruktur für die modellbasierte Terminplanung (vgl. auch Abbildung 4.1)

wahlweise über eines der beiden Softwarepakete erfolgen. Im Rahmen dieser Arbeit wurde die Anbindung an den 4D-BIM-Editor gewählt.
Die unternehmensinterne Datenkommunikation kann über proprietäre Schnittstellen und Datenformate erfolgen. Die Kommunikation mit dem Projektmodellserver erfolgt auf Basis des neutralen Datenformats IFC. Ein Datenaustausch erfolgt in beide Richtungen. Importiert werden Basisdaten sowie Terminplandaten einer vorhergehenden Iteration oder anderer Planungspartner. Exportiert werden geänderte sowie neu erstellte Daten. Nur der öffentliche Teil des eigenen Datenbestands wird auf dem Projektmodellserver veröffentlicht.

4.2 Prozess und Akteure

Der für die Terminplanung relevante Teil des Gesamtplanungsprozesses und dessen Akteure werden nachfolgend erläutert (siehe Abbildung 4.3 auf der nächsten Seite).

Kunde: Der Auftraggeber des Projektes gibt aufgrund seines Nutzungs- und Finanzierungskonzeptes funktionale, terminliche und ablauftechnische Anforderungen für das Bauprojekt vor. Während des Planungsprozesses erfolgt seinerseits eine stetige Kontrolle von Bauwerksdesign, Kosten und Terminen. Bei auftretenden Widersprüchen oder Problemen werden von ihm ggf. Anpassungen der Anforderungen vorgenommen. Die Kundenvorgaben werden in Form von Dokumenten im Projekt kommuniziert.

Architekt bzw. Konstukteur: Der Architekt oder Konstrukteur erzeugt auf Basis der funktionalen Kundenanforderungen ein geometrisches, dreidimensionales Computermodell des Bauwerks. Dieses wird zur leichteren Koordinierung der Zusammenarbeit räumlich gegliedert. Eine Gliederung erfolgt in der Regel nach Bauwerksabschnitten und Gebäudegeschossen. Zusätzlich wird vom Architekt ein Achsraster zur Adressierung einzelner Bauteile erstellt. Eine Berücksichtigung der Bauwerkserstellung und

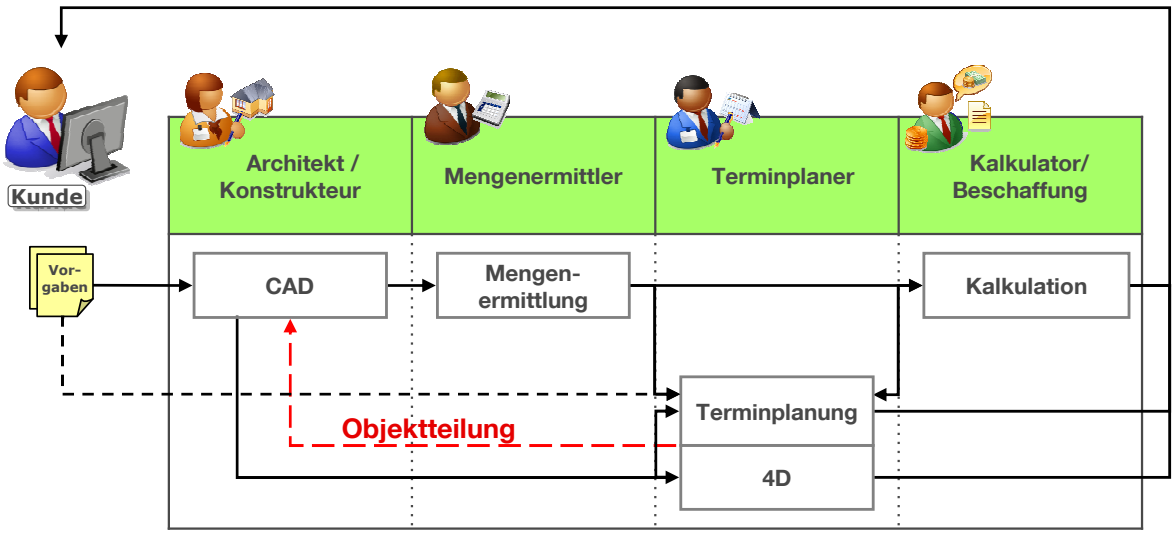


Abbildung 4.3: Einbettung der Terminplanung in den Gesamtplanungsprozess

eine damit verbundene Einteilung in Bauabschnitte erfolgt jedoch zu diesem Zeitpunkt nicht. Die gewählte Aufteilung des Modells in CAD-Objekte entspricht der Maßgabe einer möglichst effizienten Modellerstellung.
Zur weiteren Spezifikation einzelner Bauteile werden den CAD-Objekten Attribute, wie z.B. eine Materialangabe oder eine erforderliche Feuerschutzklasse, zugeordnet. Weiterhin werden Raumobjekte erstellt und klassifiziert, für die in einem separaten Raumbuch die typbezogene Beschreibung des Innenausbaus erfolgt. Dem Architekturmodell können bereits einfache Mengeninformationen, wie die Abmaße und das Volumen einzelner Bauteile oder die Grundflächen von Raumobjekten, entnommen werden. Diese Mengen werden als Basismengen bezeichnet.

Mengenermittler: Hierbei handelt es sich um eine im Rahmen der modellbasierten Arbeitsweise neu entstehende Rolle, die Kenntnisse eines Konstrukteurs und eines Kalkulators erfordert. Auf Basis des Architekturmodells und eines Raumbuches werden mittels vom Mengenermittler angegebener Formeln kostenrelevante Mengen automatisch ermittelt und im Leistungsverzeichnis (LV) zusammengestellt. Dieses ist Basis für die spätere Kalkulation. Die LV-Mengen sind gegenüber den Basismengen wesentlich detaillierter und berücksichtigen vorgeschriebene nationale Berechnungsmethoden. Die Granularität der räumlichen Zuordnung von LV-Mengen entspricht der Granularität des zugrunde gelegten Architekturmodells. Zusätzlich können mittels Formeln Mengen abgeleitet werden, die keine direkte Entsprechung im Architekturmodell besitzen (z.B. Anzahl von Bewehrungseisen ermittelt anhand von Bauteilabmessungen).

Kalkulator: In einer frühen Projektphase erstellt der Kalkulator auf Basis von Erfahrungswerten aus Vergleichsprojekten eine Kostenschätzung für die einzelnen Gewerke. In einer späteren Projektphase wird in Zusammenarbeit mit der Beschaffung auf Grundlage der von der Mengenermittlung bereitgestellten LV-Mengen eine Kostenkalkulation erstellt. Weitere Kosteneinflußfaktoren sind der von der Terminplanung ermittelte Personal- und Gerätebedarf.

Terminplaner: Der Terminplaner erstellt und aktualisiert im Laufe des Projektes den Terminplan. Abhängig vom aktuellem Planungsstand verwendet er hierzu verschiedene

Ausgangsdaten (siehe auch Kapitel 2.1.3). Im Einzelnen sind dies die LV-Mengen der Mengenermittlung (Methode A), die Basismengen des Architekturmodells (Methode B) oder die Kostenschätzung der Kalkulation (Methode C). Zwischen Terminplanung und Kalkulation ergibt sich somit je nach Bearbeitungsstand ein wechselnder Informationsfluss. Für die Erstellung einer 4D-Simulation verwendet der Terminplaner zusätzlich die Geometrie des Architekturmodells.
Ist ein Architekturmodell verfügbar, so wird es vom Terminplaner in einzelne Bauabschnitte untergliedert. Dieses erfolgt einerseits zur realistischen Visualisierung und andererseits zur Ermittlung der zu einem Bauabschnitt gehörenden Mengen als Basis für die Berechnung der Dauer der Vorgänge. Da die Erstellung des Modells zuvor ohne Kenntnis der Einteilung in Bauabschnitte erstellt wurde, wird zu diesem Zeitpunkt eine Anpassung der Objektgranularität erforderlich. Bietet die vom Terminplaner verwendete Software keine entsprechende Funktionalität zur Anpassung des Modells, so wird hierfür jeweils eine Iteration im Prozess über zwei vorherige Akteure erforderlich (Abbildung 4.3). Dieses stellt ein bedeutendes Hindernis dar. Um es zu beheben, muss der verwendete 4D-BIM-Editor eine Funktionalität zur automatischen Unterteilung von Objekten auf Basis der gewählten Bauabschnitte bieten.
Wie in Kapitel 2.1.1 beschrieben, gliedert sich der Terminplanungsprozess in die arbeitsteilige Erstellung des *projektorientierten* und *produktionsorientierten* Terminplans. Hierzu hat innerhalb des Teilprozesses der Terminplanung ein Datenaustausch zwischen Terminplaner, Nachunternehmern und Projektleitung zu erfolgen.

Zusammenfassend ergibt sich die in Abbildung 4.4 gezeigte Reihenfolge der Datenkommunikation zwischen den einzelnen Teilprozessen mittels eines Projektmodelldatenservers. Bei Änderungen in einem der Teilprozesse ist eine Iteration über nachfolgende Teilprozesse erforderlich. Der Kunde hat ständigen Zugriff auf die Planungsdaten, nimmt jedoch selbst keine Änderungen vor.

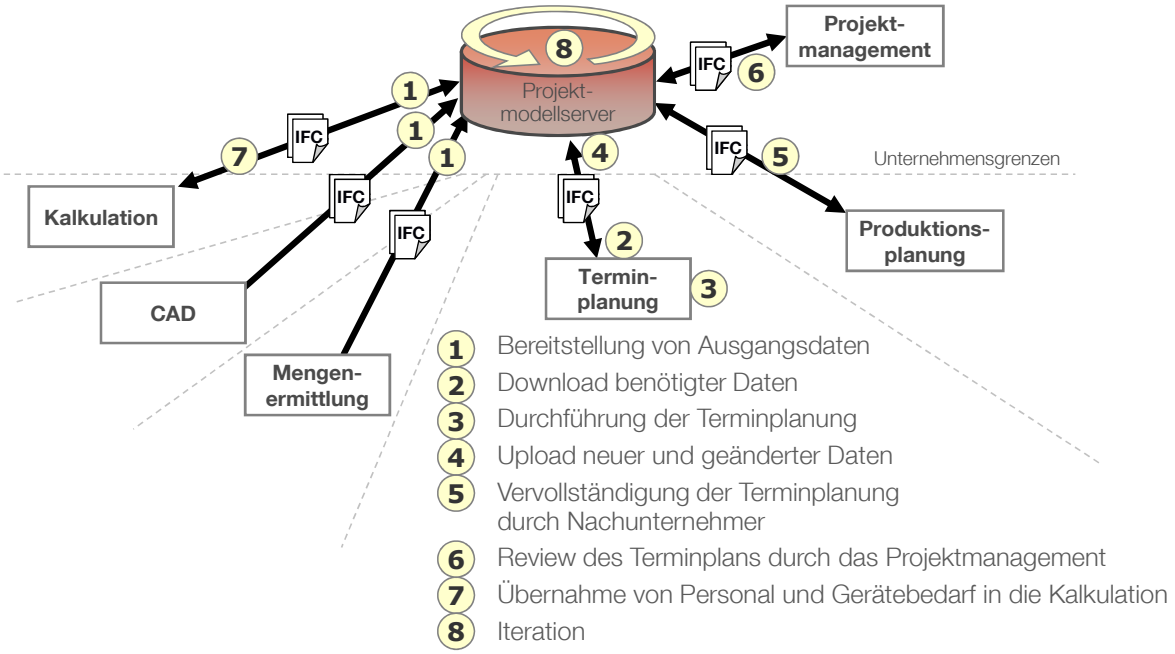
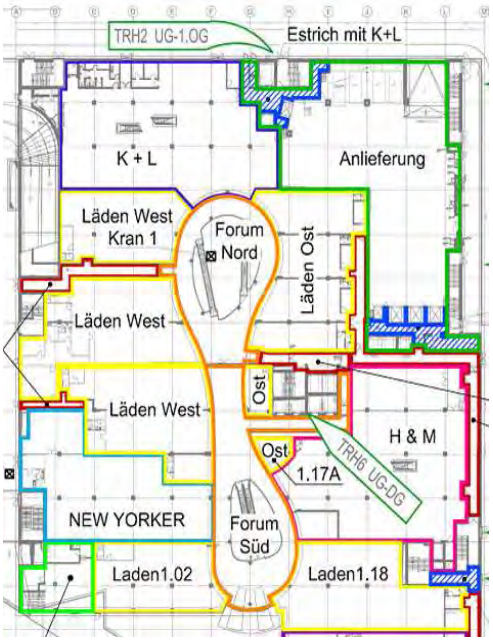


Abbildung 4.4: Bearbeitungszyklus in der modellbasierten Terminplanung unter Nutzung eines Projektmodellservers

zeichnungsbasiert



modellbasiert

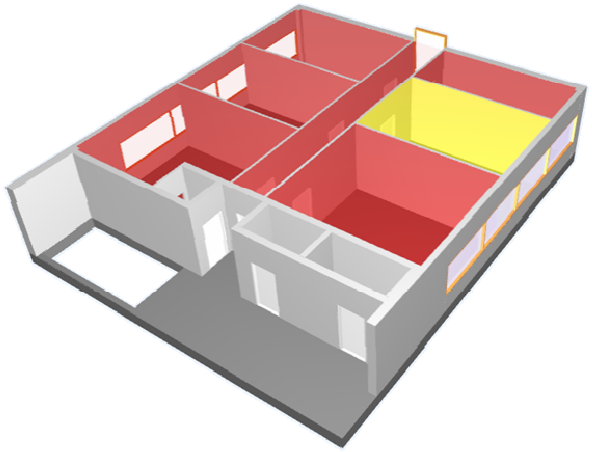


Abbildung 4.5: Definition von Bauabschnitten für den Ausbau

4.3 Gliederung in Bauabschnitte

Das Vorgehen bei der Gliederung des Bauwerks in Bauabschnitte kann von der zweidimensionalen Arbeitsweise in die dreidimensionale Arbeitsweise übernommen werden. Dabei ist eine Unterscheidung von Ausbau- und Rohbauvorgängen vorzunehmen.
Abbildung 4.5 zeigt links die beim zeichnungsorientierten Arbeiten übliche Art der Kennzeichnung von Arbeitsbereichen für den Innenausbau, hier beispielhaft für den Bereich eines Einkaufszentrums. Die auszubauenden Bereiche werden farbig umrandet und mit einer eindeutigen Bezeichnung versehen. Dadurch können die Ausbaubereiche im Beschreibungstext von Terminplanvorgängen leichter referenziert werden. Diese Art der räumlichen Adressierung ist unmittelbar auf die dreidimensionale, modellbasierte Arbeitsweise übertragbar. Um den durch umschließende Bauteile gebildeten Raum zu adressieren, werden im Architekturmodell enthaltene Raumobjekte verwendet und diesen die Raumbezeichnung als Attribut zugeordnet. Diese Arbeitsweise wird bereits von allen gängigen 3D-CAD-Programmen unterstützt. Eine Zuordnung von Raumobjekten zu Terminplanvorgängen kann somit später mittels attributbasierten Modellanfragen erfolgen. Zudem sind Raumobjekte Grundlage für eine modellbasierte Mengenermittlung. Somit sind auch die für die Ermittlung der Dauer von Vorgängen erforderlichen Mengeninformationen mit Raumobjekten verknüpft.
Das Vorgehen bei Rohbauvorgängen unterscheidet sich hiervon, da Bauabschnitte des Rohbaus in der Regel nicht der endgültigen Gebäudegliederung entsprechen. Die Kennzeichnung von Bauabschnitten erfolgt daher durch Zonen, die den technischen Erfor-

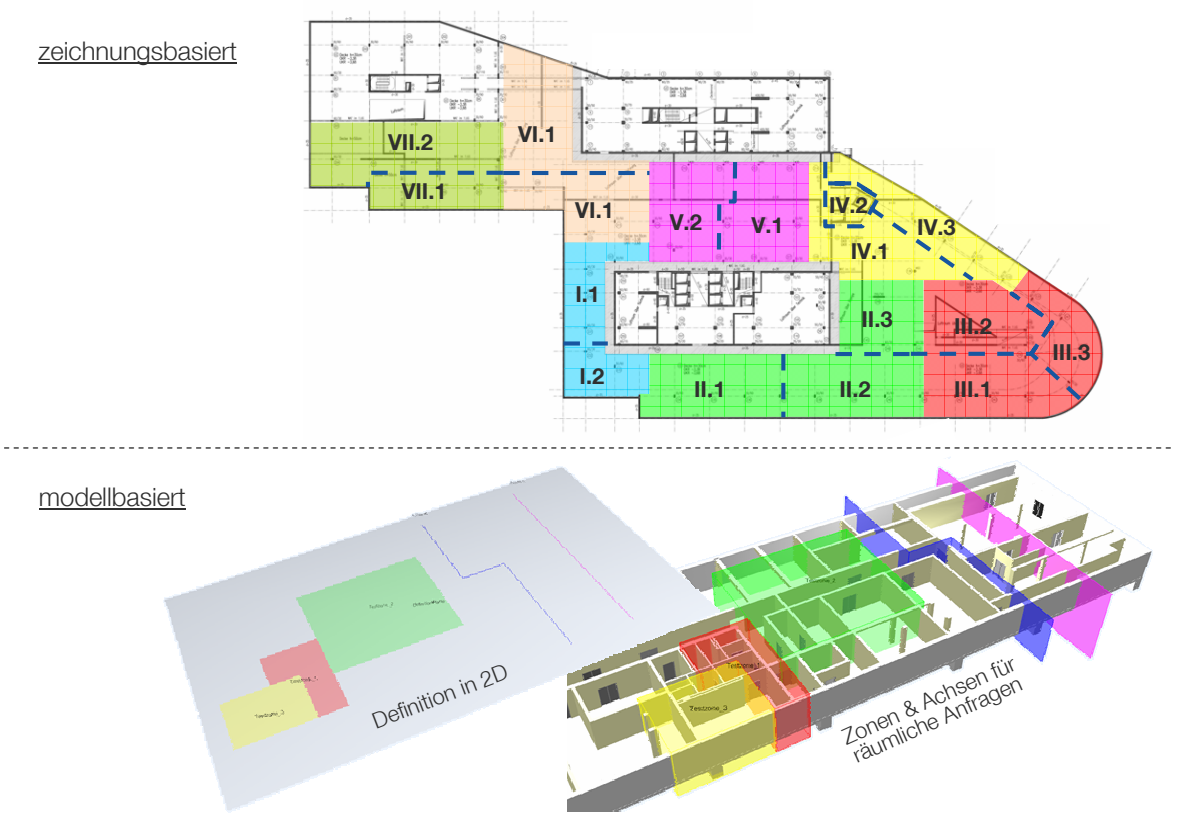


Abbildung 4.6: Definition von Bauabschnitten für den Rohbau

dernissen entsprechen und vom Terminplaner frei definiert werden. Alternativ wird auf ein vom Architekten angelegtes Achsraster Bezug genommen. D.h., Betonierabschnitte oder zu erstellende Einzelbauteile werden durch die örtliche Lage innerhalb eines Achsrasters angesprochen, z.B. „Deckenabschnitt und Stützen zwischen Achse D-H“. Dabei werden im Bauwesen nicht nur ausschließlich geradlinige, sondern auch polygonal verlaufende Achsen verwendet. Diese Arbeitsweise ist wie folgt ins Dreidimensionale übertragbar (siehe Abbildung 4.6): Um den Aufwand für die Definition von dreidimensionalen Zonen und Achsen gering zu halten, erfolgt deren Erstellung in räumlich beliebig anordbaren Ebenen und somit wie bisher üblich im 2D. Durch Extrusion entlang der Normalen der verwendeten Ebene werden aus zweidimensionalen Zonen und Achsen dreidimensionale Körper bzw. Flächen, die die Grundlage für räumliche Anfragen bilden.
In [Winstanley u. Hoshi 1992], [Akbas u. Fischer 2002], [Akbas 2003] und [Heesom 2006] werden ähnliche Konzepte zur räumlichen Gliederung eines 3D Modells mittels Zonen beschrieben. Entsprechende Algorithmen zur Anpassung der vorhandenen Objektgranularität des Modells werden jedoch nicht vorgestellt. Die Adressierung von Objekten bzw. Objektteilen anhand von Achsrastern sowie die Einbettung räumlicher Konzepte in eine Anfragesprache werden nicht betrachtet.

4.4 Modellbildung

Im Rahmen der Planung ist es notwendig, die Komplexität der Realwelt durch Modellbildung auf ein handhabbares Maß zu reduzieren. D.h., es können nur die für die Planung eines Bauwerks und dessen Erstellung wesentlichen Merkmale Berücksichtigung finden. Diese Modellbildung ist Grundlage für die spätere Umsetzung in Software. Im Bereich der computergestützten Planung hat sich hierfür der objektorientierte Modellierungsansatz durchgesetzt. Dabei werden Gegenstände, Personen und Prozesse der Realwelt im Computer durch Objekte und die realen Eigenschaftsmerkmale durch Objektattribute abgebildet. Die Gesamtmenge der betrachteten Objekte bildet das Modell.

Objekt: Ein Objekt O ist im mathematischen Sinne eine Menge von Attributen. Es stellt eine modellhafte Abbildung eines Elements der Realität dar. Der Zustand eines Objektes ist durch seine Attribute und deren Werte beschrieben.

$$O := \{a \mid a \text{ ist ein Attribut}\} \tag{4.1}$$

Attribut: Ein einzelnes Attribut a ist ein geordnetes Paar aus Attributnamen a_n und Attributwert a_w. Es beschreibt eine Eigenschaft eines Objektes. Der Attributname eines Attributes ist innerhalb eines Objektes eindeutig. Die eigentliche Information wird im Attributwert hinterlegt.

$$\begin{aligned} a := (a_n, a_w) \text{ mit } & a_n \text{ ein im Objekt eindeutiger Name} \\ & \text{und } a_w \in \text{Wertemenge} \end{aligned} \tag{4.2}$$

Jedem Attribut ist eine Wertemenge zugeordnet, die die Menge der möglichen Attributwerte festlegt. Häufig verwendete Wertemengen sind:

- Text $\mathbb{T}$
- Menge der ganzen Zahlen $\mathbb{Z} = \{\ldots, -2, -1, 0, 1, 2, \ldots\}$
- Menge der rationalen Zahlen $\mathbb{Q} = \left\{ \frac{p}{q} \mid p, q \in \mathbb{Z} \wedge q \neq 0 \right\}$
- Wahrheitswerte $\mathbb{W} = \{wahr, falsch\}$
- Datumsangaben $\mathbb{D}$ z.B. in der Schreibweise dd.mm.yyyy

Attributgruppe: Mitunter ist es zweckmäßig, gleichartige Informationen eines Objektes namentlich zusammenzufassen. Das ist insbesondere dann der Fall, wenn es sich um eine große Anzahl gleichartiger Informationen handelt oder deren Anzahl nicht von vornherein festgelegt ist bzw. sich während der Bearbeitung ändert.
Diese Informationen lassen sich durch n einzelne Attribute a_i abbilden, wobei n der Anzahl der Attributwerte a_{w_i} entspricht. Um den Gruppencharakter zu signalisieren, kann ein vollständig oder teilweise einheitlicher[19] Attributname a_{n_i} für die n Attribute

[19] In einer Datenbankumgebung mit dem in Kapitel 5.2.2 auf Seite 119 beschriebenen Datenbankschema ist eine Umsetzung dieses Konzeptes direkt möglich.
Innerhalb von Programmiersprachen hingegen ist eine Zuordnung von mehreren Attributen mit identischem Attributnamen zu einem Objekt nicht möglich. Anstatt dessen müssen, je nach Art der Gruppe, den mathematischen Begriffen *Menge* und *Folge* äquivalente Konstrukte verwendet werden, um mehrere Attributwerte unter einem Attributnamen verwalten zu können.

gewählt werden.
Prinzipiell werden zwei Fälle unterschieden:

- **ungeordnete Attributgruppe:** Hierbei handelt es sich im mathematischen Sinne um eine Menge. Die Reihenfolge der enthaltenen Attributwerte ist beliebig. Es wird ein einheitlicher Attributname für alle der Gruppe zugehörigen Attribute verwendet.

$$a_i := (a_n, a_{w_i}) \text{ mit } a_n \text{ als Attributname der Gruppe und } a_{w_i} \in \text{Wertemenge} \tag{4.3}$$

- **geordnete Attributgruppe:** Das mathematische Äquivalent ist eine Folge. Die Reihenfolge der zugehörigen Attributwerte ist festgelegt und von Bedeutung. Die Attributnamen der Gruppe angehörender Attribute werden aus einem einheitlichen Namen und einem durchzählenden Suffix zusammengesetzt. Das Suffix ermöglicht die Auswertung der Reihenfolge der Attribute.

$$a_i := (a_{n_i}, a_{w_i}) \text{ mit } a_{n_i} \text{ Attributname } a_n \text{ der Gruppe und Suffix } s_i \text{ und } a_{w_i} \in \text{Wertemenge} \tag{4.4}$$

Identifikator: Als notwendige Voraussetzung, um Änderungen im Bestand der Planungsdaten verfolgen zu können, muss jedes Objekt innerhalb eines Projektes eindeutig und persistent identifiziert sein. Hierzu wird für alle Objekte ein Attribut mit dem Attributnamen „ID“ und einem im Projekt eindeutigen Wert w_{id} als Attributwert verwendet (vgl. [Hanff 2003]). w_{id} ist der Identifikator des Objektes.

$$\bigwedge_{O} \bigvee_{a_i \in O} a_{n_i} = \text{„ID“} \wedge a_{w_i} = w_{id} \tag{4.5}$$

$$\bigwedge_{O_i, O_j} w_{id_i} \neq w_{id_j} \qquad \text{für } i \neq j \tag{4.6}$$

Klasse: In der Softwaretechnik werden gleichartige Objekte in Klassen zusammengefasst. Dabei ist eine Klasse eine feste Schablone für die Erstellung von Objektinstanzen. Die Zugehörigkeit eines Objektes zu einer Klasse wird durch den Objekttyp signalisiert. Dieser Ansatz stellt sicher, dass allen Objekten einer Klasse die gleichen Attribute zugeordnet sind. Obwohl dieses für stets vorhandene Informationen erstrebenswert ist, erweist sich dieser Ansatz im Rahmen der Planung von Bauwerken oft als sehr unflexibel.
Zu Beginn der Planung werden Objekte in der Regel entsprechend des groben Informationsstandes zunächst mit wenigen Attributen modelliert. Mit fortschreitendem Reifegrad der Planung werden dann, je nach Bedarf, zusätzliche Attribute zu den Objekten hinzugefügt. Dabei ist es oft nicht sinnvoll, die zusätzlichen Attribute allen Objekten einer Klasse zuzuordnen, sondern nur den Objekten, die zusätzliche Informationen tragen

sollen. Der übliche Aufbau von Objekten in der Softwaretechnik und deren Zusammenfassung zu Klassen lässt dieses jedoch nicht zu[20]. Daher kennzeichnet eine Klasse $\mathfrak{K}$ im Rahmen dieser Modellbildung lediglich eine Menge von Objekten und dient ausschließlich der Erleichterung der Interpretation von Informationen. Die Zugehörigkeit von Objekten zu einer Klasse wird durch ein Attribut mit dem Attributnamen „Objekttyp“ und dem Klassennamen k_n als Attributwert gekennzeichnet. Wie üblich, werden Objekte genau[21] einer Klasse zugeordnet. Unterschiedliche Objekte einer Klasse können jedoch verschiedene Attribute enthalten.

$$\bigwedge_{O} \bigvee_{a_i \in O} a_{n_i} = \text{„Objekttyp“} \tag{4.7}$$

$$\mathfrak{K} := \{O \mid \bigvee_{a_i \in O} a_{n_i} = \text{„Objekttyp“} \wedge a_{w_i} = k_n\} \tag{4.8}$$

$$\mathfrak{K}_i \cap \mathfrak{K}_j = \emptyset \qquad \text{für } i \neq j \tag{4.9}$$

Referenz: Neben den Objekten selbst sind für die Planung auch die Beziehungen zwischen Objekten von Bedeutung. Bei der Modellbildung können diese Beziehungen prinzipiell auf zwei verschiedene Arten abgebildet werden, die beliebig kombiniert werden können, um die jeweiligen Vorteile zu nutzen:

- **implizite Referenz:** Eine Beziehung zwischen zwei Objekten wird in den Objekten selbst abgebildet. Hierzu wird im Objekt ein Attribut verwendet, in dem der Identifikator des referenzierten Objektes bzw. zur Laufzeit dessen aktuelle Adresse im Speicher hinterlegt ist. Dadurch entsteht eine gerichtete Beziehung, über die von einem Objekt direkt auf das andere zugegriffen werden kann. Als Erweiterung kann im referenzierten Objekt eine entsprechende Rückreferenz vermerkt werden, um einen wechselseitigen Zugriff zu ermöglichen. Steht ein Objekt zu mehreren Objekten in gleichartiger Beziehung, so kann entsprechend eine Attributgruppe für die Abbildung der 1:n Beziehung verwendet werden.

- **objektivierte Referenz:** Hierbei werden Beziehungen zwischen Objekten als eigenständige Objekte abgebildet. Diese ist z.B. sinnvoll, wenn Attribute für die Referenz angegeben werden sollen oder wenn Beziehungen nachträglich in ein Modell eingefügt werden. Referenzobjekte verweisen über Attribute auf die eigentlich in Beziehung stehenden Objekte. Durch die Verwendung von Attributgruppen in Referenzobjekten können Mehrfachbeziehungen realisiert werden. Zusätzlich ist die Abbildung von Eigenschaften der Beziehungen selbst durch weitere Attribute der Referenzobjekte möglich. Auch wird durch die Auslagerung eine direkte Analyse der Objektbeziehungen ermöglicht. Der direkte Zugriff von einem Objekt auf das andere ist jedoch nicht möglich. Um in einem Objekt zu vermerken, dass es

[20] In einer Programmiersprache kann das Hinzufügen und Löschen von Attributen abweichend vom üblichen Konzept durch die Verwaltung der Objektattribute in einer Art erweiterbarem Wörterbuch ermöglicht werden, wobei der Attributname zum Nachschlagen verwendet wird.
In einer Datenbank ist das Hinzufügen und Löschen von Attributen mit einem entsprechendem Datenbankschema ebenfalls möglich.

[21] Vererbung ist für die hier vorgenommene Modellbildung ohne Relevanz.

in Beziehung zu einem anderen Objekt steht, kann dort eine inverse Referenz auf das entsprechende Referenzobjekt hinterlegt werden.

komplexe Eigenschaft: Komplexe Eigenschaften eines Objektes wie z.B. die Geometrie können nicht durch ein einfaches Attribut abgebildet werden. Stattdessen wird der erhöhte Informationsgehalt durch eine Menge von zueinander in Beziehung stehenden Objekten abgebildet. Zwischen einem dieser Objekte und dem Objekt, dessen Eigenschaft beschrieben wird, besteht eine Referenz, um die Zugehörigkeit der Information abzubilden.

Modell: Das Modell M ist eine mathematische Menge und umfasst alle für die Planung relevanten Objekte eines Projektes. Es beschreibt den betrachteten Ausschnitt der Wirklichkeit und ist in der Regel aus mehreren Teilmodellen zusammengesetzt.

$$M := \{O \mid O \text{ ist Gegenstand der Planung}\} \tag{4.10}$$

Teilmodell: Ein Teilmodell Θ_j ist eine Teilmenge des Modells. Der Durchschnitt verschiedener Teilmodelle ist die leere Menge, d.h., Teilmodelle sind paarweise disjunkt. Die Vereinigung aller Teilmodelle ergibt das Modell. D.h., ein Objekt ist stets Teil genau eines Teilmodells. Die Teilmodelle bilden eine Zerlegung des Modells. Zur Kennzeichnung eines Teilmodells wird ein eindeutiger Identifikator verwendet.

$$\Theta_j \subseteq M \tag{4.11}$$

$$\Theta_i \cap \Theta_j = \emptyset \qquad \text{für } i \neq j \tag{4.12}$$

$$\bigcup_j \Theta_j = M \tag{4.13}$$

Domain: Eine Domain ist ein abgegrenzter Verantwortungsbereich für einen Teil der Planung. In ihr arbeiten Fachleute mit einem speziellen Fokus und spezialisierten Anwendungen. Innerhalb der Domain können mehrere Bearbeiter mit abgestuften Datenbearbeitungsrechten existieren.

Domainmodell und Domaindaten: Ein Domainmodell D ist eine Menge von Teilmodellen, von denen innerhalb der Domain Objekte erstellt, geändert oder gelöscht werden. Die Domaindaten $\hat{D}$ sind hingegen die Menge von Attributen die in der Domain erstellt, geändert oder gelöscht werden. Dabei können die Domaindaten alle oder auch nur einzelne Attribute eines Objektes enthalten. D.h., der Begriff Domaindaten bezieht sich auf die Objekt- bzw. Attributebene, während der Begriff Domainmodell sich auf die Teilmodellebene bezieht (siehe hierzu Abbildung 4.7).
Bei beiden handelt es sich um eine mathematische Menge, doch im Gegensatz zu Domainmodellen sind Domaindaten disjunkt, d.h., ihr Durchschnitt ist die leere Menge. Die Vereinigung aller Domaindaten bzw. Domainmodelle ergibt das Modell. Die Einteilung in Domaindaten bildet somit eine Zerlegung des Modells und damit die Grundlage für die Aufteilung auf verschiedene Bearbeiter.
Ein Bearbeiter einer Domain hat stets Datenhoheit über die Domaindaten. D.h., er darf sie uneingeschränkt erzeugen, ändern oder löschen.

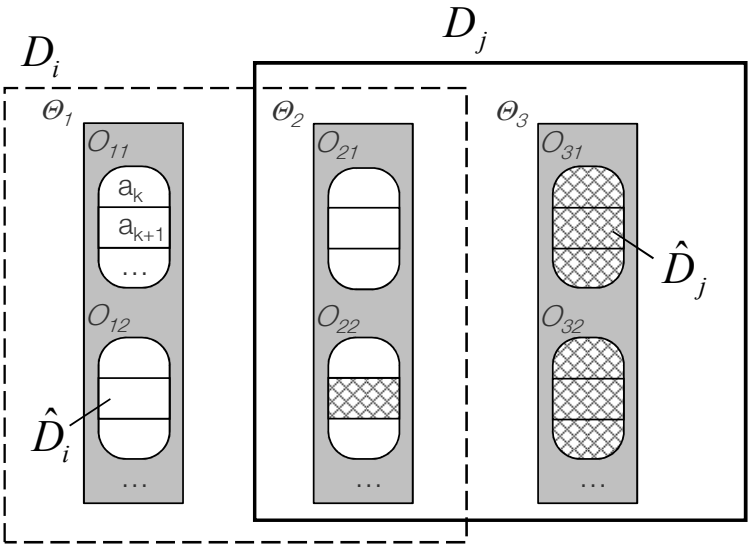


Abbildung 4.7: Domaindaten $\hat{D}$ versus Domainmodelle D
(Θ = Teilmodell, O = Objekt)

$$\hat{D}_i := \{a \mid a \text{ wird innerhalb der Domain i bearbeitet}\} \tag{4.14}$$

$$\hat{D}_i \cap \hat{D}_j = \emptyset \qquad \text{für } i \neq j \tag{4.15}$$

$$D_i := \{\Theta \mid \bigvee_{O \in \Theta} \bigvee_{a \in O} a \in \hat{D}_i\} \tag{4.16}$$

$$\bigcup_i D_i = M \tag{4.17}$$

Metadaten: Metadaten sind beschreibende Zusatzinformationen, die nicht direkt dem Zweck der Planung dienen. Insbesondere aus organisatorischen Zwecken ist es jedoch oft sinnvoll, solche Daten mitzuverwalten. Informationen über die Versionshistorie von Objekten werden als Metadaten angesehen.

Dokumente: Ein Dokument ist ein Container zur Speicherung von Informationen. Je nach dessen Inhalt kann man strukturierte und unstrukturierte Dokumente unterscheiden. Nur strukturierte Dokumente können von Computersoftware sinnvoll interpretiert werden. Daher sind sie die Grundlage für den dateibasierten Austausch von Modelldaten zwischen verschiedenen Planungsbeteiligten und deren Computeranwendungen. Dennoch existieren viele für die Planung relevante Informationen, die nur in unstrukturierten Dokumenten verwaltet und von Menschen interpretiert werden. Um den Zugriff auf diese Informationen zu vereinfachen, ist es mitunter sinnvoll, Referenzen von Objekten des Modells auf unstrukturierte Dokumente (z.B. ein beschreibendes Textdokument) zu verwalten. Das geschieht analog zu Objektreferenzen, nur dass statt der ID des referenzierten Objektes der Pfad zum Speicherort des referenzierten Dokumentes verwaltet wird.

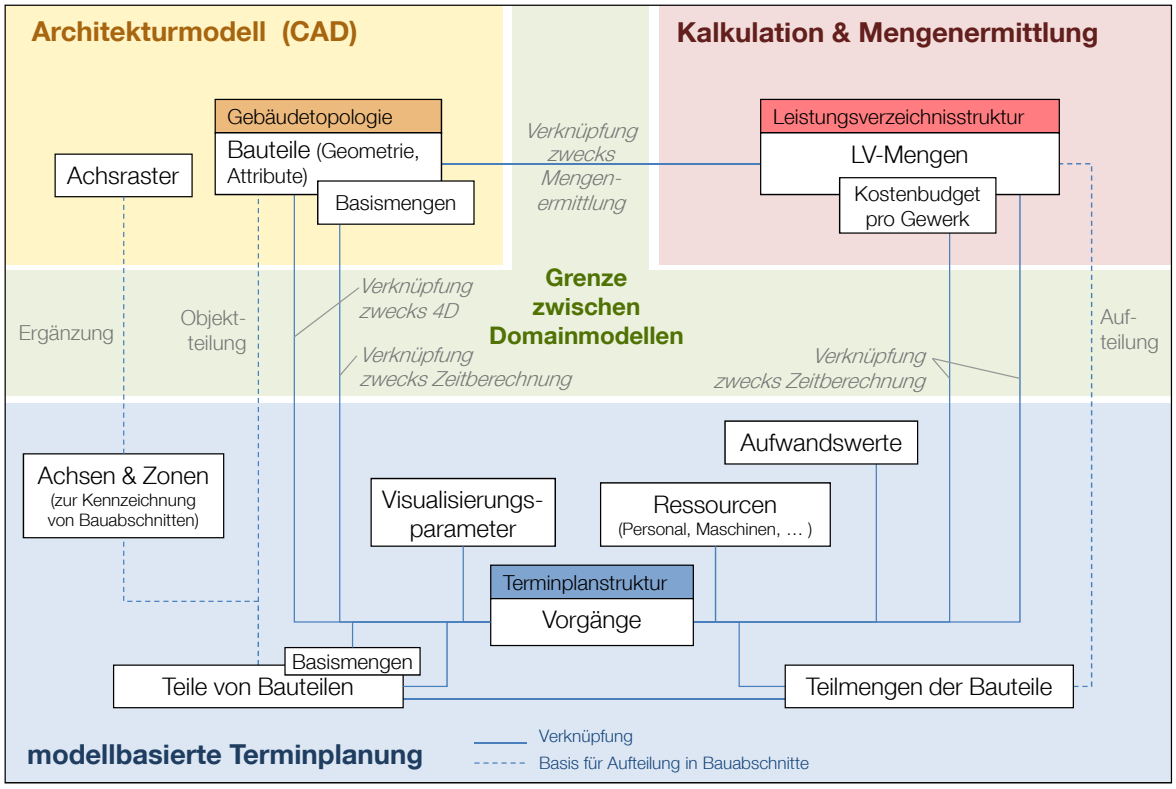


Abbildung 4.8: Daten und Verknüpfungen der drei für die Terminplanung relevanten Domainmodelle

4.5 Daten

4.5.1 Domainmodelle und Bearbeitungsmodi

Im Rahmen des Terminplanungsprozesses werden Daten aus den drei Fachdisziplinen Architektur, Kalkulation und Terminplanung verarbeitet. Traditionell liegen diese in eigenständigen, voneinander unabhängigen Datenmodellen vor. Eine Verknüpfung existiert nicht explizit, sondern nur mental durch entsprechende Gliederungen und textuelle Beschreibungen.
Im Rahmen der modellbasierten Zusammenarbeit wird hier von Domainmodellen innerhalb des Gesamtbauwerksmodells gesprochen. Zwar weist jedes dieser Domainmodelle nach wie vor eine eigene, der fachspezifischen Sichtweise entsprechende, hierarchische Ordnungsstruktur auf, jedoch sind die Einzeldaten fachübergreifend miteinander verknüpft (Abbildung 4.8). Die Erstellung der Teilmodelle und ihrer Verknüpfungen erfolgt entsprechend der in Kapitel 4.2 auf Seite 46 beschriebenen Prozessreihenfolge.
Im Unterschied zur herkömmlichen Modellnutzung im Rahmen von 4D-Simulationen erfolgt im vorgestellten Ansatz eine Einbeziehung des Domainmodells der Kalkulation sowie die Teilung von Bauteilgeometrien und zugehörigen Mengeninformationen auf Basis einer räumlichen Adressierung mit Zonen und Achsen.
Die im Bereich der Terminplanung verarbeiteten Daten können wie folgt klassifiziert werden (Abbildung 4.9 auf der nächsten Seite):

(A) Basisdaten aus fremden Domainmodellen. Sie werden von anderen Fachdiszipli-

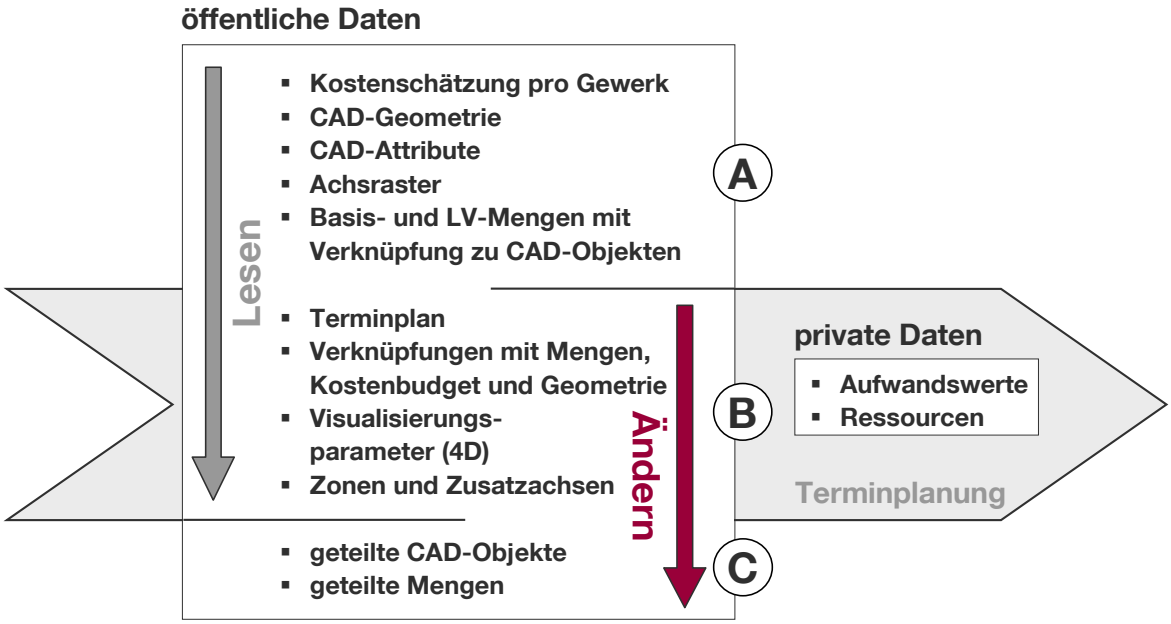


Abbildung 4.9: Klassifizierung zu verarbeitender Daten

nen erzeugt und für die Zusammenarbeit veröffentlicht. Im Rahmen der Terminplanung wird auf diese Daten nur lesend zugegriffen. Eine Änderung dieser Daten erfolgt nicht.

(B) Domaindaten der Terminplanung, bestehend aus privaten und öffentlichen Daten. Sie werden vom Terminplaner erzeugt und im Rahmen der Planung von ihm oder anderen an der Terminplanung Beteiligten geändert bzw. ergänzt.

(C) Abgeleitete Objekte zur Abbildung von Teilinformationen fremder Domainmodelle. Sind CAD-Objekte und Mengen durch die Einteilung des Modells in Bauabschnitte zu teilen, so werden hierfür neue, mittels Regeln abgeleitete Objekte erzeugt. Die Orginalobjekte bleiben unverändert erhalten, um Interferenzen mit anderen Fachdisziplinen zu vermeiden. Ergeben sich Änderungen in den Orginalobjekten, so sind die abgeleiteten Objekte zu aktualisieren.

4.5.2 Gliederung in Teilmodelle

Die in Abbildung 4.8 dargestellten Informationen werden aufgrund ihrer thematischen Trennung jeweils als eigenständige Teilmodelle abgebildet. Die wesentlichen Teilmodelle, Objekte und Attribute werden nachfolgend vorgestellt (siehe auch Tabelle 4.1 auf Seite 65).

Terminplan

Das Teilmodell, das die Terminplandaten enthält, wird mit T bezeichnet. Es enthält als wesentliche Objekte die Terminplanvorgänge t.

$$T := \{t \mid t \text{ ist ein Vorgang im Terminplan}\} \tag{4.18}$$

Attribute von Terminplanvorgängen sind die ID, eine Gliederungsnummer, ein Vorgangsname bzw. eine Beschreibung, ein Start- und Endtermin sowie die Vorgangsdauer. Die Gliederungsnummer bestimmt eindeutig die Position des Terminplanvorgangs innerhalb des hierarchisch aufgebauten Terminplans und ermöglicht so die Zuordnung von Vorgängen zu Summenvorgängen. Vorgänger- und Nachfolgerbeziehungen zwischen Terminplanvorgängen werden mittels zwei entsprechenden Attributen des Terminplanvorgangs abgebildet (implizite Referenzen). Weiterhin werden Terminplanvorgänge meistens zwecks Filterung mit zusätzlichen, benutzerdefinierten Attributen versehen.

Kosten

Die Kostenschätzung erfolgt zu einem frühen Projektzeitpunkt weitestgehend unabhängig von anderen Planungsdaten auf Basis einer Bauwerksbeschreibung und Erfahrungswerten. Detaillierte Mengeninformationen liegen zu diesem Zeitpunkt noch nicht vor. Die Daten der Kostenschätzung werden daher als eigenständiges Teilmodell K innerhalb des Domainmodells der Kalkulation betrachtet. Die darin ausgewiesenen Gesamterstellungskosten eines Bauprojektes untergliedern sich hierarchisch in einzelne Kostenbeiträge der Gewerke. Alle Kostenwerte werden als eigenständige Objekte k abgebildet.

$$K := \{k \mid k \text{ ist ein Kostenwert}\} \tag{4.19}$$

Attribute von Kostenwertobjekten sind die ID, eine Gliederungsnummer, der Name des betreffenden Gewerks, ein beschreibender Text sowie der Kostenwert und die zugehörige Währung.

CAD-Modell

Als CAD-Objekte werden Objekte des Architekturmodells bezeichnet, die physische Objekte (Bauteile oder Elemente der Außenanlagen) des Bauwerks beschreiben. Zusätzlich werden Objekte, die einen Raum beschreiben, als CAD-Objekte bezeichnet. Die Menge aller CAD-Objekte wird mit C bezeichnet.

$$C := \{c \mid c \text{ ist ein Raum oder physisches Objekt des Bauwerks}\} \tag{4.20}$$

Neben der ID sind die bedeutendsten Attribute von CAD-Objekten deren Typ, Position und Geometrie. Zur Einordnung in die räumliche Gliederung eines Projektes werden zudem die Attribute Bauwerk und Bauwerksgeschoss verwendet. Zur näheren Beschreibung der CAD-Objekte werden das Material und weitere benutzerdefinierte Attribute verwendet (z.B. die Feuerschutzklasse einer Tür).

Mengeninformationen

Zu unterscheiden sind Basis- und LV-Mengen. Basismengen q_B sind Teil des Architekturmodells und werden im Teilmodell Q_B zusammengefasst. Sie werden automatisch, rein auf Basis der Geometrie von der CAD-Anwendung berechnet. LV-Mengen q_{LV} werden im Teilmodell Q_{LV} zusammengefasst und sind Teil des Domainmodells der

Kalkulation.

$$Q_B := \{q_B \mid q_B \text{ ist eine Basismenge}\} \tag{4.21}$$
$$Q_{LV} := \{q_{LV} \mid q_{LV} \text{ ist ein einzelner Mengenwert im Leistungsverzeichnis}\} \tag{4.22}$$

Basismengen q_B erhalten die Attribute ID, Bezeichnung, Wert und Einheit sowie die ID des zugehörigen CAD-Objektes (implizite Referenz). Die Bezeichnung beschreibt die Bedeutung des Mengenwertes (z.B. Gesamtvolumen).
Die Mengeninformationen im Leistungsverzeichnis unterscheiden sich von Basismengen im Architekturmodell in den folgenden Punkten:

- Berechnung der Mengen unter Beachtung von nationalen Berechnungsregeln (z.B. in Deutschland der VOB),
- Berücksichtigung zusätzlicher, mittels vom Benutzer definierten Formeln abgeleitete Mengen für im Architekturmodell nicht explizit konstruierter Objekte (z.B. Bewehrungseisen, Schichten eines Deckenaufbaus),
- den Mengenwerten zugeordnete Beschreibungen von auszuführenden Arbeiten (relevant für die Festlegung der Dauer von Vorgängen),
- eine thematische Gliederung nach Gewerken bzw. anfallenden Arbeiten.

LV-Mengenwerte q_{LV} sind Einzelbeiträge zur Gesamtmenge einer LV-Position. Sie erhalten die Attribute ID, Positionsnummer, Wert und Einheit sowie die ID des CAD-Objektes, aus dem der Mengenwert abgeleitet ist (implizite Referenz). Die Positionsnummer bezeichnet den Eintrag innerhalb des hierarchisch strukturierten Leistungsverzeichnisses, dem der Mengenwert zugeordnet ist. Hierüber kann üblicherweise auf die weiteren Eigenschaften, die Bezeichnung und Beschreibung der LV-Position zugegriffen werden, wobei die Bezeichnung die Überschrift der LV-Position und die Beschreibung die detaillierten Informationen über die auszuführenden Tätigkeiten enthält. Zur Vereinfachung des Sachverhalts werden diese Informationen im Rahmen dieser Arbeit ebenfalls als Attribute der LV-Mengenwerte betrachtet.

Visualisierungsparameter

Für die Festlegungsart der Darstellung aktiver Vorgänge in einer 4D-Simulation ist je zu visualisierendem Terminplanungsvorgang ein Satz von Visualisierungsparametern, bestehend aus Vorgangstyp, Farbe und Transparenz vorzugeben. Der Vorgangstyp bestimmt, wann ein Objekt sichtbar geschaltet wird. Beispielsweise entspricht der Vorgangstyp „construct“ dem Errichten von Bauteilen. Die betreffenden CAD-Objekte sind in einer 4D-Simulation zunächst nicht sichtbar. Mit Beginn dessen Erstellung werden sie sichtbar geschaltet und gemäß der angegebenen Farbe eingefärbt. Nach vollendeter Bauteilerstellung verbleiben die Objekte sichtbar in der 4D-Simulation, erhalten jedoch die endgültige Farbe. Beim Vorgangstyp „demolish“, der für die Demontage bzw. den Abriss von Bauteilen verwendet wird, sind die Objekte hingegen am Anfang sichtbar und nach Vollendung des Vorgangs unsichtbar. Entsprechend können weitere Vorgangstypen definiert werden (siehe Abbildung C.10 im Anhang bzw. [Koschorrek u. a. 2008]).
Um die Eingabe im Projekt zu vereinfachen und um eine einheitliche Darstellung in

allen Projekten eines Unternehmens sicherzustellen, können Sätze von Visualisierungsparametern als Vorlagen abgespeichert werden. Ein solcher Satz wird als ein Objekt mit Attributen entsprechend den genannten Parametern abgebildet und als Visual-Objekt v bezeichnet. Jedes Visual-Objekt erhält als weitere Attribute eine ID und eine den zu visualisierenden Vorgängen entsprechende Beschreibung, z.B. „Betonieren von Wänden“. Die Vorlagen für die Visualisierungsparameter werden in einem eigenen Teilmodell V hinterlegt und können projekt- oder unternehmensweit bereitgestellt werden.

$$V := \{v \mid v \text{ ist ein Satz an Visualisierungsparametern}\} \tag{4.23}$$

Zonen und Achsen

Zur Kennzeichnung von Bauabschnitten werden Zonen und Achsen als Hilfsobjekte h verwendet. Sie werden im Teilmodell H zusammengefasst.

$$H := \{h \mid h \text{ ist ein Hilfsobjekt zur Kennzeichnung von Bauabschnitten}\} \tag{4.24}$$

Die wesentlichen Eigenschaften von Hilfsobjekten sind die ID, Bezeichung sowie deren Position und Geometrie. Zusätzlich werden Zonen und Achsen zur optischen Unterscheidung eine Farbe zugeordnet.

Weitere Teilmodelle

Mittels Hilfsobjekten werden aus dem Modell den jeweiligen Bauabschnitten entsprechende Teilinformationen abgeleitet. Das betrifft die Geometrie der CAD-Objekte und die zugehörigen Mengeninformationen (Basis- und LV-Mengen). Da es sich um aus dem Originalmodell abgeleitete Informationen handelt, brauchen diese eigentlich nicht explizit im Modell gespeichert zu werden. Da jedoch nicht alle derzeit existierenden Anwendungen die Funktionalität der regelbasierten Objektteilung bieten, ist es sinnvoll, die abgeleiteten Objekte dennoch in expliziter Form abzuspeichern bzw. auszutauschen. Hierzu werden den Teilmodellen C, Q_B und Q_{LV} zusätzliche CAD- und Mengenobjekte als Zerlegung der Originalobjekte hinzugefügt.
Ressourcen und Aufwandswerte liegen katalogisiert in firmeninternen Datenhaltungen vor. Ihre Attribute geben eine nähere Beschreibung des Einzelwertes bzw. bilden die Struktur des Katalogs ab. Da es sich um private Daten handelt, werden diese nicht näher betrachtet.

4.5.3 Beziehungen zwischen Teilmodellen

Während der Planung werden Verknüpfungen zwischen den beschriebenen Teilmodellen erstellt. Sie werden als Bindungsrelationen B_i bezeichnet und können als implizite oder objektivierte Referenzen abgebildet werden. Von besonderem Interesse für die Terminplanung sind die domainübergreifenden Bindungsrelationen unter Beteiligung des Terminplans. Diese werden nachfolgend beschrieben. Es sei darauf hingewiesen, dass im Terminplan stets Vorgänge existieren, die nicht mit Objekten des Bauwerksmodells verknüpft werden. D.h., die betrachteten Bindungsrelationen sind auf der Seite des Terminplans nicht total.
Relationen zwischen Objekten innerhalb von Teilmodellen sind im Rahmen dieser Arbeit von untergeordneter Bedeutung und werden daher nicht betrachtet.

Beziehung zwischen Terminplan und Kostenschätzung

Die binäre Bindungsrelation zwischen Vorgängen des Terminplans T und Objekten des Teilmodells der Kostenschätzung K wird mit B_1 bezeichnet (siehe Abbildung 4.10).

$$B_1 := \{(t,k) \in T \times K \mid tB_1k\} \text{ mit } t \in T,\ k \in K \tag{4.25}$$

B_1 ist rechtseindeutig, d.h., ein Terminplanvorgang ist maximal mit einem Kostenwert verknüpft. Ein Kostenwert kann jedoch Basis für die Berechnung der Dauer mehrerer Terminplanvorgänge sein. Nicht alle Kostenwerte sind für die Terminplanung relevant, somit bleiben einige unverknüpft.

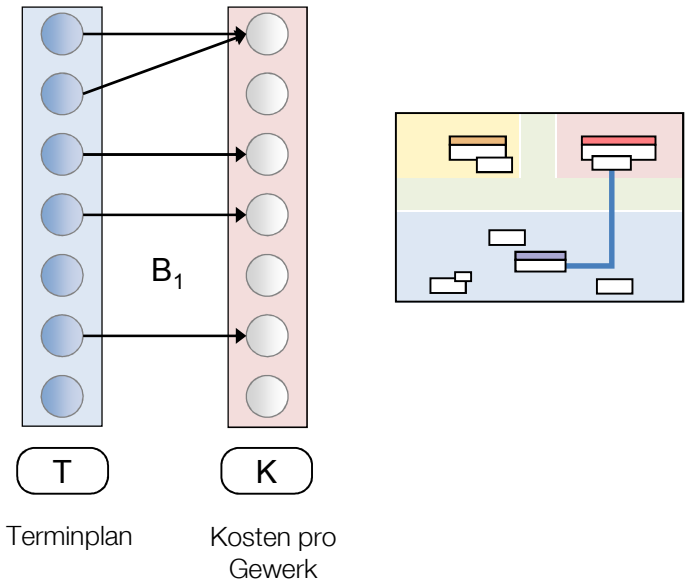


Abbildung 4.10: Binäre Bindungsrelation zwischen Terminplanvorgängen und Kostenschätzungen pro Gewerk

Anmerkung: Die Relation zwischen Terminplan und Aufwandswerten ist von gleicher Art wie B_1. Die zwischen Terminplan und Ressourcen ist ähnlich mit dem Unterschied, dass einem Terminplanvorgang mehrere Ressourcen zugeordnet sein können (nicht rechtseindeutig).

Beziehung zwischen Terminplan und CAD-Modell

Vorgänge des Terminplans und Objekte des CAD-Modells werden zwecks Erstellung einer 4D-Simulation miteinander verknüpft. Dieses kann direkt durch eine binäre Bindungsrelation B_2 erfolgen (Abbildung 4.11 auf der nächsten Seite).

$$B_2 := \{(t,c) \in T \times C \mid tB_2c\} \text{ mit } t \in T,\ c \in C \tag{4.26}$$

Dabei kann sich ein Terminplanvorgang auf mehrere CAD-Objekte beziehen und ein CAD-Objekt kann von mehreren Terminplanvorgängen referenziert sein. Es sind nicht notwendigerweise alle CAD-Objekte mit Terminplanvorgängen verknüpft. Dieses ist insbesondere nicht der Fall, wenn eine 4D-Simulation zwecks Kontrolle schon während der Terminplanerstellung eingesetzt wird.

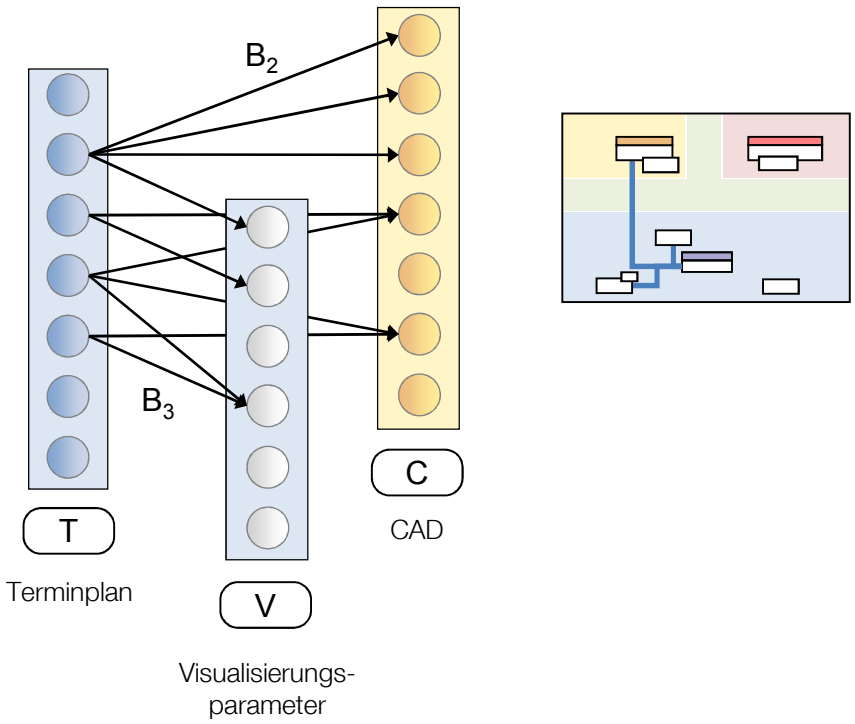


Abbildung 4.11: Binäre Bindungsrelationen zwischen CAD-Objekten, Terminplanvorgängen und Visualisierungsparametern

Um die Art der Visualisierung eines Vorgangs in einer 4D-Simulation näher zu spezifizieren, muss jedem mit CAD-Objekten verknüpften Terminplanvorgang zusätzlich ein Satz von Visualisierungsparametern, d.h. ein Visual-Objekt, zugeordnet werden. Dieses kann durch eine rechtseindeutige Bindungsrelation B_3 erfolgen.

$$B_3 := \{(t, v) \in T \times V \mid tB_3v\} \text{ mit } t \in T,\ v \in V \tag{4.27}$$

Die beiden Relationen B_2 und B_3 sind zeitgleich während der Terminplanung zu erstellen. Sie können daher alternativ durch eine dreistellige Bindungsrelation L ersetzt und durch eigenständige Objekte abgebildet werden (vgl. Abbildung 4.12).

$$L := \{(t, c, v) \in T \times C \times V \mid Ltcv\} \text{ mit } t \in T, c \in C,\ v \in V \tag{4.28}$$

Diese Modellierung bietet gegenüber der ersten den Vorteil, dass L als eigenständiges Teilmodell einer 4D-Simulation betrachtet werden kann. Sie wurde daher für die Umsetzung gewählt. Die Elemente von L werden als Link4D-Objekte bezeichnet. Sie weisen folgende Attribute auf: eine ID, jeweils eine Referenz auf einen Terminplanvorgang und ein Visual-Objekt, sowie Referenzen auf die zugeordneten CAD-Objekte.

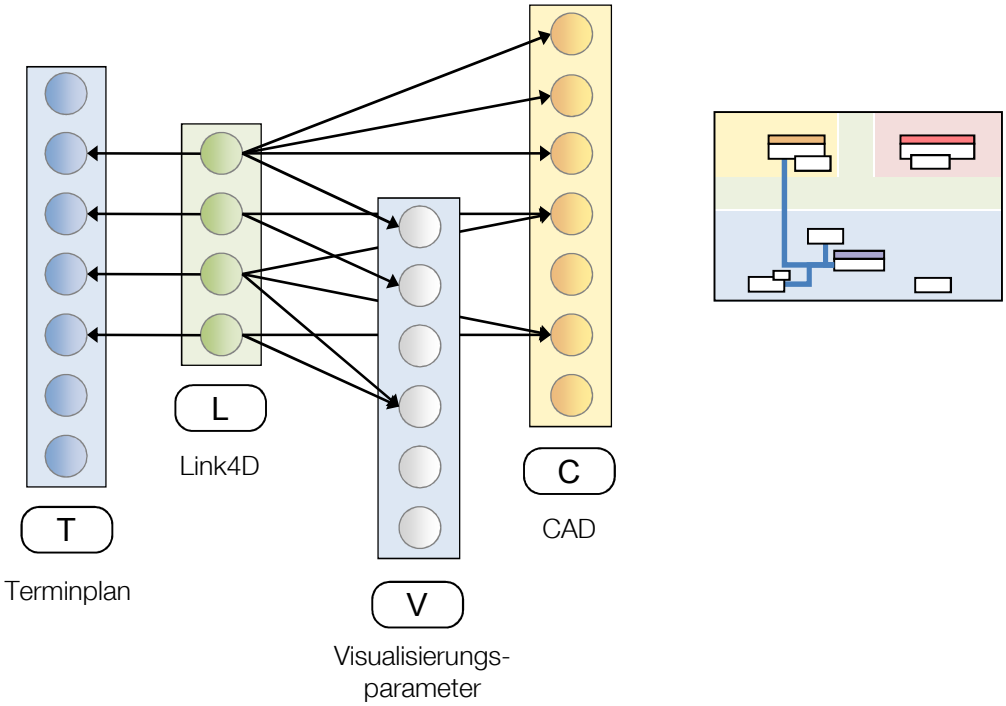


Abbildung 4.12: Durch die Einführung von Link4D-Objekten entsteht ein eigenständiges Teilmodell für die 4D-Simulation (dreistellige Relation L)

Beziehung zwischen Terminplan und Mengeninformationen

Basismengenwerte q_B werden automatisch vom CAD-Programm ermittelt und mit den zugehörigen CAD-Objekten verknüpft. LV-Mengen q_{LV} werden ebenfalls automatisch, jedoch mittels erweiterter Regeln aus dem CAD-Modell abgeleitet. *Automatisch* ermittelte Mengenwerte werden mit q_{i_a} bezeichnet und in der mathematischen Menge Q_{i_a} zusammengefasst.

$$Q_{i_a} := \{q_{i_a} \mid q_{i_a} \text{ ist ein automatisch ermittelter Mengenwert}\} \text{ mit } i \in \{B, LV\} \quad (4.29)$$

Zusätzlich zu den automatisch ermittelten Mengen kann es erforderlich sein, relevante Mengeninformationen manuell zu ergänzen. Das ist dann der Fall, wenn deren Ableitung aus dem Modell nicht oder nicht in der erforderlichen Präzision möglich ist. Diese manuell ermittelten Mengenwerte werden mit q_{i_m} bezeichnet und in der mathematischen Menge Q_{i_m} zusammengefasst.

$$Q_{i_m} := \{q_{i_m} \mid q_{i_m} \text{ ist ein manuell ermittelter Mengenwert}\} \text{ mit } i \in \{B, LV\} \quad (4.30)$$

Q_{i_a} und Q_{i_m} bilden eine Zerlegung von Basis- bzw. LV-Mengen:

$$Q_i = Q_{i_a} \cup Q_{i_m} \text{ mit } i \in \{B, LV\} \quad (4.31)$$

$$Q_{i_a} \cap Q_{i_m} = \emptyset \quad (4.32)$$

Automatisch ermittelte Mengen sind durch die Bindungsrelation B_4 mit CAD-Objekten verknüpft (Abbildung 4.13 auf der nächsten Seite).

$$B_4 := \{(q_{i_a}, c) \in Q_{i_a} \times C \mid q_{i_a} B_4 c\} \text{ mit } q_{i_a} \in Q_{i_a},\ c \in C,\ i \in \{B, LV\} \quad (4.33)$$

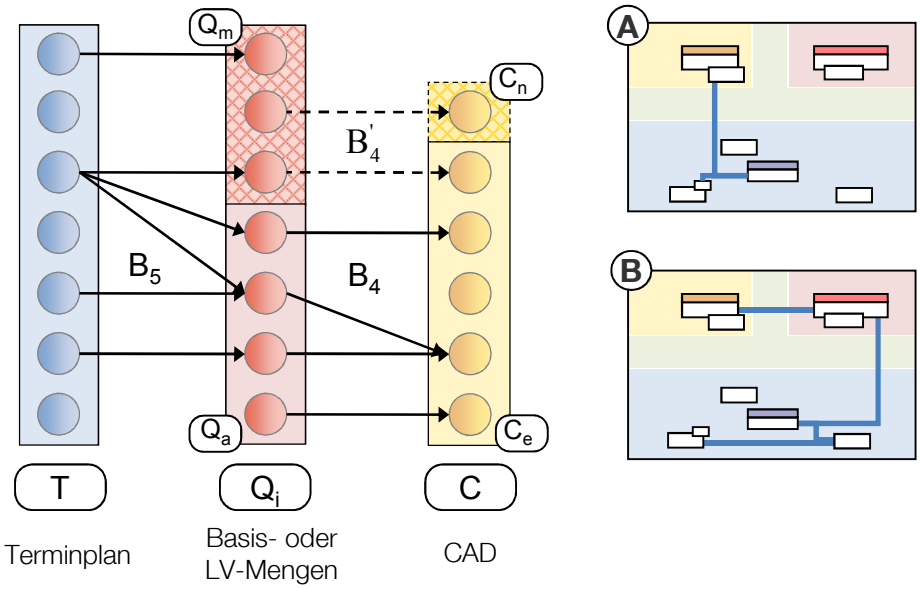


Abbildung 4.13: Relationen zwischen Terminplanvorgängen, Mengenwerten und CAD-Objekten; A: Basismengen; B: LV-Mengen

B_4 ist rechtseindeutig und linkstotal, da jeder automatisch ermittelte Mengenwert eindeutig mit dem entsprechenden CAD-Objekt verknüpft ist.

Manuell ermittelte Mengen sind hingegen zunächst nicht mit CAD-Objekten im Architekturmodell verknüpft. Um sie jedoch räumlich zuordnen zu können, ist es sinnvoll, sie manuell mit geometrischen Repräsentanten zu verknüpfen. Hierzu können bereits existierende ($c \in C_e$) oder vereinfachte, neu erstellte ($c \in C_n$) CAD-Objekte verwendet werden. Die manuell ergänzten Verknüpfungen werden in der Bindungsrelation B'_4 zusammengefasst.

$$B'_4 := \{(q_{i_m}, c) \in Q_{i_m} \times C \mid q_{i_m} B'_4 c\} \text{ mit } q_{i_m} \in Q_{i_m},\ c \in C,\ i \in \{B, LV\} \quad (4.34)$$

In der Regel existieren pro CAD-Objekt mehrere verknüpfte Mengenwerte gleicher oder unterschiedlicher Dimension. B_4 und B'_4 sind somit nicht linkseindeutig.

Die Verknüpfung zwischen Terminplanvorgängen und Mengenwerten wird als Bindungsrelation B_5 bezeichnet.

$$B_5 := \{(t, q_i) \in T \times Q_i \mid t B_5 q_i\} \text{ mit } t \in T,\ q_i \in Q,\ i \in \{B, LV\} \quad (4.35)$$

Zur Berechnung der Vorgangsdauer werden in der Regel mehrere Mengenwerte *gleicher* Dimension mit einem Terminplanvorgang verknüpft. Ein Mengenwert kann für die Ermittlung der Dauer verschiedener Vorgänge verwendet werden. B_5 ist somit weder rechts- noch linkseindeutig.

Die Bindungsrelationen B_4 und B'_4 werden bei der Erstellung des CAD-Modells bzw. während der modellbasierten Mengenermittlung erstellt. Die Bindungsrelation B_5 wird während der Terminplanung erstellt. Dabei kann der räumliche Bezug der Mengenwerte über die verknüpften CAD-Objekte genutzt werden. Bei manuell ermittelten Mengenwerten ist dieses nur möglich, wenn die Bindungsreleation B'_4 manuell ergänzt wurde. So ist z.B. beim obersten Mengenwert in Abbildung 4.13 keine räumliche Zuordnung vorhanden.

4.5.4 Zusammenfassung

In Tabelle 4.1 sind die wesentlichen Objekte und ihre Eigenschaften nochmals zusammengefasst. Bindungsrelationen wurden in der späteren Umsetzung als objektivierte Referenzen abgebildet.

Objekt	Eigenschaften
Terminplanvorgang	ID, Gliederungsnummer, Start- u. Endtermin, Dauer, ID Vorgänger, ID Nachfolger, benutzerdefinierte Attribute
Kostenwert	ID, Gliederungsnummer, Gewerk, Beschreibung, Wert, Währung
CAD-Objekt	ID, Objekttyp, Bauwerk, Geschoss, Position und Geometrie, Material, benutzerdefinierte Attribute
Basismenge	ID, Bezeichnung, Wert, Einheit, ID des CAD-Objektes
Mengenwert des LVs	ID, Positionsnummer, Wert, Einheit, ID des CAD-Objektes, Pos-Bezeichnung, Pos-Beschreibung
Visual-Objekt	ID, Beschreibung, Vorgangstyp, Farbe, Transparenz
Achsen und Zonen	ID, Bezeichnung, Position und Geometrie, Farbe
B_1-Objekt	ID, ID eines Terminplanvorgangs, ID eines Kostenwertes
Link4D-Objekt	ID, ID eines Terminplanvorgangs, IDs von CAD-Objekten, ID eines Visual-Objekts
B_4- bzw. B_4'-Objekt	ID, ID eines Mengenwertes, ID eines CAD-Objektes
B_5-Objekt	ID, ID eines Terminplanvorgangs, IDs von Mengenwerten

Tabelle 4.1: Relevante Objekte und Eigenschaften

4.6 Verknüpfungssprache

Um die Erstellung und Aktualisierung von Bindungsrelationen effizient zu gestalten, soll das Verknüpfen regelbasiert erfolgen. D.h., die Verknüpfungen werden nicht einzeln vom Anwender erstellt, sondern die Logik der Verknüpfungserstellung wird pro Terminplanvorgang in durch den Computer interpretierbarer Form hinterlegt. Dabei erfolgt jede Verknüpfung eines Terminplanvorgangs optional und aus folgenden Gründen:

1. zur Gewinnung von Basisinformationen für die Berechnung der Vorgangsdauer (Methode A,B: Bindungsrelation B_5, Methode C: Bindungsrelation B_1),

2. zur Erstellung einer 4D-Simulation (Relation L bzw. alternativ die Bindungsrelationen B_2 und B_3).

Vom Anwender sind pro Terminplanvorgang, je nach Anzahl der zu verknüpfenden Teilmodelle, mehrere Verknüpfungsregeln anzugeben. Hierfür wird nachfolgend als Werkzeug eine Sprache spezifiziert. Die vollständige Sprachdefinition ist in Anhang A in

erweiterter Backus-Naur-Form, wie sie vom Compilergenerator JavaCC[22] verwendet wird, angegeben. Eine schematische Übersicht über die Sprachkonstrukte und deren syntaktischen Zusammenhang zeigt Abbildung 4.15 auf Seite 71.
Spezieller Anwendungsbereich der Sprache ist im Rahmen der vorliegenden Arbeit die modellbasierte Terminplanung. Die Sprache ist jedoch so konstruiert, dass sie auch auf andere Problemstellungen im Zusammenhang mit der Weiterverarbeitung von Teilinformationen eines Bauwerksinformationsmodells übertragen und mit dafür erforderlichen Funktionalitäten erweitert werden kann.

4.6.1 Prinzip

Eine Verknüpfungsregel spezifiziert die Zielmenge Z, deren Objekte mit einem Terminplanvorgang t durch eine Relation R_1 zu verknüpfen sind.

$$Z := \{z \mid z \text{ ist ein zu verknüpfendes Objekt}\} \tag{4.36}$$

$$Z \subseteq K \vee Z \subseteq C \vee Z \subseteq Q_B \vee Z \subseteq Q_{LV} \tag{4.37}$$

Für die Bestimmung der Objekte von Z werden in der Verknüpfungsregel Filter definiert. f_j ist ein Filter auf einem beliebigen Teilmodell Θ_j. Er definiert Restriktionen für Objekte. Werden alle Restriktionen von einem Objekt des Teilmodells Θ_j eingehalten, so genügt das Objekt dem Filter. Alle dem Filter genügende Objekte bilden die Auswahlmenge A_j im Teilmodell Θ_j.

$$A_j := \{a \in \Theta_j \mid a \text{ genügt dem Filter } f_j\} \tag{4.38}$$

Wird nur für ein Teilmodell ein Filter definiert, so entspricht A_1 der Zielmenge Z.

Um eine differenziertere Auswahl der Objekte in der Zielmenge treffen zu können, kann es erforderlich sein, durch gekettete Relationen R_2, R_3, ... mit Objekten in A_1 in Beziehung stehende Informationen anderer Teilmodelle Θ_2, Θ_3, ... in das Auswahlkriterium mit einzubeziehen. Die Nutzung von Filtern in mehreren miteinander verknüpften Teilmodellen ermöglicht z.B. die räumliche Auswahl von Mengenwerten. Dieses wäre bei der isolierten Betrachtung des Teilmodells der LV-Mengenwerte nicht möglich. Zu diesem Zweck werden auch auf weiteren Teilmodellen Filter definiert. Somit entstehen die zusätzlichen Auswahlmengen A_2, A_3, ... (siehe Abbildung 4.14 auf der nächsten Seite). Es werden schließlich nur diejenigen Objekte des zu verknüpfenden Teilmodells Θ_1 in die Zielmenge Z aufgenommen, für die eine gekettete Relation besteht, die sich über alle Auswahlmengen erstreckt:

$$Z := \{a \in A_1 \mid \bigvee_{b \in A_2} \bigvee_{c \in A_3} \bigvee_{\dots} (aR_2b \wedge bR_3c \wedge \dots)\} \tag{4.39}$$

Beispiel: Ein Terminplanvorgang t, der das Betonieren von Wänden im dritten Geschoss eines Bauwerks beschreibt, soll mit den entsprechenden Mengenwerten des Leistungsverzeichnisses verknüpft werden, die das einzubauende Betonvolumen beschreiben. Die Relationen B_4 und B_4' sind aus dem vorherigen Planungsprozess bereits im

[22] siehe https://javacc.dev.java.net und [Copeland 2007]

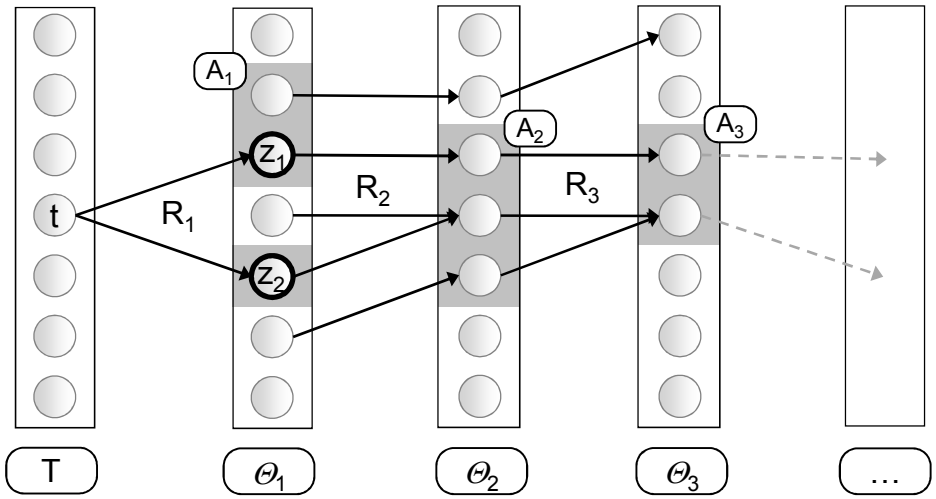


Abbildung 4.14: Prinzip der teilmodellübergreifenden Objektfilterung zwecks Bestimmung der Zielmenge Z der mit dem Terminplanvorgang t zu verknüpfenden Objekte

Datenmodell als objektivierte Referenzen in einem eigenen Teilmodell hinterlegt:

$\Theta_1 \rightarrow$ Teilmodell Q_{LV}, das die Mengenwerte des Leistungsverzeichnisses enthält
$\Theta_2 \rightarrow$ Teilmodell, das die objektivierten Referenzen von B_4 und B_4' enthält
$\Theta_3 \rightarrow$ Teilmodell C, das die CAD-Objekte enthält

$A_1 \rightarrow$ Mengenwerte, die von Objekten in A_2 referenziert werden und die Attributbelegung Einheit=m^3 aufweisen
$A_2 \rightarrow$ Relationsobjekte, die Objekte in A_3 referenzieren
$A_3 \rightarrow$ CAD-Objekte mit der Attributbelegung Objekttyp=Wand, Material=Beton und Geschoss=3.OG

$$Z = \{z_1, z_2\}$$

D.h. die Mengenwerte z_1 und z_2 sind mit dem Terminplanvorgang t durch die Relation R_1 zu verknüpfen.

Diskussion: Die Reihenfolge der Filterauswertung ist wesentlich. Im Beispiel kann die Bestimmung von A_2 erst dann erfolgen, wenn A_3 bekannt ist.
Um eine weiter differenzierte Auswahl insbesondere bzgl. der räumlichen Lage zu Bauabschnitte definierenden Achsen und Zonen zu ermöglichen, ist neben alphanumerischen Filtern für einfache Attribute wie Objekttyp und Geschoss auch die Auswertung der komplexen Eigenschaften Position und Geometrie eines CAD-Objektes erforderlich. Hierbei hat im Bedarfsfall zusätzlich eine automatische Objektteilung zu erfolgen. In der Sprache sind entsprechende Konstrukte vorzusehen. Die zugehörigen geometrischen Algorithmen sind in Kapitel 4.7 angegeben. Aufgrund des erhöhten Berechnungsaufwandes für geometrische Algorithmen ist die Menge an Objekten, auf die diese anzuwenden

sind, jedoch zu minimieren. Dieses ist bei der Auswertungsreihenfolge zu beachten. Ist die Zielmenge Z bestimmt, so sind die enthaltenen Objekte im Sinne der Terminplanung weiterzuverarbeiten. Erfolgt die Verknüpfung zwecks Visualisierung, so ist ein neues Link4D-Objekt zu erzeugen. Erfolgt sie zwecks Berechnung der Vorgangsdauer, so ist die Anzahl der in Z enthaltenen Objekte zu bestimmen oder einer ihrer Attributwerte zu summieren. Hierfür sind in der Sprache entsprechende Funktionen vorzusehen. Zusätzliche Funktionen, wie z.B. die Bestimmung von Extremwerten oder die Selektion der ausgewählten Objekte in einer 3D Ansicht, können dazu dienen, dem Terminplaner einen Überblick über die in der Zielmenge enthaltenen Objekte zu verschaffen.

4.6.2 Spezifikation

Folgende Basiskonstrukte werden von der Sprache bereitgestellt:

1. *Bestimmung einer Auswahlmenge* A_j von Objekten eines Teilmodells Θ_j mittels attributbasierten, alphanumerischen Filtern,
2. *Extraktion von Attributwerten* für alle Objekte der Auswahlmenge A_j, z.B. aller Identifikatoren der in A_j enthaltenen Objekte,
3. *Schachtelung von Ausdrücken* zur Filterung in weiteren durch Relationen verknüpften Teilmodellen. Hierbei wird das Extraktionsergebnis einer Auswahlmenge als Grundlage für die Filterung im nächsten Teilmodell verwendet.
4. *Weiterverarbeitung* der extrahierten Informationen mittels Funktionen.

Zur Berücksichtigung räumlicher Aspekte werden zusätzlich folgende Funktionalitäten bereitgestellt:

5. *räumliche Filterung*,
6. *Verfeinerung der Objektgranularität* von Θ_j (falls dieses aufgrund der Semantik eines räumlichen Filterkriteriums zur Bestimmung von A_j erforderlich ist).

Im Weiteren werden die einzelnen Elemente der Sprache thematisch gruppiert vorgestellt. Eine Erweiterung der Sprache oder Anpassung an andere Anwendungsgebiete kann durch das Hinzufügen weiterer Funktionen, Filter und Datentypen erfolgen.

Elemente: Ein Element (*element*) ist eine Basisinformation. Hierbei handelt es sich um einen Attributwert eines Objektes. Folgende Datentypen werden für Elemente unterstützt:

- Ganzzahlen (*integer*), z.B. +124, 2, -12,
- Fließkommazahlen (*floatingPoint*), z.B. -3.5, 10e-4, .03,
- Datumsangaben (*date*) mit optionaler Zeitangabe, z.B. 03.05.2009 09:43,
- Zeichenketten (*string*) in Anführungszeichen, z.B. "Vorgang_1".

Mengen von Elementen: Ein Mengenausdruck wird in der Sprachdefinition als *set_term* bezeichnet. In der Sprachsyntax wird er durch ein umschließendes Paar geschweifter Klammern gekennzeichnet. Wie in der Mathematik, kann eine Menge auf zwei Arten definiert werden:

- durch Aufzählung ihrer Elemente (*setElementList*), z.B. {3, 7, 41},
- durch Angabe eines Bildungsgesetzes (*setCreationTerm*).

Die unterstützte Form der Definition eines Bildungsgesetzes entspricht dem Prinzip der Datenfilterung und Extraktion von Attributwerten. Sie hat folgenden syntaktischen Aufbau:

$$\{\Theta_j : filterList \text{ -> } extract\}$$

Zunächst wird das Teilmodell Θ_j angegeben, das die Grundlage für die Filterung und Extraktion bildet. Ist das erforderliche Teilmodell explizit im Modell vorhanden, so kann es über seinen Namen (*identifier*) direkt angesprochen werden. Existiert es noch nicht explizit, z.B. wenn es sich um eine Relation handelt, deren Elemente erst durch Auswertung ermittelt werden müssen, so kann an dieser Stelle eine Funktion verwendet werden, die das Teilmodell während der Auswertung erzeugt (*setCreationFunction*). Je nach Erfordernis kann diese Funktion mehrere Eingabeparameter erhalten, z.B. eine Elementliste mit Identifikatoren der Objekte, für die das Zutreffen einer Relation auszuwerten ist.
Nach der Spezifikation von Θ_j auf eine der beiden genannten Arten, folgt optional die Angabe einer Liste von Filterkriterien (*filterList*) zur Spezifikation der Auswahlmenge A_j. Wird kein Filter angegeben, so folgt $A_j = \Theta_j$. Für die in A_j enthaltenen Objekte werden anschließend die durch *extract* spezifizierten Informationen extrahiert. Diese bilden die durch das Bildungsgesetz erzeugte Menge.

Extraktion: Für die Spezifikation der zu extrahierenden Informationen aus den in A_j enthaltenen Objekten kann folgende Auswahl getroffen werden:

- ein Attributname (*feature*) zur Extraktion der zugehörigen Attributwerte,
- *allValues* zur Extraktion aller vorkommenden Attributwerte,
- *allAttributes* zur Extraktion aller existierenden Attributnamen.

Die beiden zuletzt genannten Schlüsselwörter erlauben es dem Anwender, sich einen Überblick über den Datenbestand zu verschaffen. In allen Fällen stellt das Ergebnis der Extraktion stets eine Menge von Elementen dar.

Attributbasierte Filterkriterien: Die Spezifikation der Auswahlmenge A_j kann durch alphanumerische Filterkriterien für Attributwerte erfolgen. Mehrere Filterkriterien werden mit und (,) bzw. oder (|) kombiniert. Ein Filterkriterium besteht aus der Nennung des betrachteten Attributnamens und einer Operation. Bei der Angabe des Attributnamens oder -werts können folgende Platzhalter verwendet werden:

- ?, Platzhalter für ein beliebiges Zeichen,
- *, Platzhalter für eine beliebige Anzahl beliebiger Zeichen.

Die Verwendung von Platzhaltern ermöglicht es z.B., einen Operator auf alle Werte einer geordneten Attributgruppe anzuwenden. Folgende Operatoren stehen zur Verfügung, deren Auswertung stets einen Wahrheitswert ergibt:

- unäre Operatoren: *defined*, *undefined*,
- binäre Operatoren: <, >, <=, >=, =, !=, *in*, *notin*.

Mit den beiden unären Operatoren kann getestet werden, ob ein Attribut für ein Objekt existiert oder nicht. Die ersten sechs binären Operatoren führen einen Vergleich zwischen dem zu testenden Attributwert und einem angegebenem Vergleichswert durch. Sie sind für alle vier Element-Datentypen definiert. Die Operatoren *in* und *notin* prüfen, ob der Wert des betrachteten Attributs in einer Elementmenge oder einem Intervall enthalten bzw. nicht enthalten ist. Nur Objekte in Θ_j, die der Kombination aller Filterkriterien gerecht werden bilden die Auswahlmenge A_j und finden bei der Extraktion Berücksichtigung.
Durch die Angabe eines Mengenausdrucks (*set_ term*) als Vergleichswert für die Operatoren *in* bzw. *notin* wird eine Schachtelung von Mengenausdrücken und somit eine Filterung auf mehreren verknüpften Teilmodellen ermöglicht (siehe auch Abbildung 4.15 auf der nächsten Seite):

$$\{\Theta_1 : \text{ Attribut\_1 } in\{2.1, 9., -4.32\} \text{ -> ID}\}$$
$$\{\Theta_2 : \text{ Attribut\_2 } in\{\Theta_1 : filterList \text{ -> Attribut\_1}\} \text{ -> ID}\}$$

Weiterverarbeitung von Elementmengen: Zur Weiterverarbeitung der nach der äußersten Extraktion erhaltenen Elementmenge werden folgende Funktionen (*function*) zur Verfügung gestellt:

- *max* und *min* zur Ermittlung von Extremwerten,
- *sum* zum Summieren von Werten,
- *count* zum Ermitteln der Anzahl von Ergebniselementen,
- *select* zum Selektieren von Objekten in einer 3D Ansicht, deren Identifikatoren im Ergebnis des Sprachausdrucks enthalten sind,
- *linkGeometrie*[23] und *linkQuantities*[23] zur Verknüpfung von Geometrie- und Mengeninformationen mit Terminplanvorgängen.

Den als letztes aufgeführten Funktionen werden die ID des betreffenden Terminplanvorgangs sowie weitere Informationen, wie z.B. die ID des zu verknüpfenden Visual-Objekts, als zusätzliche Parameter übergeben. Parameter können entweder Elemente (*element*) oder Mengenausdrücke (*set_ term*) sein.

$$max(\{\Theta_1 : \text{ Attribut\_1} = 2.30 \text{ -> Attribut\_2}\})$$
$$linkGeometrie(\{\Theta_1 : \text{ Attribut\_1} = 2.30 \text{ -> ID}\}, \text{"Vorgang\_1"}, \text{"betonieren"})$$

[23] In der Umsetzung wurden die Kurzformen *linkg* und *linkq* verwendet.

Die Verwendung von Funktionen zur Weiterverarbeitung innerhalb eines Sprachausdrucks ist optional.

Räumliche Anfragen: Eine Einbettung des in Kapitel 4.3 beschriebenen Konzeptes der räumlichen Adressierung von Bauteilen erfordert die Kenntnis räumlicher Relationen zwischen Zonen bzw. Achsen und CAD-Objekten. Diese Informationen sind jedoch im Bauwerksinformationsmodell nicht explizit hinterlegt, sondern müssen zum Zeitpunkt der Sprachinterpretation aus Positions- und Geometrieinformationen gewonnen werden. Hierfür werden zwei Funktionen (*setCreationFunction*) in der Sprache vorgesehen:

- *inZoneRelation*(*set_ term*, *set_ term*)
  1. Parameter: Menge zu prüfender Objekte, 2. Parameter: Menge von Zonen

- *betweenAxisRelation*(*set_ term*, *set_ term*)
  1. Parameter: Menge zu prüfender Objekte, 2. Parameter: Menge von Achsen

Diese Funktionen erzeugen jeweils ein neues Teilmodell, welches diejenigen Objekte enthält, für die die angeforderte Relation zutrifft. Erfordert die strikte räumliche Auswertung eine Objektteilung, so wird diese automatisch durchgeführt und die Relation für die Teilobjekte ausgewertet.

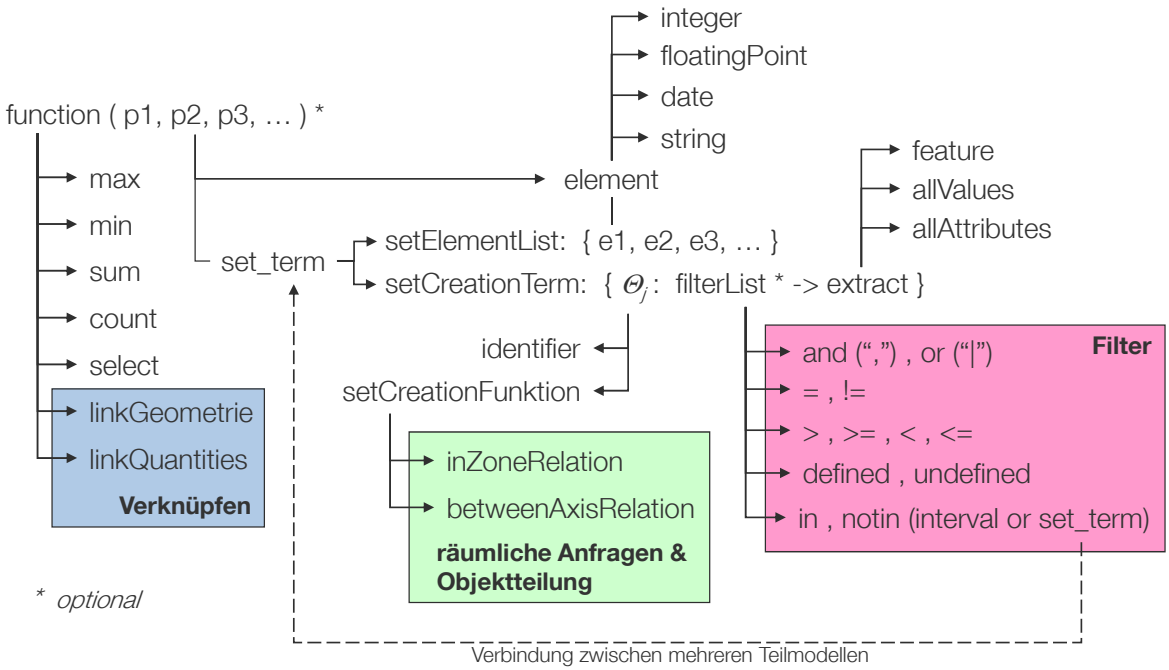


Abbildung 4.15: Überblick über die Sprachkonstrukte der Verknüpfungssprache

Auswertungsreihenfolge: Ein gültiger Sprachausdruck besteht aus:

- einer Funktion (*function*) oder

- einem Mengenausdruck (*set_ term*).

Beide können weitere Mengenausdrücke enthalten, die ihrerseits beliebig tief geschachtelt sein können. Eine Auswertung geschachtelter Ausdrücke hat stets von innen nach außen zu erfolgen. Handelt es sich bei einem Mengenausdruck um ein Bildungsgesetz (*setCreationTerm*), so hat die Auswertung des Mengenausdrucks in folgender Reihenfolge zu erfolgen:

1. Auswahl- (*identifier*) bzw. Erzeugung (*setCreationFunction*) des betrachteten Teilmodells Θ_j,
2. Auswertung der optionalen Filterliste zur Bestimmung der Objekte der Auswahlmenge A_j,
3. Extraktion der angeforderten Informationen aus A_j.

Die Auswertung mit und (,) bzw. oder (|) kombinierter Filter in der Filterliste erfolgt standardmäßig von links nach rechts, d.h. die beiden Operatoren haben die gleiche Priorität. Die Auswertungsreihenfolge kann jedoch durch das Setzen von runden Klammerpaaren beeinflusst werden. In Klammern eingeschlossene Ausdrücke werden von innen nach außen ausgewertet.
Mit der Sprache können verschiedene Sprachausdrücke gebildet werden, die das gleiche Ergebnis liefern. Dabei kann die Position von Filterkriterien innerhalb der Auswertungsreihenfolge Einfluss auf die Auswertungsgeschwindigkeit haben. Insbesondere ist dies bei der Verwendung räumlicher Anfragen der Fall, da die dabei verwendeten geometrischen Algorithmen gegenüber alphanumerischen Vergleichen einen erhöhten Berechnungsaufwand aufweisen. Beispiel:

$$\{inZoneRelation(\{C: \text{ -> ID}\}, \{\text{"Zone A"}\}): \text{ Bauteiltyp} = \text{"Stütze" -> ID}\}$$
$$\{inZoneRelation(\{C: \text{ Bauteiltyp} = \text{"Stütze"-> ID}\}, \{\text{"Zone A"}\}): \text{ -> ID}\}$$

Im ersten Ausdruck erfolgt zunächst eine geometrische Prüfung *aller* CAD-Objekte bzgl. Zone A. Diejenigen CAD-Objekte, die in Zone A enthalten sind, bilden das Teilmodell für die anschließende Filterung nach Bauteilen vom Typ Stütze. Im zweiten Ausdruck ist die Situation umgekehrt. Es werden zunächst alle Stützen aus dem Teilmodell der CAD-Objekte gefiltert und nur noch diese geometrisch bzgl. Zone A überprüft. Dieses kann bei großen Modellen eine erhebliche Einsparung an Rechenaufwand bedeuten.
Sprachausdrücke sollten vom Nutzer daher so aufgebaut werden, dass die Anzahl der mit geometrischen Algorithmen zu untersuchenden Objekte zunächst durch rechnerisch weniger aufwändige, alphanumerische Filterkriterien stark reduziert wird. Die geometrische Auswertung erfolgt anschließend in zwei Schritten:

1. Prüfung, ob das aktuell untersuchte Objekt innerhalb, außerhalb oder teilweise innerhalb des Suchbereichs liegt.
2. Liegt es teilweise innerhalb, so erfolgt eine Objektteilung, bei der die erhaltenen Teile als innerhalb oder außerhalb des Suchbereichs klassifiziert werden.

Abbildung 4.16 auf der nächsten Seite zeigt die drei verwendeten Mechanismen zur Erstellung von regelbasierten Verknüfpungen in der vorgeschlagenen Auswertungsreihenfolge.

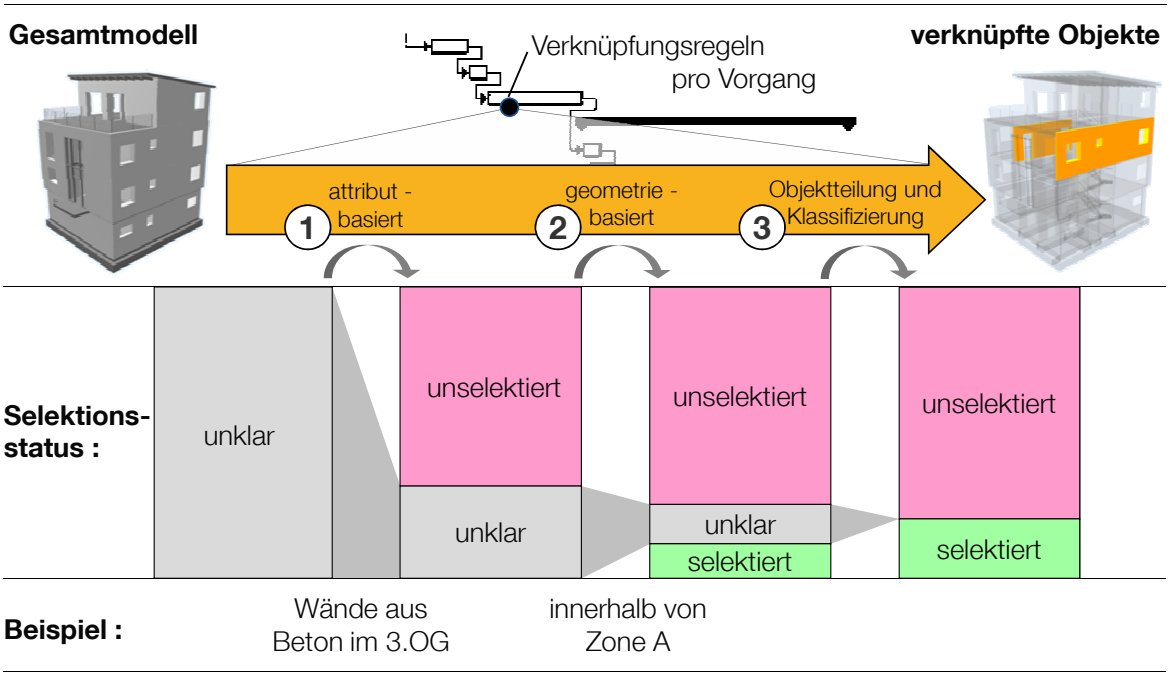


Abbildung 4.16: Prinzip und empfohlene Auswertungsreihenfolge von regelbasierten Verknüpfungen zwischen Terminplan und Bauwerksmodell

4.6.3 Makros

Sprachausdrücke, die sich über mehrere Teilmodelle erstrecken und/oder eine Vielzahl von Filterkriterien aufweisen, erreichen schnell eine beträchtliche Länge. Außerdem sind oft sehr ähnliche Verknüpfungsregeln für mehrere Terminplanvorgänge anzugeben. Z.B. wiederholen sich die Terminplanvorgänge für die Rohbauerstellung eines Geschosses in gleicher Weise für alle Bauwerksgeschosse (insbesondere bei Hochhäusern). Die zugehörigen Verknüpfungsregeln zur Erstellung einer 4D-Simulation unterscheiden sich von Geschoss zu Geschoss nur durch das Filterkriterium für die Geschossnummer.
Als Hilfsmittel wurde daher in die Sprache die Definition und Nutzung von parametrisierten Abkürzungen (Makros) integriert. Dadurch lassen sich lange Sprachausdrücke in übersichtlichere Teilausdrücke aufbrechen und durch die Parametrisierung in abgewandelter Form mehrfach verwenden. Es werden die in Tabelle 4.2 angegebenen Funktionalitäten bereitgestellt:

Aktion	Syntax
Ausgabe einer Liste aller definierten Makros	listMacros
Definition eines Makros	*#MacroName(P1, P2, ...) := Sprachausdruck*
Löschen eines Makros	*#MacroName() :=*
Aufruf eines Makros	*#MacroName(P1, P2, ...)*

Tabelle 4.2: In die Verknüpfungssprache eingebettete Makrofunktionalitäten

Die ersten drei genannten Befehle werden alleinstehend verwendet. Der Aufruf von Makros kann hingegen sowohl in die Definition eines anderen Makros als auch in einen zu

interpretierenden Ausdruck der Verknüpfungssprache eingebettet werden. Jede Eingabe wird zunächst von einem Makroprozessor analysiert und die entsprechenden Befehle ausgeführt. Wird in einem Ausdruck der Verknüpfungssprache ein Aufruf eines Makros gefunden, so wird dieser durch den Inhalt des Makros ersetzt (reine Textersetzung) und der expandierte Sprachausdruck an den Interpreter für die Verknüpfungssprache weitergegeben. Die Grammatik für den Makroprozessor ist in Anhang B ebenfalls in erweiterter Backus-Naur-Form angegeben.

Beispiel:

```
> #Macro1(parm1):= {C: Bauteiltyp="Wand", Geschoss="parm1"-> ID}
> #Macro2():= #Macro1("3.OG")
> listMacros
     #Macro1(parm1)  {C: Bauteiltyp="Wand", Geschoss="parm1"-> ID}
     #Macro2()       {C: Bauteiltyp="Wand", Geschoss="3.OG"-> ID}
> #Macro2()
     query:  {CAD: Bauteiltyp="Wand", Geschoss="3.OG"-> ID}
> #Macro2():=
> listMacros
     #Macro1(parm1)  {C: Bauteiltyp="Wand", Geschoss="parm1"-> ID}
```

4.6.4 Weiterführende Literatur

In [Riedel u. Wender 2007] und [Wender 2009] wird über eine deklarative Anfragefunktionalität auf Basis von OQL[24] berichtet, die die Auswertung von Beziehungen zwischen Objekten verschiedener Teilmodelle eines Partialmodellverbunds hinweg erlaubt. [Weiss 2005] beschäftigt sich in diesem Zusammenhang mit der dreidimensionalen Analyse des Gesamtdatenbestands.
In [Adachi 2002] wird für die Kommunikation mit einem IFC-basierten Modellserver eine Sprache zur Abfrage von Teilmodellen spezifiziert. Die Sprache basiert auf XML[24] und ist an SQL[24] angelehnt. Sie bietet jedoch keine Möglichkeit für die Auswertung räumlicher Beziehungen. Diese werden hingegen schwerpunktmäßig in [Borrmann 2008] betrachtet, allerdings werden dort weder Achsanfragen noch eine Objektteilung berücksichtigt.
Grundlegende Algorithmen für geometrische Anfragen im 2D sowie der Umgang mit numerischen Genauigkeitsproblemen werden in [Brinkhoff 2005] beschrieben. Datenstrukturen für Geoinformationsdatenbanken als Basis für räumliche Anfragen werden in [Abdul-Rahman u. Pilouk 2008] und [Lee u. Zlatanova 2009] dargestellt.

4.7 Geometrische Algorithmen

Für die Beschreibung der dreidimensionalen Geometrie von Objekten im Computer existieren mehrere verschiedene Konzepte ([Foley u. a. 1995], [Bungartz u. a. 2002], [Brüderlin u. Meier 2001]). Prinzipiell lassen sich kanten-, flächen- und volumenorientierte

[24] siehe Glossar

Beschreibungen unterscheiden, die unterschiedliche Leistungsfähigkeiten, Genauigkeiten und Speicheranforderungen aufweisen.
Zur Darstellung in Computern wird heutzutage in der Regel eine B-rep[25]-Beschreibung verwendet. Hierbei wird ein Objekt durch seine Oberfläche beschrieben, die näherungsweise aus einem Netz von geometrisch einfachen Flächen (z.B. Dreiecken, Vierecken) zusammengesetzt wird. Durch die hierfür vorhandene direkte Hardwareunterstützung erfolgt eine effiziente Visualisierung, die auch bei komplexen Modellen noch eine interaktive Inspektion ermöglicht. Jedoch ist der Speicherbedarf für eine B-rep-Beschreibung sehr hoch. Zudem gehen Informationen über die Zusammensetzung der Gesamtgeometrie aus Einzelelementen, wie z.B. bei einer Wand mit Fensteröffnungen, verloren. Daher wird in Datenmodellen für die Speicherung von Bauwerksinformationen diese Methode nur für sehr unregelmäßige Objekte verwendet und ansonsten auf andere Darstellungsarten zurückgegriffen. Üblich ist eine volumenorientierte Beschreibung mittels CSG (Constructive Solid Geometrie). Bei dieser Methode werden komplexe Körper aus einfacheren Körpern, die mit den booleschen Operationen Vereinigung, Durchschnitt und Differenz kombiniert werden, zusammengesetzt. Einfachere Körper werden meistens mit der Extrusionsmethode beschrieben. Hierbei werden die Kanten einer Grundfläche und ein zugehöriger Extrusionspfad angegeben. Die Verwendung von Non-Uniform Rational B-Splines (NURBS), die eine Beschreibung beliebiger Formen ermöglichen, haben sich im Bauwesen hingegen bisher nicht durchsetzen können.
Aufgrund der Verwendung unterschiedlicher Geometriebeschreibungsarten für die Datenhaltung und die Visualisierung hat eine Konvertierung zwischen beiden Beschreibungsformen zu erfolgen. Die nachfolgend vorgestellten geometrischen Algorithmen beziehen sich jedoch ausschließlich auf B-rep-Geometriebeschreibungen. Zusätzlich wird vorausgesetzt, dass die Körperoberflächen durch reine Dreiecksnetze beschrieben werden, deren Flächennormalen stets nach außen weisen. Dieses ermöglicht einerseits eine gleichartige, zumindest näherungsweise Erfassung jeglicher Geometrie, und andererseits stellt es die übliche Beschreibungsform zum Zeitpunkt der Visualisierung dar. Aufgrund der stets erwünschten visuellen Kontrolle kann davon ausgegangen werden, dass die Auswertung von Sprachausdrücken, die zur Nutzung geometrischer Algorithmen führen, immer parallel zu einer dreidimensionalen Visualisierung erfolgt und somit eine B-rep-Geometriebeschreibung bereits vorliegt bzw. bei neu erzeugten Objekten direkt benötigt wird. Eine Speicherung von durch einen Sprachausdruck neu erzeugten Geometrieobjekten (im Rahmen einer Objektteilung) ist zudem aufgrund der Rekonstruierbarkeit mittels Neuinterpretation des Sprachausdrucks nicht zwingend erforderlich. Somit stellen die genannten Anforderungen keine besondere Einschränkung dar.
Für die modellbasierte Terminplanung werden von der spezifizierten Verknüpfungssprache die zwei Funktionen (*setCreationFunction*) *inZoneRelation* und *betweenAxisRelation* bereitgestellt, die die entsprechenden topologischen Beziehungen zwischen den angegebenen CAD- und Hilfsobjekten überprüfen und die positiv getesteten CAD-Objekte zurückgeben. Ist die topologische Beziehung nur von Teilen des zu prüfenden Objektes erfüllt, so sind nur diese zurückzugeben. Hierfür ist eine Funktionalität zur Objektteilung anhand von Hilfsobjekten (Zonen und Achsen) erforderlich.

[25] boundary representation

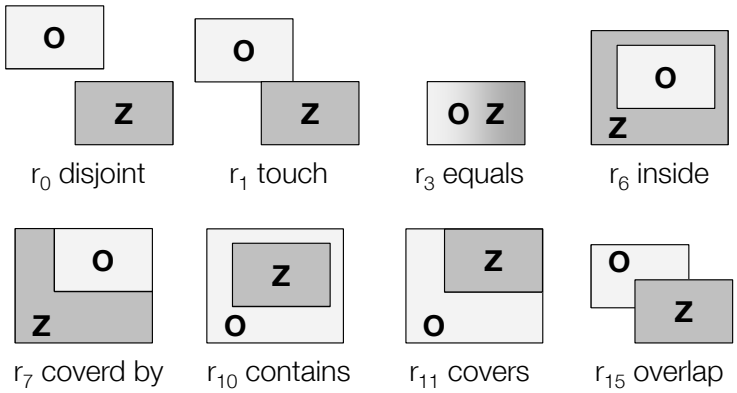


Abbildung 4.17: Veranschaulichung der acht realisierbaren aus 16 formal definierten Punktmengenrelationen nach [Egenhofer u. Franzosa 1991]

In [Borrmann 2008] werden im Rahmen einer räumlichen Anfragesprache für Bauwerksmodelle die theoretischen Grundlagen für diverse metrische, direktionale, topologische und objekterzeugende Operatoren für null- bis dreidimensionale Geometrieobjekte vorgestellt. Im Rahmen der vorliegenden Arbeit sind insbesondere die dort definierten topologischen Operatoren von Interesse. Sie gehen auf die Untersuchung von Relationen zwischen Punktmengen in [Egenhofer u. Franzosa 1991] zurück. Die Umsetzung in [Borrmann 2008] erfolgte jedoch im Gegensatz zum hier verfolgten Ansatz unter Verwendung einer volumenorientierten Geometriebeschreibung (Normzellenaufzählungsschema in Verbindung mit einer Oktalbaumcodierung). Nachteil dieses Verfahrens stellt je nach verfolgter Strategie entweder der wiederkehrende Aufwand für die Generierung einer solchen Datenstruktur zum Zeitpunkt der Anfrageauswertung oder der hohe Speicherbedarf bei angemessener Auflösungsgenauigkeit im Fall initialer Berechnung dar. Die erzielten Antwortzeiten erscheinen für eine interaktive Bearbeitung realer Projekte verhältnismäßig lang. Ein Operator zur Bestimmung von Objekten, die sich zwischen zwei Achsen befinden, sowie die Teilung vorhandener Geometrieobjekte anhand von Hilfsobjekten werden nicht betrachtet.

4.7.1 Zonenanfrage

Grundlage der Funktion *inZoneRelation(Objekte, Zonen)* ist der Test der relativen Lage zweier geometrischer Objekte O und Z. Werden O und Z als Punktmengen betrachtet und deren Oberflächen mit ∂O bzw. ∂Z sowie deren Inneres mit O° bzw. Z° bezeichnet, so ergeben sich nach [Egenhofer u. Franzosa 1991] im dreidimensionalen Raum acht mögliche Relationen (siehe Abbildung 4.17 und Tabelle 4.3). Diese werden unter Vernachlässigung einer Berührung von Oberflächen zu drei möglichen Ergebnissen für den Test der relativen Lage eines CAD-Objektes O und einer Zone Z zusammengefasst (siehe Tabelle 4.4). Die Zone Z der Anfrage wird dabei gedanklich als Stanzform betrachtet. Je nach Konfiguration liegt das gesamte Objekt innerhalb der Form, es wird ein Teil des Objektes ausgestanzt, oder das gesamte Objekt liegt außerhalb der Form. Liegt Z in O, so wird ein Hohlraum erzeugt. Dieser Fall (r_{10}) ist für die modellbasierte Terminplanung jedoch normalerweise ohne Relevanz. Er tritt z.B. auf, wenn eine Anfragezone vollständig innerhalb eines Raumobjektes liegt.

		$\partial O \cap \partial Z$	$O^\circ \cap Z^\circ$	$\partial O \cap Z^\circ$	$O^\circ \cap \partial Z$
r_0	O and Z are disjoint	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
r_1	O and Z touch	$\neg\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
r_3	O equals Z	$\neg\emptyset$	$\neg\emptyset$	$\emptyset$	$\emptyset$
r_6	O is inside of Z	$\emptyset$	$\neg\emptyset$	$\neg\emptyset$	$\emptyset$
r_7	O is covered by Z	$\neg\emptyset$	$\neg\emptyset$	$\neg\emptyset$	$\emptyset$
r_{10}	O contains Z	$\emptyset$	$\neg\emptyset$	$\emptyset$	$\neg\emptyset$
r_{11}	O covers Z	$\neg\emptyset$	$\neg\emptyset$	$\emptyset$	$\neg\emptyset$
r_{15}	O and Z overlap	$\neg\emptyset$	$\neg\emptyset$	$\neg\emptyset$	$\neg\emptyset$

Tabelle 4.3: Punktmengenrelationen im $\mathbb{R}^3$ nach [Egenhofer u. Franzosa 1991]

		$O^\circ \cap Z^\circ$	$O^\circ \cap \partial Z$
O befindet sich voll innerhalb von Z	r_3, r_6, r_7	$\neg\emptyset$	$\emptyset$
O befindet sich voll außerhalb von Z	r_0, r_1	$\emptyset$	
O wird von Z geschnitten	$(r_{10}), r_{11}, r_{15}$		$\neg\emptyset$

Tabelle 4.4: Drei mögliche Ergebnisse des Lagetests eines Objektes bzgl. einer Zone

Algorithmus 4.1 beschreibt das Vorgehen beim Analysieren der bestehenden räumlichen Beziehung zwischen O und Z auf Basis einer B-rep-Geometriebeschreibung. Die Analyse erfolgt dabei in drei Schritten. Zunächst erfolgt ein Bounding-Box-Test zwischen den Objekten O und Z. Liefert dieser keine eindeutige Aussage, so wird, wie nachfolgend erläutert, im zweiten Schritt das Oberflächennetz von O durch Zerlegung von Dreiecken so aufbereitet, dass keines der verbleibenden Dreiecke die Oberfläche von Z durchdringt. Wird dabei ein Schnitt zweier Dreiecke festgestellt, so schneiden sich O und Z. Wurde keinerlei Berührung von Dreiecken gefunden bzw. wurden die betreffenden Dreiecke von O zerlegt, so wird im letzten Schritt eine endgültige Aussage auf Basis einer Klassifikation der Dreiecke von O bezüglich Z getroffen. Eine eindeutige Klassifikation von Dreiecken ist Aufgrund der zuvor erfolgten Elimination von

Algorithmus 4.1: Objekt-in-Zone-Test

```
1  if bounding box of O and Z do not intersect then
2      return O is outside of Z
3  else                                  // all cases are still possible
4      int result = splitIntersectingTriangles(O,Z)
5      switch result do
6          case INTERSECTION_FOUND
7              return O and Z intersect
8          case CONTACT_FOUND
9              return classifyObjectByAllFaces(O,Z)
10         case DISJOINT
11             return classifyObjectByFirstFace(O,Z)
```

Algorithmus 4.2: Bounding-Box-Test

```
1  BoundingBox OB, ZB
2  if  OB.upper.x > ZB.lower.x ∧ ZB.upper.x > OB.lower.x
3   ∧ OB.upper.y > ZB.lower.y ∧ ZB.upper.y > OB.lower.y
4   ∧ OB.upper.z > ZB.lower.z ∧ ZB.upper.z > OB.lower.z   then
5      return true
6  else
7      return false
```

Durchdringungen stets möglich.

Bounding-Box-Test: Dieser verhindert, dass für Objekte, die außerhalb der Bounding-Box der betrachteten Zone Z liegen, alle Dreiecke unnötig auf Schnitt mit ∂Z getestet werden. Hierdurch reduziert sich der Gesamtberechnungsaufwand beträchtlich. Dieser Effekt ist umso größer, je mehr Schnittberechnungen vermieden werden können, also insbesondere bei Objekten, deren Oberfläche mittels vieler Dreiecke beschrieben ist (z.B. Rundstützen). Der Test erfolgt unter Verwendung achsenparalleler Bounding-Boxen entsprechend Algorithmus 4.2. Dabei bezeichnet *lower* den Eckpunkt der Bounding-Box mit den numerisch kleinsten Koordinatenwerten und *upper* denjenigen mit den größten. Die Verbindung beider Punkte beschreibt die Raumdiagonale der Boundingbox. Der Test liefert *false*, wenn $OB^\circ \cap ZB^\circ = \emptyset$ gilt, d.h. bei Lage der beiden Boundingboxen zueinander entsprechend den Relationen r_0 und r_1 (siehe Abbildung 4.17). Andernfalls wird *true* zurückgegeben. Eine Berührung der beiden Bounding-Boxen wird somit nicht als Überlappung gewertet.
Innerhalb der Funktion *inZoneRelation* der Verknüpfungssprache sind nach Algorithmus 4.1 pro angegebener Zone alle der Funktion übergebenen Objekte zu testen. Die Verwendung eines Bounding-Box-Tests ändert daran nichts, sie kann lediglich eine Reduktion des Rechenaufwands um einen konstanten Faktor erzielen. Um eine Reduzierung der zu prüfenden Objekte zu erreichen, können raumpartitionierende Baumstrukturen (z.B. AABB[26]-Bäume) verwendet werden. Hierdurch wird die Untersuchung auf Objekte in der Nähe der Prüfzone eingegrenzt. Einen Überblick über derartige Verfahren gibt [Ericson 2005]. In [Schraufstetter u. Borrmann 2007] werden AABB-Bäume auch im Zusammenhang mit denen von Borrmann untersuchten topologischen Operatoren verwendet. Einen Algorithmus zum Aufbau einer raumpartitionierenden Baumstruktur für eine B-rep-Geometriebeschreibung und die Auswertung topologischer Beziehungen findet man in [Vanecek 1990].
Im Rahmen dieser Arbeit wurde auf die Verwendung entsprechender Verfahren aus folgenden Gründen verzichtet:

- Es entsteht zusätzlicher Speicher- und Rechenaufwand für die Überführung der vorliegenden Modelldaten in die entsprechende Baumstruktur.
- Die bisher ohne Verwendung solcher Strukturen an realen Testmodellen erzielten Antwortzeiten erzeugen, selbst beim Test aller Objekte eines Modells, keine

[26] Axis Aligned Bounding Box

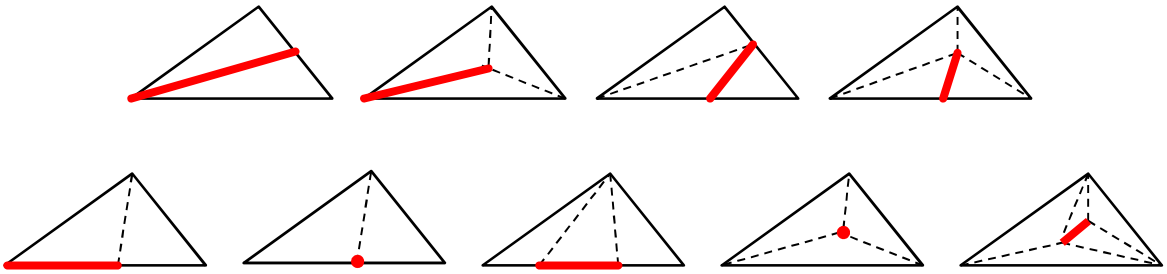


Abbildung 4.18: Zerlegung von geschnittenen Dreiecken nach [Castanheira 2003] [28]

Wartezeiten.

- In der Regel erfolgt eine geometrische Auswertung erst nach einer vorangegangenen Reduktion der Modelldaten durch vorgeschaltete alphanumerische Filterkriterien (vgl. Kapitel 4.6.2). Hierdurch wird die Antwortzeit für Anfragen nochmals deutlich verringert.

Schnitt und Zerlegung von Dreiecken: Die Schnittprüfung aller Dreiecke des Oberflächennetzes von Z mit denen von O und die ggf. erforderliche Zerlegung der Dreiecke von O erfolgt in einer eigenen Methode *splitIntersectingTriangles*. Dabei wird für jedes Dreieck von Z zunächst dessen Bounding-Box DB mit der Bounding-Box OB von O auf Überlappung geprüft. Weisen diese eine Überlappung auf, so wird die Überprüfung mit DB und der Bounding-Box jedes einzelnen Dreiecks von O fortgesetzt. Erst wenn auch hierbei eine Überlappung festgestellt wird, erfolgt eine Schnittberechnung zwischen den entsprechenden beiden Dreiecken. Im Gegensatz zum Bounding-Box-Test zwischen den Objekten O und Z wird bei den hier verwendeten Bounding-Box-Tests eine Berührung als Überlappung betrachtet (durch Ersetzen von $>$ durch $>=$ in Algorithmus 4.2). D.h., diese Bounding-Box-Tests liefern *false* nur dann zurück, wenn zwischen den Boxen eine Konfiguration entsprechend der Relation r_0 vorliegt.
Die eigentliche Berechnung des Schnittes zwischen den beiden Dreiecken erfolgt nach [Möller 1997] [27]. Wird hierbei ein Schnitt gefunden, so ist zwischen einem echten Schnitt und einer Berührung beider Dreiecke zu unterscheiden. Ein echter Schnitt liegt vor, wenn die Schnittlänge größer null ist und für *beide* Dreiecke das Produkt der vorzeichenbehafteten Abstände zweier Punkte zur Ebene des jeweils anderen Dreiecks negativ ist.
Wurde ein echter Schnitt gefunden, so wird die Untersuchung innerhalb der Methode *splitIntersectingTriangles* sofort abgebrochen und ein entsprechender Wert zurückgegeben (INTERSECTION_FOUND). Wird eine Berührung entlang einer Kante oder in einem Punkt gefunden, so wird das entsprechende Dreieck von O so in kleinere Dreiecke zerlegt, dass die Berürung im Oberflächennetz von O einer Kante oder einem Knoten entspricht. Hierbei sind die in Abbildung 4.18 gezeigten neun Fälle zu behandeln. Die neu erzeugten Dreiecke von O werden in die weitere Schnittprüfung einbezogen.
Tritt der Sonderfall ein, dass zwei getestete Dreiecke koplanar sind, so braucht keine

[27] Eine Implementierung in C ist unter http://jgt.akpeters.com/papers/Moller97/ verfügbar.
[28] Eine Implementierung in Java ist unter www.geocities.com/danbalby/ verfügbar.

V :	the query point v coincides with a Vertex of polyhedron Z.
E :	the query point v is in the relative interior of an Edge of polyhedron Z.
F :	the query point v is in the relative interior of a Face of polyhedron Z.
i :	the query point v is strictly interior to polyhedron Z.
o :	the query point v is strictly exterior to polyhedron Z.

Tabelle 4.5: Ergebnisse eines Punkt-in-Polyeder-Tests nach [O'Rourke 1998]

genauere Schnittberechnung und Zerlegung zu erfolgen. Denn ist das betreffende Dreieck von O Teil einer Berührungsfläche zwischen O und Z, und ragt es darüber hinaus, so existiert an jeder Begrenzungskante der Berührungsfläche noch ein anderes Dreieck von Z, das eine Berührung mit diesem erzeugt und im Weiteren zur Zerlegung des Dreiecks führt. Somit werden die Begrenzungskanten einer Berührungsfläche nach Prüfung bzw. Zerlegung aller Dreiecke vom Oberflächennetz von O korrekt berücksichtigt. Im Inneren einer Berührungsfläche können O und Z jedoch durchaus unterschiedliche Dreiecksnetze aufweisen.

Wurden alle Dreiecke behandelt, so existiert keine Durchdringung mehr zwischen einem Dreieck von O und der Oberfläche von Z und jedes Dreieck von O liegt entweder vollständig auf der Oberfläche von Z oder vollständig innerhalb bzw. außerhalb von Z. Je nachdem, ob bei der Aufbereitung eine Berührung festgestellt wurde oder nicht, wird von der Methode *splitIntersectingTriangles* der entsprechende Wert CONTACT_FOUND oder DISJOINT zurückgegeben.

Klassifikation von Dreiecken und Objekten: Ein Dreieck von O kann gemäß seiner Lage bzgl. Z als innerhalb (INSIDE), außerhalb (OUTSIDE) oder auf der Oberfläche liegend klassifiziert werden. Im letzteren Fall kann noch zwischen einer gleich (SAME) oder entgegengesetzt (OPPOSITE) orientierten Flächennormalen unterschieden werden. Das Vorgehen bei der Klassifizierung ist ausführlich in [Laidlaw u. a. 1986] und [Castanheira 2003][28] beschrieben. Grundlage dafür ist die Durchführung eines Punkt-in-Polyeder-Tests für den Mittelpunkt v des zu klassifizierenden Dreiecks. Dabei werden die Schnitte eines Strahls entlang der Flächennormalen beginnend in v mit den Oberflächenelementen von Z ausgewertet. Ein entsprechender Algorithmus ist auch in [O'Rourke 1998][29] beschrieben. Dieser kann die in Tabelle 4.5 aufgeführten Ergebnisse liefern. Im Fall o liegt das Dreieck außerhalb von Z, im Fall i innerhalb. In den Fällen V, E und F liegt das zu klassifizierende Dreieck auf der Oberfläche von Z.

Die endgültige Bestimmung der relativen Lage eines Objektes O bzgl. einer Zone Z erfolgt auf Basis der Klassifikation der Dreiecke. Hat die vorangegangene Prüfung auf Schnitt ergeben, dass die Oberflächen von O und Z sich nicht berühren (DISJOINT), so reicht die Klassifikation jeweils eines Dreiecks von O und Z bzgl. des jeweils anderen Objektes, um gemäß Algorithmus 4.3 auf die relative Lage der beiden Objekte zu schließen. Die Klassifikation eines Dreiecks von Z (Zeile 9) ist dabei erfor-

[29] Eine Implementierung in C und Java ist unter `http://maven.smith.edu/~orourke/books/compgeom.html` verfügbar.

Algorithmus 4.3: Bestimmung der räumlichen Beziehung von O und Z auf der Basis der Klassifikation eines Dreiecks

```
1   method classifyObjectByFirstFace(Object O, Object Z)
2   begin
3       triangle t = get first triangle of O
4       int result = classifyTriangle(t, Z)
5       switch result do
6           case INSIDE
7               return O is inside of Z
8           case OUTSIDE
9               t = get first triangle of Z
10              if classifyTriangle(t, O) = INSIDE then
11                  return O and Z intersect          // Z is inside of O
12              else
13                  return O is outside of Z
```

Algorithmus 4.4: Bestimmung der räumlichen Beziehung von O und Z auf der Basis der Klassifikation von Dreiecken

```
1   method classifyObjectByAllFaces(Object O, Object Z)
2   begin
3       triangles[] ts = get triangles of O
4       int firstResult = classifyTriangle(ts[0], Z)
5       int currentResult
6       int i = 1
7       if firstResult = INSIDE or SAME then
8           for i < number of triangles in O do
9               currentResult = classifyTriangle(ts[i], Z)
10              if currentResult = OUTSIDE or OPPOSITE then
11                  return O and Z intersect
12              i = i+1
13          return O is inside of Z
14      else
15          for i < number of triangles in O do
16              currentResult = classifyTriangle(ts[i], Z)
17              if currentResult = INSIDE or SAME then
18                  return O and Z intersect
19              i = i+1
20          triangle t = get first triangle of Z
21          if classifyTriangle(t, O) = INSIDE then
22              return O and Z intersect              // Z is inside of O
23          else
24              return O is outside of Z
```

derlich, um zu prüfen, ob Z innerhalb von O liegt (Relation r_{10}). Diese Möglichkeit besteht nur, wenn das Dreieck von O als außerhalb von Z liegend klassifiziert wurde. Wurden hingegen zuvor Dreiecksschnitte gefunden (CONTACT_FOUND), so können einige Dreiecke von O vollständig innerhalb und andere vollständig außerhalb von Z liegen. Daher müssen gemäß Algorithmus 4.4 mindestens zwei und im schlimmsten Fall alle Dreiecke von O klassifiziert werden, um die relative Lage von O und Z eindeutig zu bestimmen. Liegen alle Dreiecke von O außerhalb von Z, so ist auch hier anhand eines Dreiecks von Z zu prüfen, ob Z in O liegt (Zeile 20).
Kann ein Objekt O aus mehreren separaten Oberflächennetzen, d.h. aus separaten Objektteilen bestehen, so ist zunächst die relative Lage jedes einzelnen Teils bzgl. Z zu bestimmen und anschließend daraus auf die Lage von O zu schließen. Liegen einige Teile innerhalb und andere außerhalb von Z, so schneiden sich O und Z. Ansonsten liegt O innerhalb bzw. außerhalb von Z.

Aufwandsbetrachtung: Der Berechnungsaufwand für die Auswertung einer Zonenanfrage für ein Objekt O setzt sich aus den Aufwänden der Einzelalgorithmen für die Bounding-Box-Tests sowie die Schnittberechnung, Zerlegung und Klassifikation von Dreiecken zusammen. Da diese annähernd[30] einen linearen Aufwand aufweisen, ist maßgebend, wie oft ein Algorithmus ausgeführt werden muss. Im schlimmsten Fall müssen alle Dreiecke von O mit allen Dreiecken von Z auf Schnitt geprüft sowie alle Dreiecke von O bzgl. ihrer Lage zu Z klassifiziert werden. Somit ergibt sich ein Aufwand von $O(n_O * n_Z)$, wobei n_O bzw. n_Z die Anzahl der Dreiecke in O bzw. Z bezeichnen. Allerdings ist zu beachten, dass die Anzahl der Dreiecke von O infolge Zerlegung zunimmt und es vorkommen kann, dass bei einer Zerlegung neu erzeugte Dreiecke später nochmals zerlegt werden. Somit ist für n_O die Anzahl der im Laufe der Zonenanfrage in O existierenden Dreiecke anzunehmen. In praktisch relevanten Fällen ist allerdings die Bestimmung der relativen Lage von O und Z bereits frühzeitig durch den ersten Bounding-Box-Test bzw. durch das Auffinden eines Schnitts zweier Dreiecke möglich. Für Objekte, die die Bounding-Box von Z überlappen, Z jedoch weder schneiden noch berühren, müssen hingegen zumindest die Bounding-Boxen aller Dreiecke von Z auf Überlappung mit der von O geprüft werden, jedoch findet keine Zerlegung von Dreiecken statt und eine Klassifikation ist nur für maximal zwei Dreiecke erforderlich. Der maximale Aufwand entsteht somit nur, wenn O die Bounding-Box von Z überlappt und Z berührt oder schneidet, jedoch keine sich schneidenden Dreiecke vorliegen.

4.7.2 Achsenanfrage

Kern der Auswertung der Funktion *betweenAxisRelation(Objekte, Achsen)* ist der Test, ob ein Objekt O sich zwischen zwei nicht schneidenden Achsen A und B befindet. Dabei ist zu beachten, dass im Bauwesen eine Achse nicht ausschließlich durch eine Gerade, sondern allgemeingültiger durch einen sich nicht selbstschneidenden Streckenzug gebildet wird. Die Problemstellung wird gemäß Abbildung 4.19 auf eine Zonenanfrage zurückgeführt, wobei die Zone Z den Zwischenbereich der beiden Achsen A und B beschreibt. Es können somit in Analogie zur Zonenanfrage die drei in Tabelle 4.6

[30] siehe [O'Rourke 1998] bzgl. des Aufwands für einen Punkt-in-Polyeder-Test

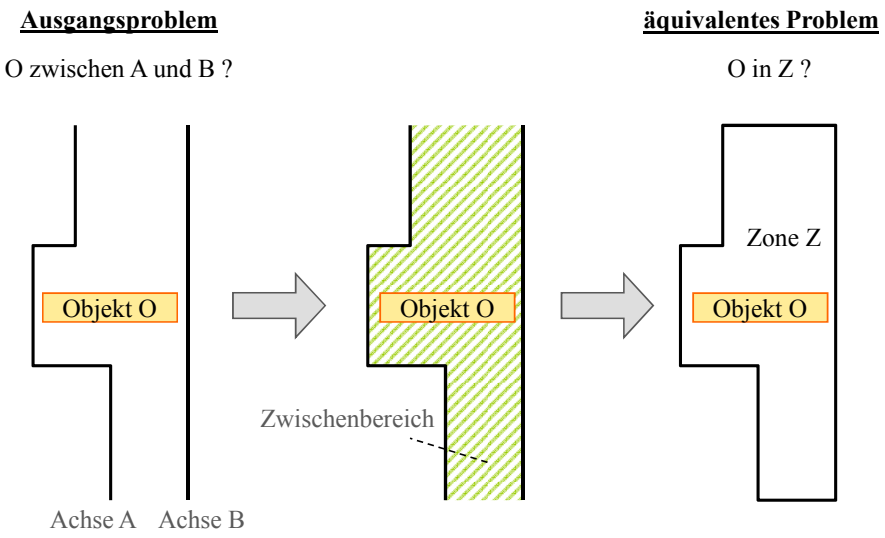


Abbildung 4.19: Transformation einer Achsenanfrage in eine Zonenanfrage

Achsenanfrage	Zonenanfrage
O liegt *zwischen* A und B	O befindet sich voll innerhalb von Z
O liegt *nicht zwischen* A und B	O befindet sich voll außerhalb von Z
O liegt *teilweise zwischen* A und B	O wird von Z geschnitten

Tabelle 4.6: Mögliche Ergebnisse des Lagetests eines Objektes bzgl. zweier Achsen und deren Entsprechung bei einer Zonenanfrage

angegebenen Situationen vorgefunden werden.
Als verbleibende Problemstellung ist der Zwischenbereich zweier gegebener Achsen zu berechnen. Die Lösung dieses Problems kann zunächst im Zweidimensionalen erfolgen. Das erhaltene Polygon kann anschließend, durch Extrusion entsprechend der beteiligten Achsen, in ein dreidimensionales Zonenobjekt überführt werden.
Zunächst ist der Zwischenbereich zu definieren. Dieses hängt eng mit der Bedeutung des im allgemeinen Sprachgebrauch verwendeten Begriffs „zwischen“ zusammen. Hierfür eine Definition zu finden, die in allen denkbaren Situationen einer intuitiven Interpretation entspricht, gestaltet sich als schwierig. Ähnliche Erfahrungen wurden in [Borrmann 2008] mit der Definition direktionaler Operationen gemacht. Im Rahmen dieser Arbeit wurden zwei Definitionen für den Zwischenbereich zweier Achsen betrachtet.

Strenge Definition des Zwischenbereichs:

Sei V die Menge aller Verbindungsstrecken von Achse A nach Achse B

$$V := \{\overline{ab} \mid a \in A \wedge b \in B\} \tag{4.40a}$$

und U_v die Menge aller ungeschnittenen Verbindungsstrecken

$$U_v := \{v \in V \mid v \text{ hat keinen inneren Schnittpunkt mit } A \text{ oder } B\} \tag{4.40b}$$

dann ist der Zwischenbereich Z von A und B definiert durch

$$Z_{streng} := \{z \in \mathbb{R}^2 \mid \bigvee_{u \in U_v} z \in u\} \tag{4.40c}$$

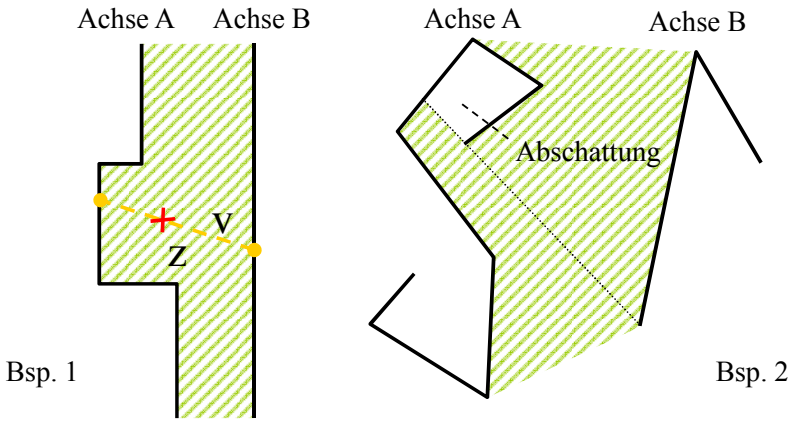


Abbildung 4.20: strenge Definition des Zwischenbereichs

Relaxierte Definition des Zwischenbereichs:

Sei P die Menge aller Pfade von Achse A nach Achse B

$$P := \{\widetilde{ab} \mid a \in A \wedge b \in B\} \tag{4.41a}$$

und H_{AB} die konvexe Hülle der Vereinigung von A und B sowie U_p die Menge aller ungeschnittenen Pfade

$$U_p := \{p \in P \mid p \text{ hat keinen inneren Schnittpunkt mit } A, B, \text{ und } H_{AB}\}, \tag{4.41b}$$

dann ist der Zwischenbereich Z von A und B definiert durch

$$Z_{relaxiert} := \{z \in \mathbb{R}^2 \mid \bigvee_{u \in U_p} z \in u\} \tag{4.41c}$$

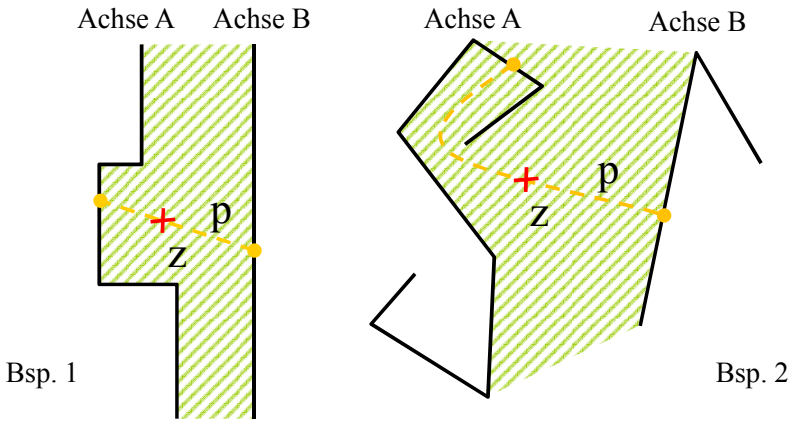


Abbildung 4.21: relaxierte Definition des Zwischenbereichs

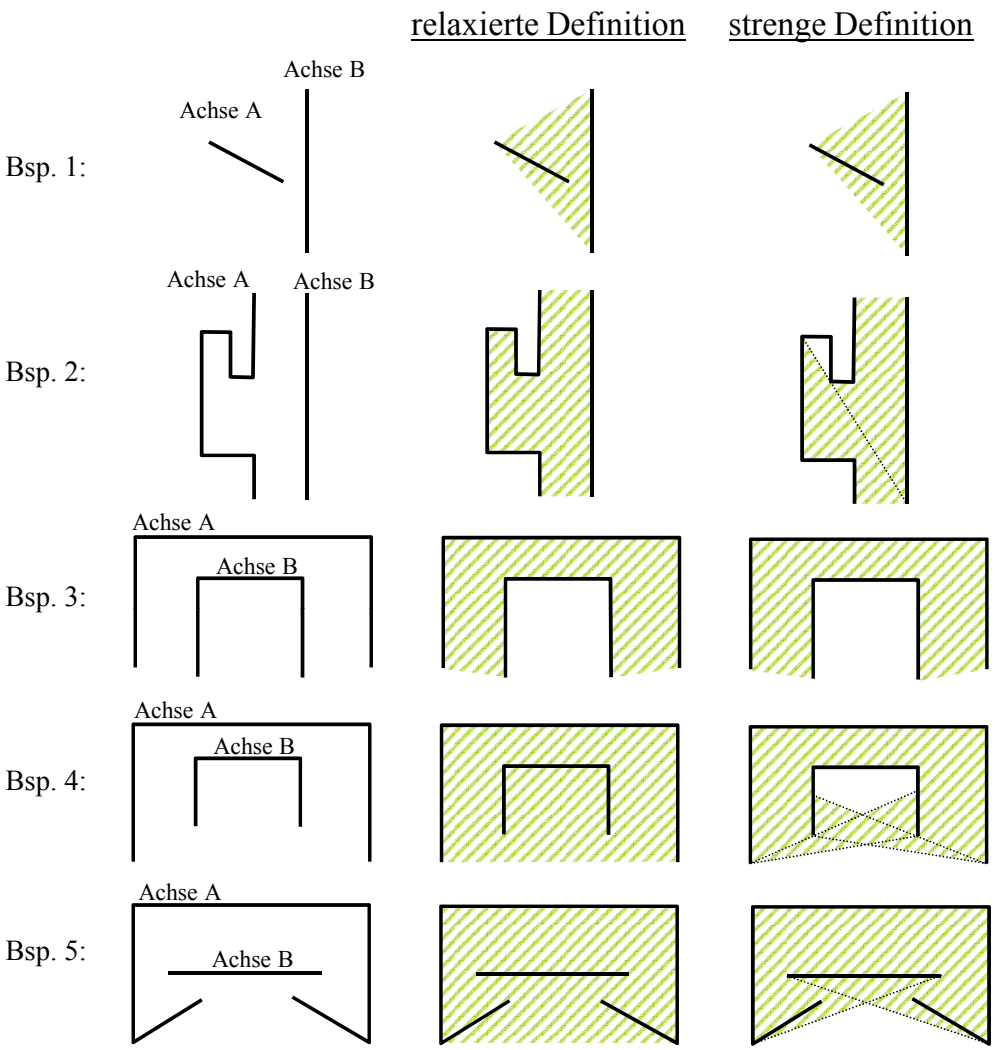


Abbildung 4.22: Vergleich der beiden Definitionen des Zwischenbereichs

Rein von der Definition her erscheint zunächst die strenge Definition dem intuitiven Verständnis am nächsten. Dabei wird davon ausgegangen, dass ein Punkt zwischen zwei Achsen liegt, wenn beim Blick entlang einer Geraden von diesem Punkt aus beide Achsen sichtbar sind. Dieses kann allerdings zu Abschattungsbereichen führen (Abbildung 4.20 rechts), die im Sinne der Anwendung von Achsenanfragen für die Terminplanung in der Regel unerwünscht sind. Durch die relaxierte Definition wird dieses vermieden. Der Zwischenbereich umfließt hierbei Sichthindernisse. Doch auch bei dieser Definition entstehen zum Teil nicht der intuitiven Erwartung entsprechende Ergebnisse. In Abbildung 4.22 ist ein Vergleich der durch die beiden Definitionen erzeugten Zwischenbereiche für einige Beispiele gezeigt. Die algorithmische Berechnung des durch die strenge Definition bestimmten Zwischenbereichs stellt sich zunächst als schwieriger dar, da die Grenzlinien zwischen Abschattungsbereichen und Zwischenbereichen zu bestimmen sind. Die Problemstellung erfordert somit die Berechnung des gegenseitigen Sichtbarkeitsbereiches zwischen zwei Streckenzügen und gehört damit zur Gruppe der Sichtbarkeitsprobleme innerhalb des Fachgebiets der Computergeometrie ([Ghosh 2007], [de Berg u. a. 2008]). Aufbauend auf bereits existierenden Algorithmen in diesem

Bereich wurde ein Algorithmus zur Lösung dieses Problems in optimaler, linearer Zeit gefunden und in [Langetepe u. a. 2009] veröffentlicht. D.h., der Rechenaufwand dieses Algorithmus beträgt $O(n)$, wobei n der Anzahl der in den beiden Achsen A und B enthaltenen Knoten entspricht.

Die Berechnung des Zwischenbereichs der relaxierten Definition kann in drei Schritten erfolgen:

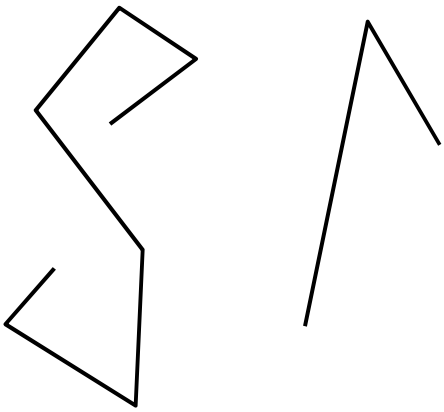
1. Berechnung der konvexen Hülle aller Punkte von Achse A und B,
2. eingeschränkte Triangulation des Inneren der konvexen Hülle unter Berücksichtigung der Strecken von A und B als Zwangskanten,
3. Entfernen aller Dreiecke in Bereichen, die nur durch eine Achse und eine Kante der konvexen Hülle begrenzt werden (Außentaschen).

Die Berechnung der konvexen Hülle kann auf einfache Weise nach [Andrew 1979][31] erfolgen. Eine ausführliche Beschreibung weiterer Algorithmen zur Bestimmung der konvexen Hülle findet man in [O'Rourke 1998]
Die Knoten und Kanten der Achsen A und B sowie die Kanten der konvexen Hülle bilden zusammen einen *Planar Straight Line Graph (PSLG)*. Die Triangulation eines PSLG kann nach der Kettenmethode aus [Preparata u. Shamos 1985] erfolgen. Die zugehörigen Algorithmen sind auch in [Chen 1996] dargestellt. Hierbei wird eine Triangulation ausschließlich durch das Hinzufügen von nicht kreuzenden Kanten zwischen existierenden Knoten des PSLG erreicht. Der Algorithmus erfolgt nach folgenden Schritten:

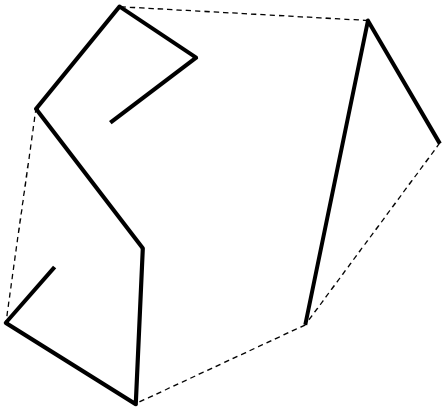
1. Regularisierung des PSLG,
2. Überführung des PSLG in eine Menge monotoner Ketten (Streckenzüge),
3. Triangulation des Zwischenbereichs je zweier Ketten.

Bei der Regularisierung werden im PSLG Kanten so ergänzt, dass bis auf die beiden Extrempunkte in der Betrachtungsrichtung für alle Punkte mindestens eine eingehende und eine ausgehende Kante existiert. Dem in Betrachtungsrichtung kleinsten Punkt ist mindestens eine ausgehende Kante zugeordnet, dem größten mindestens eine eingehende Kante. Aus dem so regularisierten PSLG kann anschließend eine Menge monotoner, sich nicht kreuzender Streckenzüge zwischen den beiden Extrempunkten so gebildet werden, dass jede Kante des PSLG mindestens Teil einer Kette ist. Hierzu wird das Flussprinzip angewandt, wonach an jedem Nichtextrempunkt die Summe der eingehenden Kanten gleich der Summe der ausgehenden Kanten sein muss. Eine Kante kann dabei Teil mehrerer Ketten sein (siehe Abbildung 4.23). Aufgrund der Monotonie der erzeugten Ketten bilden jeweils zwei benachbarte Ketten ein monotones Polygon, das auf einfache Art trianguliert werden kann ([de Berg u. a. 2008]).
Nachdem das gesamte Innere der konvexen Hülle auf diese Weise trianguliert wurde, sind die Dreiecke in den Außentaschen zu entfernen. Eine Außentasche ist ein Fläche,

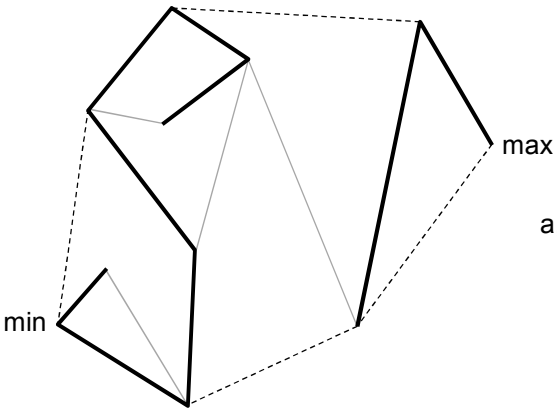
[31] entsprechenden Java-Quellcode findet man in [Brinkhoff 2005]



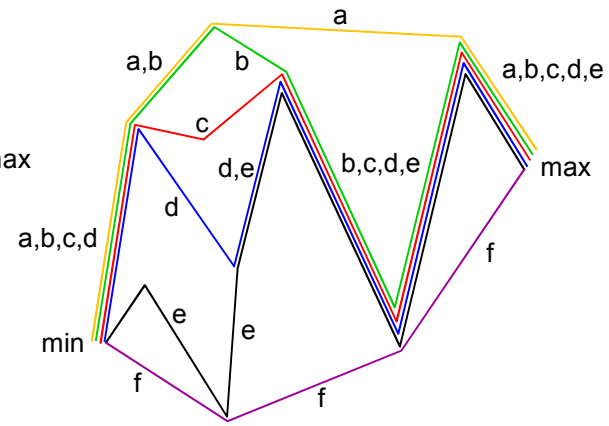
Achsen A und B



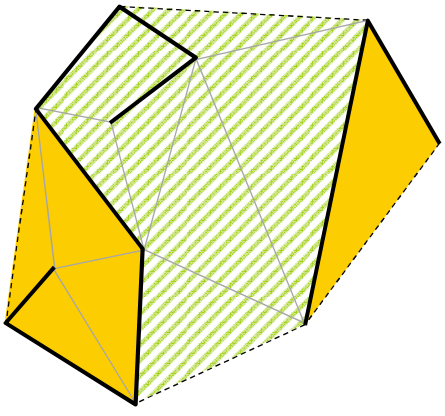
Achsen und konvexe Hülle



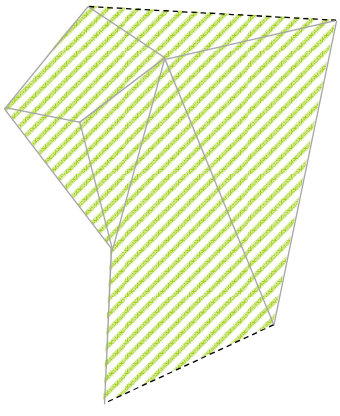
regularisierter PSLG



Ketten a-f



triangulierter PSLG
mit zwei Außentaschen



Zwischenbereich nach
relaxierter Definition

Abbildung 4.23: Schritte der Zwischenbereichsbestimmung nach der relaxierten Definition

die, außer von einer Kante der konvexen Hülle, nur von Kanten einer der beiden Achsen A oder B begrenzt wird und im Inneren keine Kante der jeweils anderen Achse enthält. Außentaschen beginnen stets an einer Kante der konvexen Hülle, die zwei Punkte der selben Achse verbindet, jedoch nicht selbst Teil einer der beiden Achsen ist. Entsprechende Kanten können bereits während der Erzeugung der konvexen Hülle vermerkt werden (*markedHullEdges*). Ausgehend von solchen Kanten werden Außentaschen durch das rekursive Voranschreiten über benachbarte Dreiecke bestimmt, ohne dabei die Begrenzung durch eine Achskante zu überschreiten. Wird eine Kante der anderen Achse angetroffen, so handelt es sich nicht um eine Außentasche. Die Dreiecke werden nicht entfernt. Beispielsweise stellt der Zwischenbereich in Beispiel 4 von Abbildung 4.22 auf Seite 85 keine Außentasche dar, obgleich eine Kante der konvexen Hülle existiert, die zwei Punkte der gleichen Achse verbindet. Dreiecke erkannter Außentaschen werden entfernt. Übrig bleiben Dreiecke, die den Zwischenbereich gemäß der relaxierten Definition beschreiben. Algorithmus 4.5 stellt das Vorgehen beim Entfernen der Außentaschen dar.

Algorithmus 4.5: Entfernen von Außentaschen

```
1  method removePocketEdges(Set markedHullEdges)
2  begin
3      Set foundAxes
4      Set pocketTriangles
5      forall the edges e in markedHullEdges do
6          triangle a = adjacent triangle of e
7          recursivelyExplorePocket(a, foundAxes, pocketTriangles)
8          forall the triangles t in pocketTriangles do
9              remove t
10         clear foundAxes
11         clear pocketTriangles
12 method recursivelyExplorePocket(triangle t, Set foundAxes, Set
   pocketTriangles)
13 begin
14     add t into pocketTriangles
15     forall the edges e of t do
16         if e is part of any axis x then
17             if x is not in foundAxes then
18                 add x into foundAxes
19                 if size of foundAxes > 1 then
20                     clear pocketTriangles
21                     return
22         else
23             forall the adjacent triangles a of e do
24                 if a is not in pocketTriangles then
25                     recursivelyExplorePocket(a, foundAxes, pocketTriangles)
```

Der anfallende Berechnungsaufwand hängt von der tatsächlichen Wahl der verwendeten Einzelalgorithmen ab und ist maßgeblich durch die Berechnung der konvexen Hülle und der Triangulation des PSLG bestimmt. Bei den angegebenen Algorithmen beträgt der Gesamtaufwand $O(n\ log\ n)$.
Das vorgestellte Konzept für die Beantwortung von Achsenanfragen beschränkt sich auf die Auswertung der Lage eines Objektes bezüglich genau zweier Achsen. Praktisch relevant ist allerdings auch die Anfrage, ob ein Objekt innerhalb einer Rasterzelle, gebildet aus zwei sich kreuzenden Achsenpaaren, liegt. Innerhalb der Sprache kann dieses durch die Schachtelung zweier Mengenausdrücke mit Achsanfragen für je ein Achsenpaar realisiert werden:

```
> #Macro1(p1,p2,p3):= {betweenAxisRelation(p1,{"p2","p3"}):-> ID}
> #Macro2(p1,p2,p3,p4,p5):= #Macro1(#Macro1(p1,p2,p3),p4,p5)
> #Macro2("{C:-> ID}","Achse_1","Achse_2","Achse_A","Achse_B")
```

Für die Terminplanung von Interesse sind zusätzlich Anfragen der Art: „alle Stützen *in* Achse A“. Anfragen dieser Art wurden im Rahmen der vorliegenden Arbeit nicht weiter betrachtet. Sie können jedoch mittels einer weiteren Funktion (*setCreationFunction*) in die Sprache aufgenommen werden. Algorithmisch entspricht eine solche Anfrage der Prüfung auf Schnitt zwischen der angegebenen Achse und den geometrischen Objekten des Modells. Die Achse stellt dabei nach ihrer Extrusion eine aus Dreiecken beschriebene Fläche im 3D dar, so dass die Problemstellung auf das Prüfen von Dreieckschnitten nach [Möller 1997] zurückgeführt werden kann.

4.7.3 Objektteilung

Ergibt eine Achsen- bzw. Zonenanfrage, dass ein Bauteilobjekt nur teilweise innerhalb des Anfragebereichs liegt, so ist das Objekt in Teilobjekte zu zerlegen, die eindeutig innerhalb bzw. außerhalb des Anfragebereichs liegen. Dabei ist zu beachten, dass je nach Form von Anfragebereich und Objekt mehrere verschiedene Objektteile entstehen können (Abbildung 4.24).
Kern der verwendeten Methodik zur Zerlegung der Geometrie ist der in [Hubbard 1990] veröffentlichte Algorithmus für die Durchführung regularisierter boolescher Operationen zwischen zwei durch triangulierte Oberflächen beschriebene Objekte A und B. Dieser stellt eine Verbesserung und Spezialisierung des ursprünglich in [Laidlaw

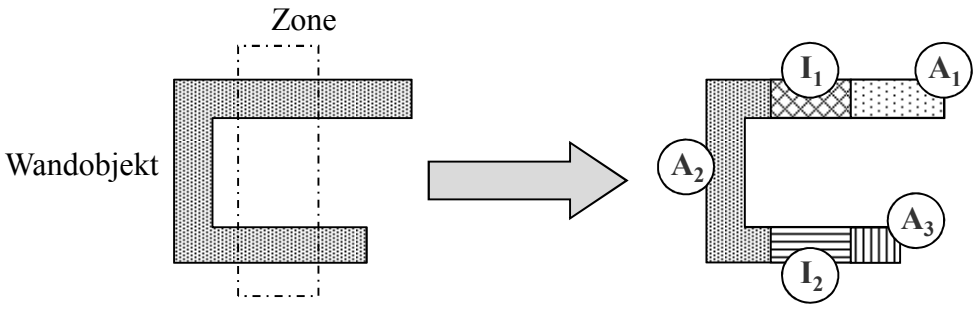


Abbildung 4.24: Zerlegung eines Wandobjektes in fünf Einzelobjekte; zwei innerhalb und drei außerhalb der Anfragezone

Operation	Dreiecke von Objekt A bzgl. B: innerhalb	außerhalb	gleich	entgegeng.	Dreiecke von Objekt B bzgl. A: innerhalb	außerhalb	gleich	entgegeng.
$A \cup B$	nein	ja	ja	nein	nein	ja	nein	nein
$A \cap B$	ja	nein	ja	nein	ja	nein	nein	nein
$A - B$	nein	ja	nein	ja	ja*	nein	nein	nein

*jedoch invertiert

Tabelle 4.7: Dreiecksauswahl für die Zusammensetzung der Geometrie des Ergebnisobjektes einer booleschen Operation nach [Hubbard 1990]

u. a. 1986] veröffentlichten Algorithmus für B-rep-Geometriebeschreibungen auf Basis konvexer Polygone dar. Ein ähnliches Vorgehen wurde in [Qu u. a. 2005] veröffentlicht. Der Algorithmus umfasst drei wesentliche Schritte:

1. Herstellen der Kompatibilität *beider* Oberflächennetze der beiden an der booleschen Operation beteiligten Objekte durch Einbeziehung der Schnittkanten in die Netzgenerierung.
2. Klassifikation aller Dreiecke der beiden Objekte je nach ihrer relativen Lage zum jeweils anderen Objekt als *innerhalb*, *außerhalb* oder auf dessen Oberfläche liegend. Im letztgenannten Fall wird zusätzlich zwischen einer *gleich* orientierten und einer *entgegengesetzt* orientierten Flächennormale unterschieden.
3. Zusammensetzen der Geometrie des Ergebnisobjektes. Hierbei werden je nach verwendeter boolescher Operation nur Dreiecke von Objekt A bzw. B verwendet, die Klassifkationen gemäß Tabelle 4.7 aufweisen.

Für den Algorithmus nach [Laidlaw u. a. 1986] ergibt sich ein Rechenaufwand von $O(V^2 + P^2)$ wobei V die Anzahl der Knoten und P die Anzahl der Dreiecke im endgültigen Oberflächennetz beider Objekte ist. Eine detaillierte Analyse des Gesamtrechenaufwands erfolgt in [Vanecek u. a. 1992]. Für den hier verwendeten verbesserten Algorithmus wird in [Hubbard 1990] eine Beschleunigung der Rechenzeit um einen Faktor zwischen 7 und 78 angegeben. Die Schritte 1. und 2. des Algorithmus stellen den wesentlichen Berechnungsaufwand dar. Sie nutzen die in den Kapiteln 4.7.1 und 4.7.2 bereits vorgestellten Algorithmen für die Verschneidung von Dreiecken nach [Möller 1997], die eingeschränkte Triangulation nach [Preparata u. Shamos 1985] zur Neugenerierung der Oberflächennetze unter Berücksichtigung der Schnittkanten sowie den Punkt-in-Polyeder-Test aus [O'Rourke 1998] zur Klassifikation von Dreiecken anhand der Lage ihres Mittelpunktes. Für die drei betrachteten booleschen Operationen Vereinigung, Durchschnitt und Differenz sind die beiden ersten Schritte in gleicher Weise durchzuführen. Die booleschen Operationen unterscheiden sich lediglich im 3. Schritt, der Auswahl der Dreiecke für die Beschreibung des Ergebnisobjektes.
Im Rahmen der Durchführung einer Objektzerlegung werden zwei boolesche Operationen für dasselbe Objektpaar durchgeführt. Das Ergebnis der Bildung des Durchschnitts zwischen dem zu teilenden Bauteilobjekt und dem Zonenobjekt beschreibt das Bauteilvolumen innerhalb der Zone. Das Ergebnis der Differenz zwischen Bauteilobjekt und

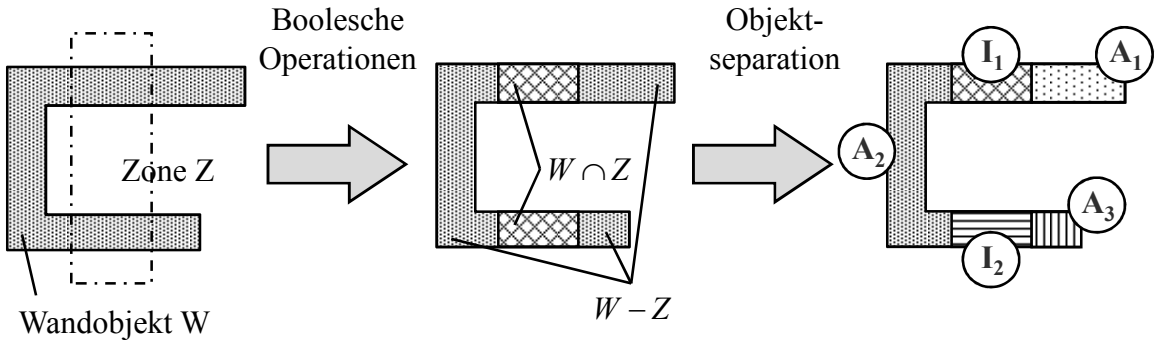


Abbildung 4.25: Objektzerlegung mittels booleschen Operationen und anschließender Separation nicht zusammenhängender Volumenteile

Zone ergibt hingegen das Bauteilvolumen außerhalb der Zone. Zur Durchführung beider boolescher Operationen brauchen die ersten beiden Schritte des Algorithmus nur einmal ausgeführt werden.

Als Ergebnis einer booleschen Operation wird jedoch stets ein Objekt erzeugt, das das gesamte aus der Operation resultierende Volumen beschreibt, selbst wenn dieses nicht zusammenhängend ist. Dieses entspricht nicht dem erwünschten Ergebnis im betrachteten Anwendungsfall. Um Einzelobjekte für jeden separaten Volumenteil zu erhalten, ist im Nachgang eine Objektseparierung vorzunehmen (siehe Abbildung 4.25).

Hierfür müssen nichtzusammenhängende Oberflächennetze erkannt und in Einzelobjekte überführt werden. Dieses Problem wird mittels einer booleschen Wegalgebra ([Pahl u. Damrath 2000]) gelöst. Da in einer B-rep-Geometriebeschreibung in der Regel nicht die direkte Nachbarschaft von Dreiecken entlang ihrer Kanten, sondern nur die Zugehörigkeit von Knoten zu Dreiecken explizit hinterlegt ist, wird davon ausgegangen, dass Dreiecke des Oberflächennetzes nur in ihren Knoten miteinander verbunden sind. Das Oberflächennetz kann dann wie folgt auf einen bipartiten Graphen $G(V_1, V_2; R_1, R_2)$ abgebildet werden (siehe Abbildung 4.26):

Knoten des Oberflächennetzes	$\rightarrow$	Knotenmenge V_1
Dreiecke des Oberflächennetzes	$\rightarrow$	Knotenmenge V_2
Verbindungen zwischen Knoten und Dreiecken	$\rightarrow$	Relation R_1 (Kanten)
Verbindungen zwischen Dreiecken und Knoten	$\rightarrow$	Relation R_2 (Kanten)

Der mit G assoziierte schlichte Graph $G_s(V; R)$ mit $V = V_1 \cup V_2$ und $R = R_1 \sqcup R_2$ wird durch die quadratische boolesche Matrix $\overline{R}$ der Dimension $n+m$ beschrieben, wobei n der Anzahl der Knoten und m der Anzahl der Dreiecke im gesamten Oberflächennetz entspricht. $\overline{R}$ ist symmetrisch und weist im Matrixschema eine charakteristische Struktur auf. Die Untermatrizen für die Knoten-Knoten- sowie Dreieck-Dreieck-Relationen sind Nullmatrizen. Für die anderen beiden Untermatrizen gilt aufgrund der Symmetrie $\overline{R}_2 = \overline{R}_1{}^T$ (Abbildung 4.27). Eine Spalte von $\overline{R_1}$ beschreibt die Verbindung zwischen einem Dreieck und den zugehörigen Knoten. Sie enthält daher an genau drei Stellen den Wert wahr. Bildet man die Hülle $\overline{H}_R$ von $\overline{R}$,

$$\overline{H}_R = \overline{R} \cup \overline{R}^2 \cup \overline{R}^3 \cup \overline{R}^4 \cup \ldots \,, \tag{4.42}$$

so beschreibt diese, für welche Knoten $i, k \in V$ des Graphen mindestens ein Weg beliebiger Länge von i nach k existiert. Das zugehörige Berechnungsschema in Abbildung 4.27

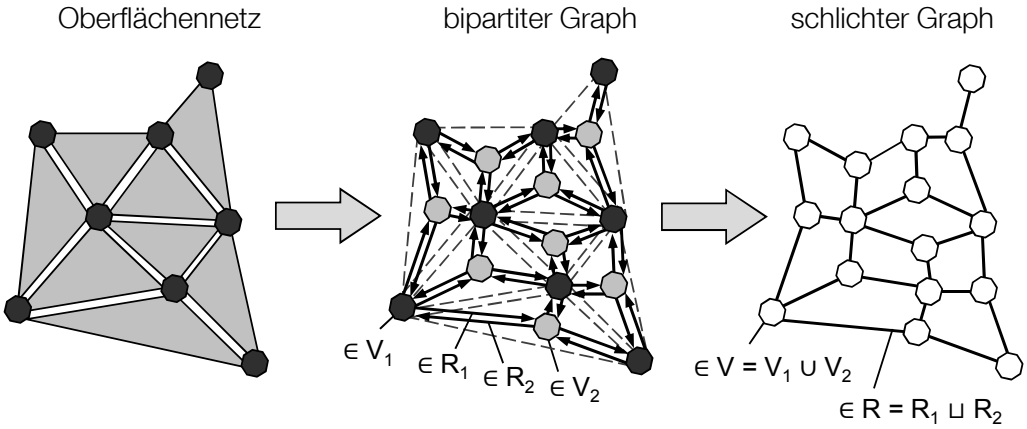


Abbildung 4.26: Abbildung eines in den Knoten verbundenen Oberflächennetzes auf einen Graphen

zeigt, dass die relevante Untermatrix, welche die Existenz von Wegen zwischen Dreiecken beschreibt, sich nur für gerade Potenzen von $\overline{R}$ von der Nullmatrix unterscheidet. Ihre Belegung in der Hülle kann somit direkt wie folgt berechnet werden:

$$\overline{D} = \overline{R_1}^T \, \overline{R_1} \tag{4.43}$$

$$\overline{H_D} = \overline{D} \cup \overline{D}^2 \cup \overline{D}^3 \cup \overline{D}^4 \cup \ldots \tag{4.44}$$

Die Reihenfolge der Nennung von Dreiecken in $\overline{R}_1$ kann so gewählt werden, dass $\overline{H}_D$ den Wert wahr nur in quadratischen Untermatrizen symmetrisch zur Hauptdiagonalen enthält (Abbildung 4.28). Dieses bedeutet, dass zwischen Dreiecken unterschiedlicher

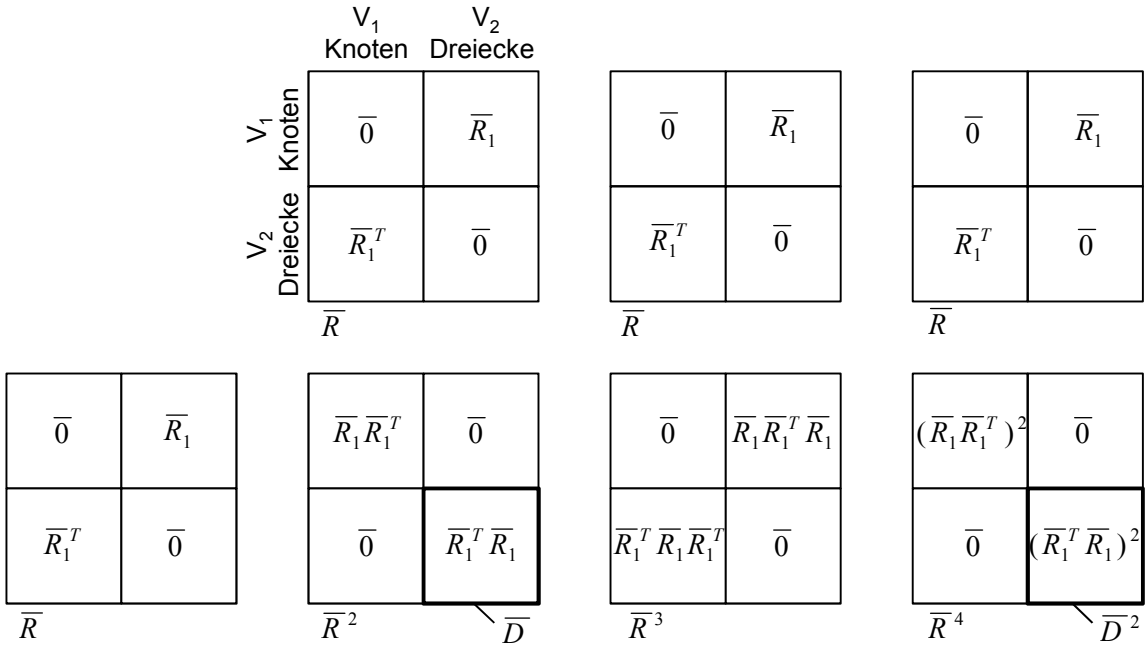


Abbildung 4.27: Berechnung der Hülle des schlichten Graphen G_s

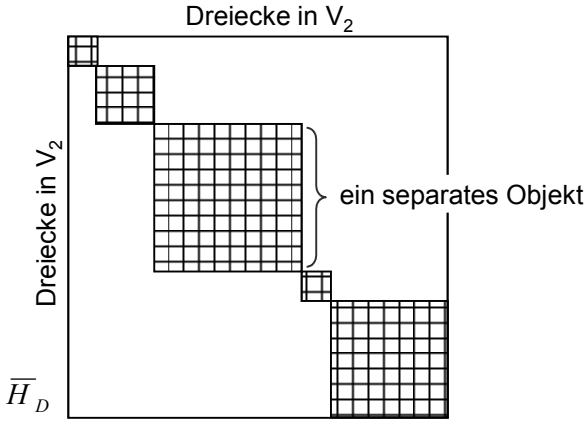


Abbildung 4.28: Separate Objekte in der Untermatrix $\overline{H}_D$ der Hülle von $\overline{R}$

Untermatrizen keine Wege existieren. Somit kennzeichnen die Untermatrizen Oberflächennetze separater Objektteile. Sämtliche Dreiecke eines Objektteils können einer Spalte von $\overline{H}_D$ entnommen werden.
Algorithmus 4.6 beschreibt das Vorgehen bei der Objektseparierung. Die Berechnung der Hülle stellt den Hauptrechenaufwand dar. Sie erfolgt nach dem Algorithmus von Warshall ([Thulasiraman u. Swamy 1992]) mit einem Aufwand der Größenordnung $O(n^3)$.

Probleme können entstehen, wenn die beschriebene Methode der Objektteilung auf Objekte angewandt wird, die schon vorher aus mehreren getrennten Oberflächennetzen bestanden. Diese werden, unabhängig davon, ob sie geschnitten wurden, in getrennte Objekte aufgelöst, welches unerwünscht ist. Ist z.B. eine Wand zur Darstellung des Wandaufbaus durch mehrere parallele Kuben repräsentiert und wird nur eine Schicht durch die Anfragezone geschnitten, so werden auch für die anderen Schichten sepa-

Algorithmus 4.6: Separierung von getrennten Oberflächennetzen

```
1   Set separatedObjects
2   Set processedTriangles
3   boolean matrix R1, D, HD
4   R1 = builtR1(triangles)
5   D = calculateD(R1)
6   HD = calculateHull(D)
7   forall the columns c of HD do
8       if processedTriangles does not contain c then
9           Set objectMesh
10          forall the rows r of HD do
11              if HD[r][c] = true then
12                  add r to objectMesh
13                  add r to processedTriangles
14          add objectMesh to separatedObjects
15  return separatedObjects
```

rate Wandobjekte erzeugt. Um diese Problematik zu umgehen, sollten Objekte, für die später eine Teilung vorgenommen werden soll, nur aus einem zusammenhängenden Oberflächennetz modelliert werden und z.B. der Schichtenaufbau nur als Attribut hinterlegt werden.
Neben der Aufteilung der Geometrie ist für den Terminplanungsprozess auch eine Aufteilung der zugehörigen Mengeninformationen erforderlich. Dieses kann durch eine erneute Ausführung einer automatisierten Mengenermittlung für die erzeugten Objektteile erfolgen. Hierbei werden die gleichen Berechnungsregeln verwendet wie für die Ausgangsobjekte. Um eine Prozessschleife zu vermeiden, ist eine entsprechende Funktionalität in die Endanwendung des Terminplaners zu integrieren oder ein Service einer spezialisierten Software zu nutzen. Diese Problematik wird im Rahmen dieser Arbeit nicht weiter betrachtet.
Eine Aufteilung von Objektattributen erscheint nicht erforderlich. Vielmehr verbleiben die Orginalobjkete im Modell und werden von den Teilobjekten referenziert. Somit ist ein Zugriff auf deren Informationen wie Schichtenaufbau, Material usw. stets möglich.

4.8 Versionierung

Heute übliche Planungssoftware bietet in der Regel keine Unterstützung für das Aufzeichnen einer Änderungshistorie auf Objektebene. Außerdem weist das für die Datenkommunikation zwischen Projektpartnern verwendete neutrale Datenformat IFC mit dem „OwnerHistory"-Konzept ([Liebich u. a. 2004]) nur eine sehr eingeschränkte Unterstützung für den Austausch von Änderungsinformationen auf. Um dennoch eine Versionierung zu ermöglichen, hat diese nachträglich zu erfolgen, und zwar zum Zeitpunkt des Datenimports in den Workspace, auf Basis eines detaillierten Vergleichs zwischen den importierten und den bereits vorhandenen Daten des letzten Imports. Erkannte Unterschiede entsprechen somit den *kumulierten* Änderungen zwischen zwei Datenimporten. Eine Verwaltung der Änderungshistorie mit allen Zwischenschritten ist auf diese Weise nicht möglich. Diese ist jedoch insbesondere bei fremden Domaindaten auch nicht von Interesse. Bei eigenen Domaindaten kann die Detaillierung der aufgezeichneten Änderungshistorie durch die Wahl des Intervalls für die Speicherung in den Workspace eigenmächtig beeinflusst werden.
Um für die verschiedenen in der modellbasierten Terminplanung genutzten Teilmodelle unabhängige Aktualisierungszyklen zu unterstützen, erfolgt die Versionierung je Teilmodell getrennt. Dabei wurde aufbauend auf der Arbeit von [Vad 2007] ein Verfahren zum Speichern von Vorwärtsdeltas[32] gewählt. D.h., je Teilmodell Θ wird im Workspace sowohl ein Referenzdatenbestand mit den aktuellsten Versionen aller Objekte $O \in \Theta$ als auch eine Änderungshistorie gespeichert. Die Änderungshistorie wird durch mehrere miteinander in Beziehung stehende gerichtete Graphen beschrieben: Ein Modellversionsgraph beschreibt die Änderungen in der Zusammensetzung des Teilmodells aus Objektversionen, wohingegen ein Objektversionsgraph je Objekt die Änderungen in den Objektattributen beschreibt. Die jeweiligen Änderungen in den Modelldaten werden als Kantenbewertungen der gerichteten Graphen abgespeichert. Durch die Auswertung die-

[32] bzgl. verschiedener Verfahren der Versionierung siehe [Weise 2006]

ser Kantenbewertungen entlang von Pfaden in den Graphen ist eine Wiederherstellung früherer Versionsstände möglich.

Objektversion: Im Laufe der Planung wird der Zustand eines Objektes durch das Hinzufügen oder Löschen von Attributen bzw. das Anpassen von Attributwerten verändert. Jeder Zustand (Version) eines im Workspace gespeicherten Objektes O wird als eine Objektversion OV_i bezeichnet. Die Menge aller Versionen eines Objektes O wird mit V_O bezeichnet.

$$OV_i := \text{ Die } i\text{-te Version des Objekts } O \tag{4.45}$$

$$V_O := \{OV_i \mid i = 1, 2, \dots\} \tag{4.46}$$

Modellversion: Im Verlauf der Planung werden Objekte erzeugt, gelöscht oder deren Zustand geändert. Eine Modellversion MV_k ist eine mathematische Menge, die die Zusammensetzung des betrachteten Teilmodells Θ aus Objektversionen zum Zeitpunkt eines Datenimports k beschreibt. Die Menge aller Modellversionen eines Teilmodells Θ wird mit V_M bezeichnet.

$$MV_k := \{OV_i \mid O \in \Theta \ \wedge \ i \text{ die Version von } O \text{ zum Zeitpunkt des Datenimports } k \ \} \tag{4.47}$$

$$V_M := \{MV_k \mid k = 1, 2, \dots\} \tag{4.48}$$

Objektversionsgraph: Die binäre Relation RN_O auf der Menge der Objektversionen V_O beschreibt die logische Reihenfolge der im Worspace gespeicherten Zustände des Objektes O. Sie hat die Bedeutung "ist nachfolgende Objektversion von". Der zugehörige gerichtete Graph wird Objektversionsgraph genannt und mit G_O bezeichnet.

$$G_O := (V_O; RN_O) \qquad RN_O \subseteq V_O \times V_O \tag{4.49}$$

Für jedes Objekt O existiert genau ein Objekversionsgraph G_O. Für jedes Teilmodell Θ existieren n Objektversionsgraphen, wobei n der Anzahl aller jemals in den Workspace importierten Objekte $O \in \Theta$ entspricht.

Modellversionsgraph: Die binäre Relation RN_Θ auf der Menge der Modellversionen V_M beschreibt die logische Reihenfolge der in den Workspace importierten Zustände des Teilmodells Θ. Sie hat die Bedeutung "ist nachfolgende Modellversion von". Der zugehörige gerichtete Graph wird Modellversionsgraph genannt und mit G_Θ bezeichnet.

$$G_\Theta(V_M; RN_\Theta) \qquad RN_\Theta \subseteq V_M \times V_M \tag{4.50}$$

Für jedes Teilmodell Θ existiert genau ein Modellversionsgraph G_Θ.

Kantenbewertungen: Jede Kante eines Modellversionsgraphen erhält eine Menge von Bewertungen. Diese beschreibt die Anpassungen, die an einer Vorgängerversion MV_x vorgenommen werden müssen, um sie in die Nachfolgeversion MV_y zu überführen. Analog beschreiben die Bewertungen einer Kante zwischen zwei Objektversionen OV_x und OV_y die vorzunehmenden Anpassungen an einem Objekt im Zustand x, um es in den Zustand y zu überführen. Da es sich sowohl bei Modellversionen als auch

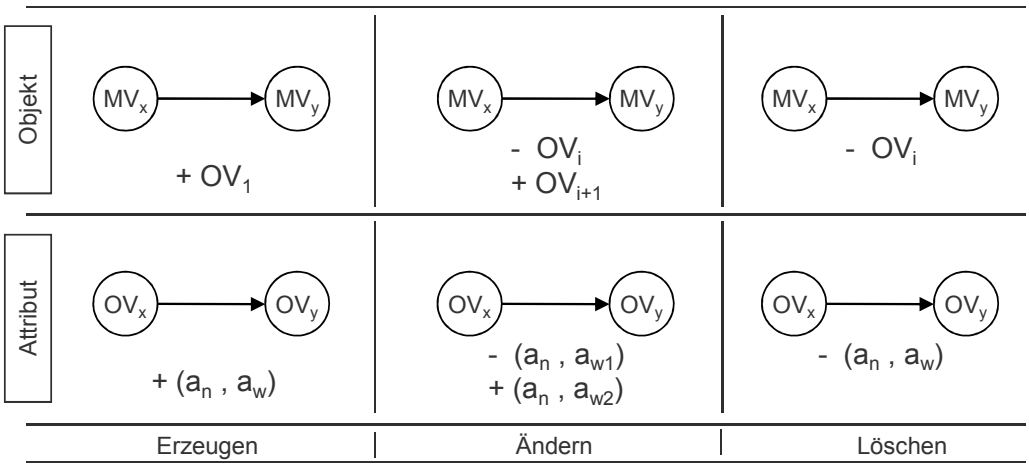


Abbildung 4.29: Mögliche Änderungen am Datenbestand und deren Erfassung durch Kantenbewertungen in Modell- und Objektversionsgraphen

bei Objekten um mathematische Mengen handelt, können die Anpassungen vollständig mittels der beiden Operationen *Hinzufügen (+)* und *Entfernen (-)* von Elementen beschrieben werden. Die Datenoperationen *Erzeugen*, *Ändern* und *Löschen* eines Objektes bzw. Attributes können gemäß Abbildung 4.29 auf diese beiden Grundoperationen zurückgeführt werden. Durch das Umdrehen der Vorzeichen der Bewertungen einer Kante kann ein Zurückrechnen von einer Nachfolgeversion auf die Vorgängerversion erfolgen.

Revision vs. Variante: Modell- und Objektversionsgraphen stellen Wurzelbäume dar. Die Modellversion, die beim ersten Datenimport des Teilmodells Θ in den Workspace erstellt wird, bildet die Wurzel von G_Θ. Wird ein weiterer Datenbestand des Teilmodells Θ in den Workspace importiert, so wird eine neue Modellversion MV_y in den Modellversionsgraphen G_Θ eingefügt. Im Allgemeinen ergeben sich hierbei zwei Möglichkeiten(Abbildung 4.30):

1. Der neue Datenbestand stellt eine *Revision* von Θ dar.
   In diesem Fall wird MV_y an einen Blattknoten MV_x von G_Θ angehängt.

2. Der neue Datenbestand stellt eine *Variante* von Θ dar.
   In diesem Fall wird MV_y an einen inneren Knoten MV_x von G_Θ angehängt.

Die Entscheidung, ob es sich bei den zu importierenden Daten um eine Revision oder um eine Variante von Θ handelt, ist vom Anwender durch die Wahl der als Vorgänger zu betrachtenden Modellversion MV_x vorzunehmen.
Beim Erstellen von Objektversionen ergibt sich eine ähnliche Situation. Eine Objektversion, die beim ersten Import eines Objekts O erstellt wird, bildet die Wurzel von G_O. Eine neu zu G_O hinzuzufügende Objektversion OV_y kann ebenfalls an einen Blattknoten oder einen inneren Knoten angehängt werden. Aufgrund der mitunter hohen Anzahl auftretender Objektversionen ist dieses jedoch automatisch zu entscheiden. Folgende Regel wird verwendet: Wenn die als Vorgänger gewählte Modellversion MV_x eine Objektversion OV_x von O enthält, dann wird OV_y an OV_x angehängt. Ansonsten kann OV_y an eine beliebige, existierende Objektversion OV_i angehängt werden.

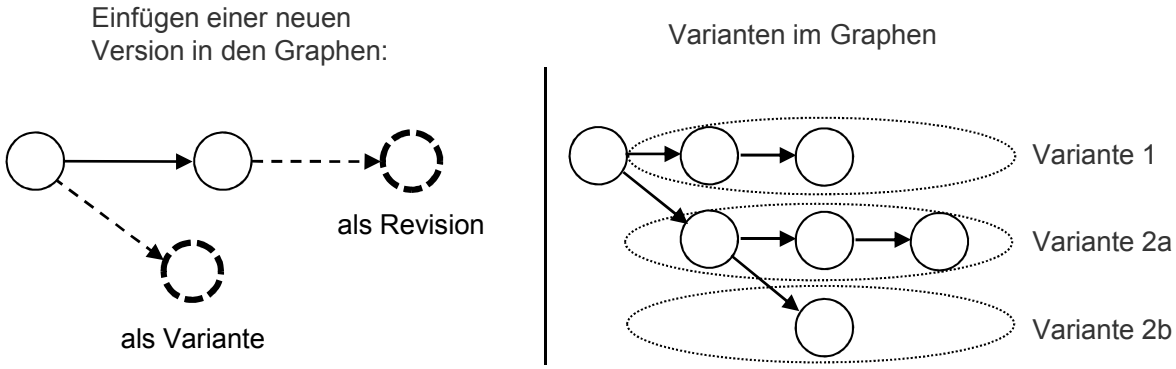


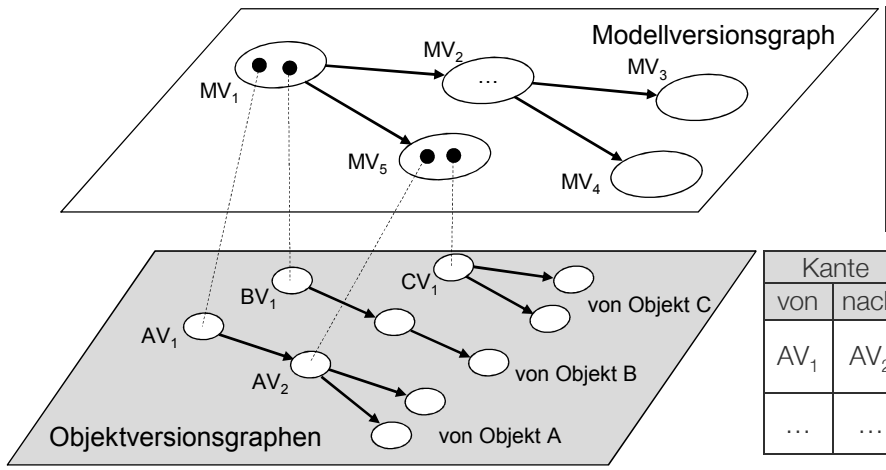
Abbildung 4.30: Baumstruktur von Modell- und Objektversionsgraphen infolge Berücksichtigung von Varianten

Wird beim Import eines neuen Datenbestandes keine Änderung gegenüber dem gewählten Vergleichsdatenbestand gefunden, so wird G_Θ trotzdem eine neue Modellversion MV_y hinzugefügt. Sie enthält die gleichen Objektversionen wie die vorhergehende Modellversion MV_x. Die Kante von MV_x nach MV_y erhält daher keine Bewertung und die Objektversionsgraphen bleiben unverändert. **Änderungshistorie:** Die Modell- und

Versionsgraphen bilden zusammen mit der Kantenbewertung die Änderungshistorie der in den Workspace importierten Daten. Der logische Zusammenhang der Graphen ist in Abbildung 4.31 verdeutlicht.

Referenzdatenbestand: Da die Änderungshistorie nur die Unterschiede zwischen den einzelnen Modellversionen beschreibt (Deltainformationen), ist zusätzlich für jedes Objekt ein Referenzdatenbestand vorzuhalten. D.h., für genau eine Objektversion OV_i ist der Zustand des Objektes O vollständig abzuspeichern. Es wird stets der zuletzt importierte Objektzustand als Referenzzustand gewählt, da auf diesen im Rahmen des Terminplanungsprozesses am häufigsten zugegriffen wird.

Pfad im Graphen: Ein Pfad P in einem gerichteten Graphen bezeichnet eine Folge von Kanten, bei denen der Endknoten der Kante i dem Anfangskonten der kante $i+1$ entspricht.



Kante		Bewertung	
von	nach	-	+
MV_1	MV_5	AV_1	AV_2
		BV_1	~
		~	CV_1
...	...		

Kante		Bewertung	
von	nach	-	+
AV_1	AV_2	...	
...	...		

Abbildung 4.31: Änderungshistorie eines Teilmodells

Kettung von Pfaden: Die Kettung zweier Pfade $P_1 \circ P_2$ ergibt sich aus dem Anfügen der Kantenfolge von P_2 an das Ende der Kantenfolge von P_1.

Restauration eines Teilmodellzustands: Soll ein früherer Zustand oder eine andere Variante eines Teilmodells, die durch die Modellversion MV_F repräsentiert wird, wieder hergestellt werden, so müssen alle Änderungen zwischen MV_F und dem Referenzdatenbestand, repräsentiert durch MV_R, rückgängig gemacht werden. Dieses kann durch das Ausführen aller Kantenbewertungen entlang des Pfades von MV_R nach MV_F im Modellversionsgraphen G_Θ und den korrespondierenden Pfaden in den Objektversionsgraphen erfolgen. Dabei ist ggf. die Umkehr des Vorzeichens der Bewertungen zu beachten und die Reihenfolge der Änderungen gemäß der Pfade einzuhalten. Die Reihenfolge der Änderungen einer Kantenbewertung ist unwesentlich.
Der Pfad zwischen zwei Modellversionen von MV_R nach MV_F kann wie folgt bestimmt werden:

1. Bestimmung des Pfades A von der Wurzel zu MV_R (bzw. umgekehrt).
2. Bestimmung des Pfades B von der Wurzel zu MV_F (bzw. umgekehrt).
3. Ausgehend von der Wurzel weisen die Pfade A und B n gemeinsame Kanten auf. Diese werden entfernt.
4. Invertieren des Pfads A.
5. Der gesuchte Pfad ist $A \circ B$.

Die Bestimmung eines Pfades zwischen zwei Objektversionen in einem Objektversionsgraphen kann auf gleiche Weise erfolgen.

Konsistenz des Gesamtmodells: Ausgehend von einem konsistenten Gesamtmodell ergeben sich beim Import eines neuen Datenbestands für ein einzelnes Teilmodell in der Regel zunächst Inkonsistenzen im Gesamtmodell. D.h., teilmodellübergreifende Referenzen verweisen mitunter auf nicht mehr vorhandene bzw. geänderte Informationen (Objekte, Attribute). Außerdem können im neuen Datenbestand zusätzliche Informationen existieren, die bisher bei der Erstellung von Verknüpfungen nicht berücksichtigt wurden. Ebenso können im aktualisierten Teilmodell vorgefundenen Planungsänderungen eine generelle Umplanung in anderen Teilmodellen erforderlich machen. Um wieder einen konsistenten und sinnvollen Gesamtplanungsstand zu erhalten, sind daher die anderen Teilmodelle entsprechend zu aktualisieren. Die Aktualisierung kann nicht automatisch erfolgen, sondern ist vom Anwender durchzuführen. Er wird dabei wie folgt unterstützt:

- Die getrennte Versionierung der Teilmodelle erlaubt eine schrittweise, themen- bzw. domainbezogene Aktualisierung der Planungsdaten. Dieses dient der Übersicht und entspricht der Aufteilung des Planungsprozesses auf mehrere Akteure.
- Grundlage von Anpassungen sind die in bereits aktualisierten Teilmodellen vorgenommenen Änderungen. Da diese im Workspace automatisch identifiziert werden, können sie zielgerichtet visualisiert werden. Eine Bestimmung abhängiger Informationen kann gemäß [Hanff 2003] erfolgen

- Bei Verwendung von regelbasierten Verknüpfungen auf Basis der spezifizierten Sprache enthalten die Regeln bereits im gewissen Umfang die Intention des Planers. Daher ist oftmals eine reine Reinterpretation dieser Regeln und die anschließende Kontrolle der neuen Ergebnisse ausreichend.

Beispiel:
Wurden gegenüber einem früheren Entwurf im Architekturmodell in einem Bauwerksbereich die Anordnung von tragenden Stützen und Wänden sowie deren Bauteildicken und Materialien verändert, so wird dieses beim Import in den Workspace erkannt und kann in einem Viewer visualisiert werden. Für die betroffenen Elemente kann dann eine Aktualisierung der Mengenermittlung vorgenommen bzw. angefordert werden. Danach erfolgt ggf. die Anpassung von Zonen und Achsen, die die Bauabschnitte im betreffenden Bereich definieren. Da Vorgänge im Terminplan über Regeln mit Bauteilmengen und CAD-Objekten innerhalb der angepassten Bauabschnitte verknüpft sind, reicht die Reinterpretation dieser Regeln aus, um die Berechnung der Dauer der Vorgänge auf Basis der neuen Mengen und die Aktualisierung der 4D-Animation durchzuführen.

Infolge der sukzessiven Aktualisierung der einzelnen Teilmodelle ergeben sich im Workspace zwischenzeitlich inkonsistente Kombinationen von Teilmodellversionen. Erst nachdem alle Teilmodelle aktualisiert wurden, liegt im Workspace wieder ein konsistentes Gesamtmodell vor. Um bei der Restaurierung vorheriger Modellversionen direkt zwischen konsistenten Gesamtmodellzuständen wechseln zu können, sind die entsprechenden Kombination von Teilmodellversionen zu vermerken. Dieses ist vom Anwender zu veranlassen. Die Verwaltung der Historie von Kombinationen aus Teilmodellen, die ein konsistentes Gesamtmodell ergeben, erfolgt in einem eigenen Modellversionsgraphen. Hierbei ist ebenfalls die Berücksichtigung von Modellvarianten möglich.

Versionsbezug in der Verknüpfungssprache: Die in Kapitel 4.6 spezifizierte Verknüpfungssprache ist vom Problem der Versionierung entkoppelt. Die damit erstellten Regeln beziehen sich stets auf die aktuell geladenen Teilmodellversionen. Das Laden bzw. die Verwaltung der einzelnen Teilmodellversionen hat durch einen Versionsmanager innerhalb des Workspaces zu erfolgen.

4.9 Datenaustausch mittels IFC

Die Basis für einen reibungslosen Datenaustausch zwischen verschiedenen Projektbeteiligten und unterschiedlichen Fachanwendungen über einen gemeinsamen Projektdatensever stellt die Verwendung eines einheitlichen Datenformats dar. Aus den in Kapitel 3.2 genannten Gründen wurde hierfür ein Austausch auf Basis von IFC gewählt. Für die im Rahmen der modellbasierten Terminplanung relevanten Daten gemäß Kapitel 4.5 wurde daher untersucht, wie diese in IFC abgebildet werden können. Dabei wurde festgestellt, dass eine Abbildung mit der derzeit aktuellen IFC Version 2x3 bereits größtenteils möglich ist. Für identifizierte Defizite wurde zusätzlich die Version 2x4, die bereits als beta-Version verfügbar ist[33], auf Lösungen untersucht. Die nächsten Kapitel geben getrennt für die einzelnen Teilmodelle einen Überblick über die

[33] http://www.iai-tech.org/groups/msg-members/news/ifc2x4-beta2-available

gewählte Abbildung. Dabei wird, wo vorhanden, auf existierende Arbeiten verwiesen. Anhang C enthält für einige Aspekte zusätzliche Diagramme, die die Datenabbildung in IFC verdeutlichen. Eine detaillierte Zusammenfassung der vorgenommenen Analyse, teilweise mit Beispieldaten, findet man in [Weise u. a. 2009]. Detailierte Informationen zum gesamten IFC-Datemodell kann man der html-Dokumentation[34] sowie dem Model Implementation Guide[34] der IFC-Version 2x2 entnehmen. Einen Überblick über viele Aspekte der Datenabbildung geben darüber hinaus auch [Karstila u. Serén 2004] und [Karstila u. Serén 2005].

4.9.1 CAD-Daten

Die Abbildung von dreidimensionalen CAD-Daten ist der Bereich, in dem die IFC bisher ihre weiteste Verbreitung gefunden haben. Zudem ist dieser Datenbereich Kern des IAI Zertifizierungsverfahren für IFC-Softwareschnittstellen[35]. Die Anforderungen der modellbasierten Terminplanung gemäß Kapitel 4.5 in diesem Bereich werden durch die IFC-Datenmodell bereits voll erfüllt. Probleme ergeben sich lediglich aus der oftmals trotz Zertifizierung mangelhaften Implementierung. Aufgrund der Komplexität wird nachfolgend auf eine detaillierte Darstellung der Abbildungsmöglichkeiten von CAD-Daten in IFC verzichtet und statt dessen die aus Sicht der Terminplanung wesentlichen Datenkonzepte umrissen. Ausführlichere Informationen findet man in den oben angegebenen Literaturquellen.

- Abweichend von der Modellbildung in Kapitel 4.4 wird in IFC der Objekttyp nicht primär durch ein Attribut bestimmt, sondern die Objekte, wie in der Softwaretechnik üblich, durch deren Zugehörigkeit zu Klassen typisiert. Ein Attribut *ObjectType* wird jedoch oftmals verwendet, um eine zusätzliche Verfeinerung der Typisierung vorzunehmen.
- Alle von `IfcRoot` abgeleiteteten Objekte haben eine global eindeutige ID, über die sie identifiziert werden.
- Sämtliche Bauteile eines Bauwerks stellen Subtypen von `IfcBuildingElement` dar, welches sich von `IfcRoot` ableitet. Sie haben somit ebenfalls eine eindeutige ID, weisen jedoch kaum Attribute zur Beschreibung designtechnischer Eigenschaften auf.
- Mittels eines Relationsobjektes vom Typ `IfcRelAssociatesMaterial` kann einem Bauteil ein Material oder ein Schichtenaufbau zugeordnet werden.
- Sollen Bauteilobjekte darüber hinausgehende Eigenschaften erhalten, so erfolgt deren Definition über ein `IfcPropertySet`, das einen Container für ein oder mehrere Eigenschaften darstellt. Dieses wird dem Bauteilobjekt über ein Relationsobjekt vom Typ `IfcRelDefinesByProperties` zugeordnet. Durch dieses Konzept

[34] http://www.buildingsmart.de/4/4_01.htm

[35] Gegenstand der Zertifizierung ist der sog. Extended Coordination View, der im Wesentlichen den Datenbereich der Architekturplanung umfasst (siehe auch www.ifcwiki.org/index.php/IFC-Developers).

wird eine ähnlich flexible Zuordnung von Eigenschaften zu einzelnen Objektinstanzen erreicht, wie in der Modellbildung nach Kapitel 4.4.

- Jedes Bauteil wird über ein Subtyp von `IfcObjectPlacement` positioniert. Dieses kann entweder über die Angabe von Koordinaten (möglicherweise innerhalb eines mehrfach geschachtelten lokalen Koordinatensystems) oder in Bezug zu einem Achsraster erfolgen.
- Die Darstellung von Bauteilen wird durch ein `IfcProductRepresentation`-Objekt beschrieben. Dieses kann mehrere alternative Representationen enthalten, z.B. je ein `IfcShapeRepresentation` für die geometrische Darstellung in 2D und 3D. Unter anderen stehen dabei folgende Konzepte für die Geometriebeschreibung zur Verfügung: BoundingBox, GeometricCurveSet, Brep, CSG, SweptSolid, Clipping.
- Die räumliche Gliederung eines Projektes in Baugrundstücke, Bauwerke, Geschosse und Räume wird durch Subtypen von `IfcSpatialStructureElement` abgebildet. In diese werden Bauteile über Relationsobjekte vom type `IfcRelContained-InSpatialStructure` eingeordnet.

Mehrere Objektgranularitäten: Die Einteilung des Bauwerks in Bauabschnitte erzwingt oftmals (insbesondere bei der Planung der Rohbauerstellung) eine von der im Architekturmodell abweichende Objektgranularität. Während eine gröbere Granularität durch die mögliche Adressierung (z.B. im Rahmen einer 4D-Simulation) mehrerer Objekte innerhalb eines Bauabschnitts unproblematisch ist, führt eine freie Festlegung von Bauabschnitten zur Teilung von Objekten (Abbildung 2.4 auf Seite 24). Die dabei entstehenden Teilobjekte werden als zusätzliche Objekte in das Modell aufgenommen. Die Originalobjekte verbleiben zur Vermeidung von Interferenzen mit anderen Fachdisziplinen im Modell. Werden mehrere Terminplanvarianten untersucht, denen unterschiedliche Einteilungen des Bauwerskmodells in Bauabschnitte zugrunde liegen, so ergibt sich für diese eine gemeinsame Objektgranularität nach dem Prinzip des größten gemeinsamen Teilers (siehe Abbildung 4.32). Somit ist die Abbildung einer zweiten Objektgranularität, auch bei Berücksichtigung mehrerer Terminplanvarianten, ausreichend. Diese kann, neben der Architektursicht, als zweite Sichtweise des selben Bauteils betrachtet werden.
Ein Konsens, wie verschiedene Sichtweisen innerhalb *eines* IFC-Modells abzubilden sind, existiert derzeit jedoch nicht. In [Rönneblad u. Olofsson 2003] wird im Kontext der Zerlegung eines Rohbaumodells in Fertigteile innerhalb der IFC-Version 2.0 ein Relationsobjekt vom Typ `IfcRelAssemblesElements` verwendet, um Teilobjekte dem Originalobjekt zuzuordnen. In [Karstila u. Serén 2004, S. 35/43] werden zwei Abbildungsmöglichkeiten vorgeschlagen. Die erste entspricht der zuvor genannten, jedoch wird das auch in IFC 2x3 existierende Relationsobjekt `IfcRelAggregates` verwendet. Bei dieser Relation verbleibt der Kritikpunkt, dass eine Abweichung des Typs der Teilobjekte von dem des Originalobjekts zulässig ist. Bei der zweiten vorgeschlagenen Abbildungsmöglichkeit werden die Objekte der beiden Sichten durch die Zuordnung verschiedener `IfcOwnerHistory`-Objekte, und damit verschiedener Besitzer, unterschieden. Nachteil hierbei ist, dass aufgrund der fehlenden Hierarchie ohne Kenntnis der Rolle der Besitzer nicht zwischen grober und feiner Sichtweise unterschieden werden kann.

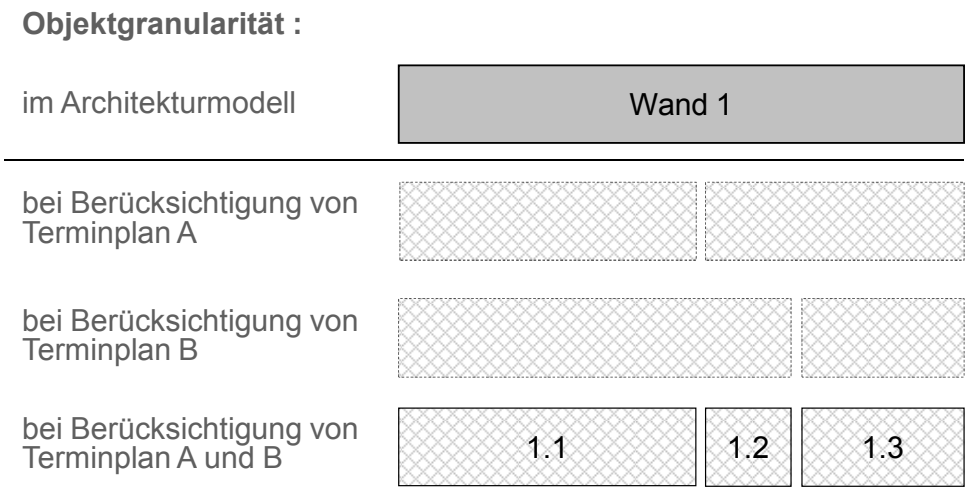


Abbildung 4.32: Bestimmung der Objektgranularität für die Terminplanung nach dem Prinzip des größten gemeinsamen Teilers

Im Rahmen dieser Arbeit wurde daher eine Zuordnung von Teilobjekten zum Originalobjekt des Architekturmodells mittels einem Relationsobjekt vom Typ `IfcRelNests` vorgenommen(siehe auch [Tulke u. a. 2008]). Dieses setzt einen einheitlichen Objekttyp für Original- und Teilobjekte voraus. Bzgl. der Zerlegung von Objekten mittels `IfcRelAggregates` existieren innerhalb der Implementation Support Group (ISG)[36] drei Implementierungsvereinbarungen, die ggf. auch eine Abbildung mittels `IfcRelNests` betreffen. Sie dienen der Vereinfachung der Interpretation.

1. #CV-2x3-119: Die Geometrie eines zerlegten Objektes soll entweder für das Originalobjekt *oder* die Teilobjekte angegeben sein. Nicht jedoch für beide.
2. #CV-2x3-120: Materialinformationen sollen nur für die Teilobjekte hinterlegt werden.
3. #CV-2x3-121: Die Zerlegungstiefe soll maximal eins betragen. D.h. Teile eines zerlegten Elements dürfen nicht weiter zerlegt werden.

Während der hier verfolgte Anwendungsfall die letzte Implementierungsvereinbarung erfüllt, sind die ersten beiden problematisch. Eine Geometrie sollte im betrachteten Anwendungsfall für beide Granularitäten hinterlegt sein. Eine Modellierungs- bzw. Viewersoftware sollte hingegen, je nach Sichtweise des Anwenders, nur eine der beiden visualisieren. Materialinformationen werden zudem nicht vom Terminplaner festgelegt bzw. verändert und sollten daher dem Originalobjekt zugeordnet bleiben. Zwar stellen die Implementierungsvereinbarungen keine offiziellen Datenmodellierungsregeln innerhalb der IFC dar, sondern lediglich eine Übereinkunft von softwareimplementierenden Personen bzw. Firmen, jedoch ist bei einer Abweichung ggf. mit Darstellungsfehlern bei Verwendung fremder Software zu rechnen. Somit bedarf dieses weiterer Abstimmung innerhalb der ISG.

[36] www.iai.hm.edu

4.9.2 Mengeninformationen

Basismengen: Der deutsche IAI Arbeitskreis Modellbasierte Mengenermittlung hat eine Spezifikation zur Abbildung von Basismengen innerhalb der IFC erarbeitet ([Mengenermittlung 2007a], [Mengenermittlung 2007b]). Danach wird entsprechend der Relation B_4 aus Abbildung 4.13 auf Seite 64 einem Objekte des Bauwerksmodells mittels eines Relationsobjektes vom Typ `IfcRelDefinesByProperties` ein `IfcElementQuantity` zugeordnet. Letzteres Objekt ist ein Container für alle dem Bauteil zugeordneten Basismengen. Es hält daher eine Liste von Referenzen auf Objekte vom abstrakten Typ `IfcPhysicalQuantity`, die die eigentliche Mengeninformationen enthalten. Hierfür stehen Subtypen zur Definition von Länge, Fläche, Volumen, Stückzahl, Gewicht, Zeit und komplexeren Mengenwerten, die sich wiederum aus mehreren Einzelwerten zusammensetzen, zur Verfügung. Das Attribut *Name* des `IfcPhysicalQuantity` gibt an, um was für eine Menge es sich handelt, z.B. um eine Nettowandfläche (NetWallArea). In der oben genannten Spezifikation ist angegeben, welche Mengentypen je Architekturbauteil vordefiniert sind. Raumobjekten können ebenfalls Mengenwerte zugeordnet werden.
Aus Sicht der modellbasierten Terminplanung erweisen sich die damit vorhandenen Möglichkeiten zur Abbildung von geometrischen Mengen als ausreichend. Allerdings werden heutzutage nur von wenigen existierenden CAD-Softwarepaketen Mengeninformationen nach IFC exportiert[37]. Von den im Rahmen des IFC Anwenderhandbuchs[38] von der IAI zur Verfügung gestellten Testdateien enthält beispielsweise nur die aus dem CAD-System ArchiCAD entsprechende Basismengen.

LV-Mengen: Zur Abbildung einer LV-Struktur werden Konstrukte des Kostenmodells der IFC gemäß den Abbildungen C.1 bis C.3 im Anhang verwendet. Dabei entspricht ein `IfcCostSchedule` einem LV. Dessen Attribut *PredefinedType* wird zu .UNPRICEDBILLOFQUANTITIES. gesetzt, um zu kennzeichnen, dass es sich um eine strukturierte Auflistung von Mengen ohne Kosten handelt. Die einzelnen LV-Positionen werden durch Objekte vom Typ `IfcCostItem` abgebildet die mit einem Relationsobjekt vom Typ `IfcRelSchedulesCostItems` dem LV zugeordnet werden. Durch eine Schachtelung von `IfcCostItem`-Objekten mittels Relationsobjekten vom Typ `IfcRelNests` können Unterpositionen und einzelne Mengenbeiträge abgebildet werden.
Die Zuordnung von Mengenwerten zu den `IfcCostItems` erfolgt analog zu den Basismengen, d.h. über ein `IfcElementQuantity`-Objekt, das über ein `IfcRelDefniesByProperties` einem `IfcCostItem` zugeordnet wird. Zusätzlich kann zur Abbildung des räumlichen Bezugs eines Einzelmengenbeitrags (`IfcCostItem`) diesem über ein Relationsobjekt des Typs `IfcRelAssignsToControl` gemäß Relation B_4 aus Abbildung 4.13 auf Seite 64 ein Objekt des Bauwerksmodells zugeordnet werden.
Werden infolge einer räumlichen Anfrage CAD-Objekte geteilt, so sind durch eine spezialisierte Softwareanwendung auch die zugehörigen LV-Mengen entsprechend zu teilen. Abbildung C.4 im Anhang zeigt, wie das Ergebnis in IFC abgebildet werden kann. Der

[37] Wenn ein Export von Mengen nach IFC überhaupt unterstützt wird, so muss er in der Regel als Zusatzoption explizit angewählt werden. So z.B bei AutoDesk Revit 2010 und Graphisoft ArchiCAD 12.
[38] www.buildingsmart.de/2/2_02_01.htm

ursprüngliche Mengenwert (ein Sutbtyp von `IfcPhysicalQuantity`) verbleibt dabei als Summenwert im LV und wird durch den Wert SUM des Attributs *Description* gekennzeichnet. Zusätzlich werden die einzelnen Teilmengen hinzugefügt und deren Bezeichnung im Attribut *Description* hinterlegt.
Die für die modellbasierte Terminplanung erforderliche Übermittlung von LV-Mengen und deren Aufgliederung in Einzelbeiträge samt Bezug zu Bauteilobjekten im Modell ist somit unter Verwendung der IFC möglich. Es sei jedoch erwähnt, dass sich in Deutschland für den Datenaustausch im Bereich Ausschreibung, Vergabe, Abrechnung (AVA) bisher das GAEB-Datenformat durchgesetzt hat, wohingegen ein IFC-Export von heutigen AVA-Programmen nicht geboten wird. Aufgrund fehlender internationaler Bedeutung des GAEB-Datenformats und dessen Beschränkung auf einen speziellen Datenbereich innerhalb der Bauprojektabwicklung wird im Rahmen dieser Arbeit auf dieses Format jedoch nicht weiter eingegangen. Aufgrund der Kooperation[39] zwischen dem GAEB und der IAI ist zudem in Zukunft eine engere Verzahnung der beiden Datenmodelle zu erwarten. Weitere Informationen zum Kostenmodell der IFC findet man im Information Delivery Manual (IDM)[40] unter dem Thema Cost Modelling.

4.9.3 Kostenbudget

Die Abbildung der Struktur einer Kostenaufstellung erfolgt zunächst Analog zu der von LV-Mengen, d.h. mittels der Objekttypen `IfcCostSchedule`, `IfcRelSchedules-CostItems`, `IfcCostItem` und `IfcRelNests` (siehe Abbildung C.5 im Anhang). Die Eigenschaft *PredefinedType* des `IfcCostSchedule` wird allerdings zu .BUDGET. oder zu .COSTPLAN. gesetzt. Einer Position der Kostenaufstellung (`IfcCostItem`) wird über ein Relationsobjekt vom Typ `IfcRelAssociatesAppliedValue` der eigentliche Kostenwert vom Typ `IfcCostValue` zugeordnet.
Da das durch die Terminplanung als Ausgangsinformation genutzte Kostenbudget je Gewerk zu einem frühen Zeitpunkt erstellt wird, in dem noch keine Modelldaten bzw. detaillierte Mengen vorliegen, wird eine Verknüpfung derartiger Informationen mit dem Kostenbudget nicht weiter betrachtet. In [Staub-French 2000] wird weiterführend eine Kostenermittlung auf Basis eines Modells und deren Abbildung in IFC beschrieben.

4.9.4 Achsraster und Zonen

Achsen: Ebene Achsraster werden in IFC innerhalb eines eigenen Koordinatensystems, das beliebig im Raum positioniert werden kann, durch ein `IfcGrid` abgebildet. Dieses setzt sich aus mehreren benannten Achsen (`IfcGridAxis`) zusammen, die in bis zu drei Hauptrichtungen gruppiert werden. Die Kurvenform einer Achse kann aus einer der Subtypen von `IfcCurve` gewählt werden. Für polygonale Streckenzüge steht der Typ `IfcPolyline` zur Verfügung. Zusätzlich kann ein `IfcGrid` über ein Relationsobjekt vom Typ `IfcRelContainedInSpatialStructure` einem Element der räumlichen Gliederung des Modells zugewiesen werden. Dadurch kann ein Achsraster z.B. einem

[39] www.buildingsmart.de/pdf/pm_2008-05.pdf
[40] http://idm.buildingsmart.no/confluence/display/IDM/Home

bestimmten Geschoss zugeordnet werden. Die zugehörige Geschosshöhe kann dann als Grundlage für die Extrusion von Achsen bei der Auswertung geometrischer Anfragen genutzt werden.

Zonen: Bauabschnitte des Rohbaus entsprechen oftmals nicht der räumlichen Gliederung des fertiggestellten Bauwerks. Zu ihrer Kennzeichnung sind somit Zonenobjekte erforderlich, deren Verwendung unabhängig von der hierarchischen, räumlichen Gliederungsstruktur des Bauwerksmodells möglich ist. Ein solches Objekt steht innerhalb der IFC 2x3 nicht zur Verfügung. In IFC 2x4 ist jedoch der neue Objekttyp `IfcSpatialZone` vorhanden, der diese Anforderungen erfüllt. Der Wert dessen Attributs *PredefinedType* wird zu .USERDEFINED. gesetzt, da (bisher) kein vordefinierter Typ für den Kontext der Terminplanung existiert. Die Definition der Geometrie einer `IfcSpatialZone` erfolgt mittels eines `IfcProductDefinitionShape`. Somit bestehen die gleichen Möglichkeiten wie für ein Bauteilobjekt. Es können mehrere Representationsarten verwendet werden, beispielsweise für 2D ein `IfcGeometricCurveSet` und für 3D ein `IfcSweptAreaSolid`. Für die Repräsentation im 2D können sogar direkt die `IfcPolyline`-Objekte begrenzender Achsen referenziert werden (siehe Abbildung C.6 und C.7 im Anhang). Somit können Zonen direkt durch Achsen begrenzt werden. Wurde in einer IFC-Datei keine 3D-Repräsentation hinterlegt, so kann die Höhe des dem Achsraster zugewiesenen Geschosses als Extrusionslänge verwendet werden.

Aufgrund des Fehlens eines geeigneten Objektes zur Abbildung von für die Einteilung des Bauwerks in Bauabschnitt verwendeten Zonen ist ein Datenaustausch rein auf Basis von IFC 2x3 nicht ausreichend für die Umsetzung der entwickelten Konzepte. Andererseits befindet sich IFC 2x4 noch im beta-Stadium und wird daher derzeit noch von keiner kommerziellen Software unterstützt. Aufgrund von Änderungen in wesentlichen Aspekten wie z.B. der Definition von Materialien besteht jedoch keine Abwärtskompatibilität. Daher wurde im Rahmen dieser Arbeit anstatt einer Nutzung von IFC 2x4 das EXPRESS-Schema von IFC 2x3, unter Verzicht auf eine Vererbung, um die beiden Objekte `IfcSpatialElement` und `IfcSpatialZoneTypeEnum` ergänzt. Auf diese Weise konnten die im Rahmen der modellbasierten Terminplanung gestellten Anforderungen an den IFC-basierten Datenaustausch von Zonen und Achsen erfüllt werden und gleichzeitig die Daten kommerzieller CAD-Anwendungen verarbeitet werden.

4.9.5 Terminplan

Eine ausführliche Beschreibung, wie Terminplandaten in IFC abgebildet werden, findet man in [Seren u. Karstila 2001] und [Froese u. a. 1999]. Es existieren Objekte zur Abbildung eines Terminplans (`IfcWorkPlan, IfcWorkSchedule`), einzelner Vorgänge und Meilensteine (`IfcTask`), deren hierarchischen Aufbaus (`IfcRelNests`) sowie Objekte für die Abbildung verschiedener Typen von Vorgänger- und Nachfolgerbeziehungen (`IfcRelSequence`). Eine Zuordnung von Objekten des Bauwerksmodells zu einem Terminplanvorgang gemäß Relation B_2 aus Abbildung 4.11 auf Seite 62 erfolgt mittels eines Relationsobjektes des Typs `IfcRelAssignsToProcess`. Der gleiche Relationstyp kann verwendet werden, um Terminplanvorgängen verschiedene Arten von Ressourcen (Subtypen von `IfcConstructionResource`) zuzuordnen. Innerhalb einer IFC Datei können zudem mehrere unterschiedliche Terminpläne z.B. für Soll und Ist existieren,

deren Typ durch das Attribut *WorkControlType* bzw. *UserDefinedControlType* festgelegt wird.
Als Defizit wurde identifiziert, dass in IFC Vorgänge innerhalb einer Hierarchieebene keine Reihenfolge besitzen. D.h., es ist nicht sichergestellt, dass solche Vorgänge nach einem Datenaustausch in einer Terminplanungssoftware wieder in der gleichen Reihenfolge dargestellt werden. Um dieses Problem zu beheben, wurde jedem `IfcTask` über ein `IfcPropertySet` eine die Position im Terminplan angebende Nummer[41] zugewiesen (Abbildung C.8 im Anhang).
Ein Austausch von Terminplandaten kann zudem nur fehlerfrei erfolgen, wenn in der sendenden und empfangenen Anwendung die gleichen Kalendereinstellungen, d.h. Arbeitstage pro Woche, Arbeitstunden am Tag, Feiertage im Jahr usw., verwendet werden. Ansonsten erfolgt eine unterschiedliche Interpretation der Dauer von Vorgängen und damit eine terminliche Verschiebung der gesamten Prozesskette. In größeren Projekten ist es zudem durchaus üblich, innerhalb eines Terminplans für Vorgänge bzw. Ressourcen verschiedener Projektbeteiligter unterschiedliche Kalendereinstellungen zu verwenden. Somit sind mitunter zahlreiche Einstellungen abzugleichen, woraus der Bedarf zum Austausch auch dieser Daten resultiert. Die IFC bieten bisher jedoch keine spezifischen Objekte für die Abbildung von Kalendereinstellungen. Im Rahmen dieser Arbeit wurde daher auf eine weitere Betrachtung dieser Daten verzichtet und eine projektweit einheitliche Einstellung in den verwendeten Terminplanungsprogrammen vorausgesetzt.
Das Fehlen von IFC-Objekten zur Abbildung von Kalenderdaten wurde zudem im Rahmen des Reviews von IFC2x4 beta2 in die *Issue Resolution Database (IRD)* eingetragen. Eine Lösung wird derzeit unter dem Eintrag IFR-445 diskutiert. Im Zuge dessen wurde das Objekt `IfcWorkCalendar` neu eingeführt.

4.9.6 4D-Simulation

Zwar können Bauteilobjekte, wie bereits beschrieben, über Relationsobjekte des Typs `IfcRelAssignsToProcess` mit Vorgängen im Terminplan verknüpft werden, für die Abbildung von Visualisierungsparametern existieren jedoch in IFC bisher keine spezifischen Objekte. Daher wurde hierfür ein Satz frei definierter Eigenschaften (`IfcPropertySet`) gemäß Abbildung C.9 im Anhang verwendet. Dieser enthält die Farbe und Transparenz, mit der verknüpfte Bauteile aktiver Terminplanvorgänge dargestellt werden, sowie den die Darstellung beeinflussenden Prozesstyp. Dabei werden für die Farbe die Komponenten eines RGB-Farbwertes einzeln abgebildet und mit Werten zwischen 0 und 255 belegt. Es wurde keine Möglichkeit gefunden, ein `IfcColourRgb` innerhalb eines `IfcPropertySet`s zu verwenden. Die Transparenz wurde als Prozentzahl zwischen 0 und 100 abgebildet. Für den Prozesstyp wurde eine Auswahl von sechs Typen vordefiniert, die gemäß Abbildung C.10 im Anhang zu interpretieren sind. Weitere Prozesstypen können bei Bedarf hinzugefügt werden.
Das so definierte `IfcPropertySet` erhält für das Attribut *Name* den Prefix `4DVisualParameter;` und wird gemäß Relation B_3 aus Abbildung 4.11 auf Seite 62 über ein `IfcRelDefinesByProperties` einem oder mehreren Terminplanvorgängen zugeordnet. Die bevorzugte Variante der Repräsentation von 4D-Verknüpfungen und ihren Visuali-

[41] Hierfür wurde wie üblich der Projektstrukturplan-Code (PSP-Code) verwendet.

sierungsparametern gemäß Abbildung 4.12 auf Seite 63 konnte in IFC nicht umgesetzt werden, da Relationsobjekten (Subtypen von `IfcRelationship`) keine zusätzlichen Eigenschaften zugeordnet werden können.

4.9.7 Weitere Verknüpfungen

Im Rahmen einer modellbasierten Arbeitsweise werden vom Terminplaner neben den Verknüpfungen für eine 4D-Simulation auch noch die Verknüpfungen der zwei Relationen B_1 und B_5 gemäß Abbildung 4.10 auf Seite 61 bzw. Abbildung 4.13 auf Seite 64 erstellt:

- B_1: Verknüpfung zwischen Terminplanvorgang und Kostenwerten,
- B_5: Verknüpfung zwischen Terminplanvorgang und Mengenwerten.

In der heute üblichen Arbeitsweise werden diese weder erstellt noch ausgetauscht. Insbesondere für die Nachvollziehbarkeit, aber auch im Zusammenhang mit der Reaktion auf Änderungen in zugrunde liegenden Ausgangsdaten ist die Speicherung bzw. Weitergabe dieser Beziehungen jedoch sinnvoll.

Terminplan mit Kosten: Gemäß der rechtseindeutigen Relation B_1 sind Beziehungen zwischen einem Kostenwert und ein oder mehreren Terminplanvorgängen abzubilden. D.h., es ist ein `IfcCostValue` mit ein oder mehreren `IfcTask` zu verknüpfen. Dieses kann durch ein Relationsobjekt vom Typ `IfcRelAssociatesAppliedValue` direkt realisiert werden (siehe Abbildung C.11 im Anhang). Da ein `IfcCostValue` jedoch kein inverses Attribut aufweist, ist es in diesem Fall nicht ohne Weiteres möglich, auf die zugehörige Position (`IfcCostItem`) und deren Beschreibung innerhalb der Aufstellung des Kostenbudgets zuzugreifen.
Um dieses Problem zu umgehen, können alternativ ein oder mehrere `IfcRelAssignsToProcess`-Objekte verwendet werden, die jeweils ein `IfcCostItem` mit einer `IfcTask` verknüpfen (siehe Abbildung C.12 im Anhang). Aufgrund der eineindeutigen Zuordnung von `IfcCostValue`-Objekten und `IfcCostItem`-Objekten innerhalb der Aufstellung des Kostenbudgets können die gewünschten Verknüpfungen gemäß B_1 auf diese Weise indirekt hergestellt werden.

Terminplan mit Mengen: Gemäß Relation B_5 sind ein Terminplanvorgang mit einem oder mehreren Mengenwerten bzw. ein Mengenwert mit einem oder mehreren Terminplanvorgängen zu verknüpfen. Wie bereits erläutert, werden sowohl Basis- als auch LV-Mengenwerte durch Subtypen von `IfcElementQuantity` abgebildet. In IFC existiert jedoch kein Relationsobjekt, das es erlaubt, einen Terminplanvorgang (`IfcTask`) direkt mit Objekten diesen Typs zu verknüpfen.
Statt dessen kann ein `IfcRelAssignsToProcess`-Objekt verwendet werden, um eine Beziehung zwischen einem Terminplanvorgang und ein oder mehreren Einzelbeiträgen zu einer LV-Positionen (`IfcCostItem`) herzustellen. Da letztere eineindeutig mit Mengenwerten Verknüpft sind, wird so wiederum indirekt die gewünschte Zuordnung erreicht. Durch die Verwendung von mehreren Relationsobjekten kann ein Mengenwert mit mehreren Terminplanvorgängen verknüpft werden (siehe Abbildung C.13 im Anhang). Im Attribut *QuantityInProcess* der Relationsobjekte kann zusätzlich der Summenwert von einem Terminplanvorgang zugeordneten Mengenwerten hinterlegt werden.

Die Verknüpfung von Terminplanvorgängen und Basismengen kann analog erfolgen. Ein `IfcRelAssignsToProcess` verknüpft dabei eine `IfcTask` mit ein oder mehreren `IfcBuildingElement`-Objekten (siehe Abbildung C.14 im Anhang). Da aber mehrere Basismengen mit einem `IfcBuildingElement` verknüpft sein können, ergibt sich keine eindeutige Zuordnung zwischen Mengenwert und Terminplanvorgang. Um zumindest den Summenwert der tatsächlich referenzierten Basismengen zu hinterlegen, wird das Attribut *QuantityInProcess* des `IfcRelAssignsToProcess`-Objektes genutzt.
Relationsobjekte vom Typ `IfcRelAssignsToProcess`, die Terminplanvorgänge mit Bauteilobjekten verknüpfen, werden zudem auch für die Abbildung der Beziehungen im Rahmen einer 4D-Simulation verwendet. Werden für die Berechnung der Vorgangsdauer eines Terminplanvorgangs die Basismengen der selben Bauteilobjekte verwendet wie für dessen Visualisierung innerhalb einer 4D-Simulation, so wird nur ein `IfcRelAssignsToProcess` Objekt benötigt. Werden für die Visualisierung jedoch andere Bauteilobjekte verwendet als für die Berechnung der Vorgangsdauer (z.B. werden für die Visualisierung von Malerarbeiten in der Regel Raumobjekte verwendet wohingegen für die Berechnung der Vorgangsdauer die Seitenflächen der Wände maßgebend sind), so werden zwei Relationsobjekte vom Typ `IfcRelAssignsToProcess` verwendet. Dasjenige Relationsobjekt, das die Verknüpfung zu den Bauteilobjekten mit den relevanten Mengenwerten herstellt, erhält zur Unterscheidung den Wert `QuantityAssignmentOnly` für das Attribut *Description*.

4.9.8 Verknüpfungsregeln

Wie in [Weise u. a. 2009] näher erläutert, beschränkt sich der Anwendungsbereich der IFC derzeit auf die Beschreibung eines einzigen Modellzustands. Die Abbildung einer Datenhistorie sowie verwendeter Designregeln liegt nicht im Betrachtungsbereich der IFC. Sollen Regeln für die Verknüpfung zwischen Terminplanvorgängen und Modellinformationen, die auf Basis der in Kapitel 4.6 entwickelten Verknüpfungssprache erstellt wurden, ausgetauscht werden, so kann dieses jedoch in einem referenzierten, externen Dokument gemäß Abbildung C.15 im Anhang erfolgen. Dabei werden sämtliche erstellten Regeln zeilenweise in einer Textdatei aufgelistet. Über ihre Zeilennummer können sie aus IFC über ein `IfcDocumentReference`-Objekt referenziert werden. Ein `IfcRelAssociatesDocument` dient der Zuordnung eines Terminplanvorgangs, einer Regel im Dokument und den als Ergebnis der Regel referenzierten Bauteilobjekten im Modell. Werden in einer Regel geometrische Anfragen verwendet, so können auf gleiche Art zusätzlich die beteiligten Zonen oder Achsen referenziert werden. Unter Verwendung von IFC 2x4 können zusätzlich Bauteilobjekte, die als innerhalb einer Zone liegend erkannt wurden, der Zone über ein Relationsobjekt vom Typ `IfcRelReferencedInSpatialStructure` zugeordnet werden, um diese Information Softwareanwendungen zur Verfügung zu stellen, die keine derartige räumliche Auswertungsfunktionalität besitzen.

5 Umsetzung

Als Basis für die Erprobung der entwickelten Konzepte wurde ein verfügbares 4D-Simulationspaket benötigt, das einerseits einen vollständigen IFC Im- und Export als auch eine interaktive, bidirektionale Schnittstelle zu einer üblichen kommerziellen Terminplanungssoftware aufweist und andererseits eine leistungsfähige Programmierschnittstelle mit vollem Datenzugriff zum Ergänzen weiterer Funktionalitäten zur Verfügung stellt. Keines der im Rahmen der Marktanalyse betrachteten Programmpakete konnte diese Anforderungen unmittelbar erfüllen. Eine IFC-Unterstützung existiert bei gängigen kommerziellen Terminplanungsprogrammen aufgrund des branchenübergreifenden Zielmarkts nicht. Doch auch bei 4D-Softwarepaketen fehlt stets eine IFC-Unterstützung, die alle im Rahmen der modellbasierten Terminplanung relevanten Daten umfasst. In der Regel ist maximal ein (selektiver) Import von Geometrieinformationen verfügbar. Der Import von Terminplan-, Mengen- und Kostendaten sowie ein Export von IFC-Daten wird in der Regel nicht geboten. Es wurde daher ein entsprechendes Softwarepaket neu erstellt und um einen Interpreter für die spezifizierte Verknüpfungssprache sowie eine Anbindung an einen versionierten Workspace ergänzt. Die im Rahmen dieser Arbeit wesentlichen Komponenten der entwickelten Software sind in Abbildung 5.1 dargestellt.

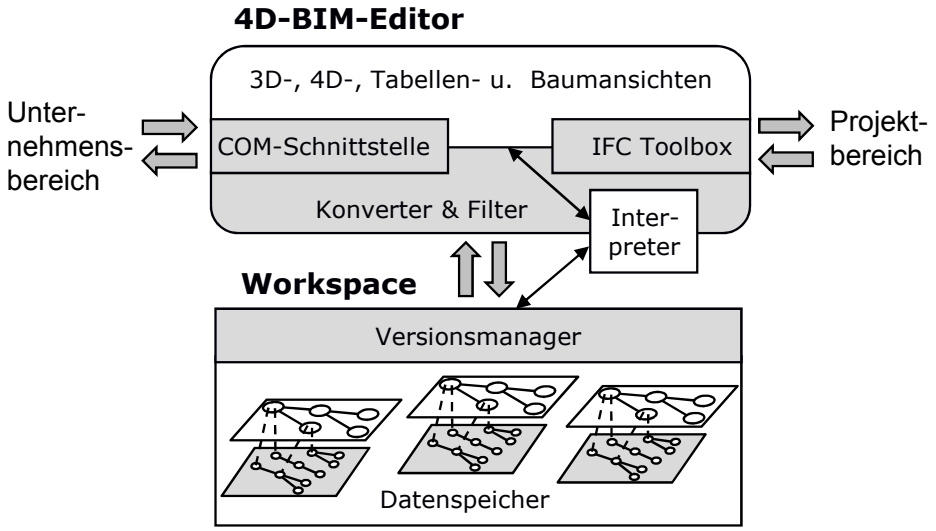


Abbildung 5.1: Komponenten der Umsetzung

Die Umsetzung erfolgte in Java. Wesentlicher Grund dafür ist, dass diese Programmiersprache in der Forschung und Lehre der Bauinformatik derzeit weit verbreitet ist und die Ergebnisse dieser Arbeit hierfür als Open Source zur Verfügung gestellt werden sollten[42].

[42] Die Software samt Dokumentation wird unter www.openifctools.de bereitgestellt.

5.1 4D-BIM-Editor

Der 4D-BIM-Editor stellt die Brücke zwischen dem unternehmensbezogenen und dem projektbezogenen Bereich dar und bietet dem Teminplaner eine graphische Benutzerschnittstelle für die Verwaltung und regelbasierte Verknüpfung der in Kapitel 4.5 betrachteten Modell- und Terminplandaten. Für die Anbindung einer gängigen Terminplanungssoftware wurde beispielhaft MS Project gewählt. Die einzelnen Komponenten werden nachfolgend vorgestellt.

5.1.1 IFC Toolbox

Aufgrund des Mangels frei verfügbarer und leicht anwendbarer Werkzeuge für das Lesen, Schreiben und Verwalten von Informationen aus IFC-Dateien wurde eine IFC-Toolbox entwickelt, die einen objektorientierten Zugriff auf die in einer Datei enthaltenen IFC-Objekte und deren Eigenschaften ermöglicht. Eine Schnittstelle gemäß des *Standard Data Access Interface (SDAI)* nach ISO 10303-22 und -27 wurde aufgrund des dort erfolgenden indirekten, funktionalen Zugriffs nicht umgesetzt[43].
Die IFC Toolbox besteht aus drei wesentlichen Komponenten:

- Ein Satz von Java-Klassen, die die in IFC definierten Objekte und Datentypen abbilden und Funktionalitäten zum Setzen und Lesen ihrer Objekteigenschaften bieten.
- Ein Parser für IFC-Dateien, die gemäß ISO 10303-21 kodiert sind (STEP physical files).
- Ein Modell zur Verwaltung von IFC Daten.

EXPRESS nach Java Compiler: Mit Hilfe des Parser Generators JavaCC [Copeland 2007] und der Grammatik der EXPRESS-Sprache[44] wurde ein limitierter EXPRESS nach Java Compiler erstellt[45]. Die wesentliche Limitation liegt darin, dass im EXPRESS-Schema enthaltene Regeln und Funktionen vernachlässigt werden. Mit dem erstellten Compiler wurde anschließend einmalig das EXPRESS-Schema der IFC-Version 2x3-TC1[46] in Java-Klassen übersetzt. Die gewählte Umsetzung in Java kann der Dokumentation der IFC Toolbox[42] entnommen werden. Für jedes Objektattribut werden *set*- und *get*-Methoden bereitgestellt.
Alle generierten Java-Klassen implementieren zusätzlich das *IfcRootInterface*. Dieses Interface ist nicht Teil der IFC-Specifikation, sondern wurde ergänzt, um einen übergeordneten Datentyp für alle mit IFC in Zusammenhang stehende Objekte zu erhalten. Es enthält die Methode `String getStepLine()`, welche die Textrepräsentation des

[43] Eine Lösung hierfür sind das Open Source Paket JSDAI (www.jsdai.net) und die IFC Engine DLL (www.ifcbrwoser.com).
[44] Grammatik der EXPRESS-Sprache aus ISO 10303-11:2004 Anhang A.
[45] Detaillierte Informationen über das Parsen der EXPRESS-Sprache kann der Dokumentation des *Open Source EXPRESS Parsers* von S. Lardet unter http://osexpress.sourceforge.net entnommen werden.
[46] www.iai-tech.org/products/ifc_specification/ifc-releases/ifc2x3-tc1-release/

entsprechenden Objektes für das Schreiben in eine IFC-Datei zurückliefert. Abgeleitet von `IfcRootInterface` wurden die Interfaces `IfcType` und `IfcClass` für die Unterscheidung zwischen Datentypen und Objekten der IFC ergänzt.

STEP-Datei-Parser: Beim Lesen einer IFC-Datei müssen die enthaltenen Informationen geparst und die vorgefundenen Objekte instanziiert werden. Hierzu wurde auf Basis der Syntaxbeschreibung[47] von STEP-Dateien mit JavaCC ein Parser entwickelt. Da nicht sichergestellt ist, dass Objekte in einer STEP-Datei definiert werden, bevor sie von anderen Objekten referenziert werden, müssen die Objekte beim Parsen zunächst mit einem Leerkonstruktor erzeugt und die Werte der Attribute zwischengespeichert werden. Nachdem die gesamte Datei geparst wurde, können dann die Attributwerte und Referenzen gesetzt werden.
Die wesentlichen Methoden des STEP-Datei-Parsers sind `void readStepFile(File)` zum Einlesen der übergebenen Datei und `Collection<IfcClass> getIfcObjects()` zum Anfordern der instanzierten IFC-Objekte.

Verwaltung von IFC Daten: Die Klasse `IfcModel` bildet die Schnittstelle für die Nutzung der IFC Toolbox. Von hier aus werden der STEP-Datei-Parser gestartet sowie die zurückerhaltenen IFC-Objekte verwaltet. Es werden Methoden zum Lesen und Schreiben von IFC-Dateien, zum Zugriff auf die enthaltenen Objekte sowie zum Hinzufügen und Entfernen von IFC-Objekten bereitgestellt. Die wichtigsten Methoden sind:

```
void readStepFile(File file)
void writeStepFile(File file)

Collection<IfcClass> getAllObjects()
Collection<?> getCollection(String type)
boolean containsIfcObject(String globalUniqueId)
IfcClass getIfcObjectByID(String globalUniqueId)
HashMap<String, IfcRootInterface>
         getIfcObjects(Collection<String> globalUniqueIds)

void addIfcObject(IfcClass ifcObject)
void addIfcObjects(Collection<IfcClass> ifcObjects}
void removeIfcObject(IfcClass ifcObject)
void removeIfcObjects(Collection<IfcClass> ifcObjects)
void removeIfcObjectById(String  globalUniqueId)

void resolveInverseAttributes()
String getNewGloalUniqueId()
```

Die Methode `getCollection` dient zum Abfragen aller Objekte eines bestimmten Typs. Ihr wird als Parameter der Name einer Klasse übergeben. Zurückgegeben werden alle Objekte, die in der Vererbungshierarchie Typ dieser Klasse sind. Mit dieser Methode können z.B. alle Objekte vom Typ `IfcBuildingElement` extrahiert werden. Hierzu gehören alle Türen, Fenster, Wände usw.

[47] Tabellen 1, 2 und 4 in ISO 10303-21

Objekte besitzen in IFC mitunter sog. inverse Attribute. Diese verweisen auf das Objekt referenzierende Relationsobjekt. Durch diese beidseitige Referenzierung wird das Navigieren durch IFC-Daten wesentlich vereinfacht. Jedoch werden inverse Attribute bei einem dateibasierten Datenaustausch nicht berücksichtigt. Daher müssen sie bei Bedarf durch das Analysieren von Relationsobjekten restauriert werden. Hierzu dient die Methode `resolveInverseAttributes`.

5.1.2 COM-Schnittstelle

Zur Realisierung einer interaktiven, bidirektionalen Schnittstelle zwischen dem 4D-BIM-Editor und der beispielhaft gewählten Terminplanungssoftware MS Project wurde die COM-Technologie (Component Object Model) von Microsoft genutzt. Diese ermöglicht das externe Starten bzw. Aktivieren von MS Project sowie einen lesenden bzw. schreibenden Zugriff auf dessen proprietäres Datenmodell. Für die Nutzung der COM-Technologie aus Java wurde mit dem Quellcodegenerator des Open Source Projekts Jawin[48] aus der Typenbibliothek von MS Project ein Satz von Java-Klassen generiert, über die die COM-Schnittstelle angesprochen werden kann. Zusätzlich ist dafür die Bibliothek jawin.dll in das entwickelte Java Projekt einzubinden. Diese enthält den für die Kommunikation erforderlichen nativen Code auf Basis von Java JNI. Die Methoden für die Kommunikation mit MS Project wurden in der Klasse `MSProjClient` gekapselt. Sie stellt folgende Methoden zur Verfügung:

```
void openCOMConnection()
void closeConnection()
boolean isConnected()

void setMSProjectVisible(boolean status)
void quitMSProject()

ArrayList<ScheduleTask> importSchedule()
void exportSchedule(TreeSet<ScheduleTask> schedule)
```

Der Abgleich der Terminplandaten in MS Project mit denen im 4D-BIM-Editor erfolgt mittels der beiden Methoden `importSchedule` und `exportSchedule`. Er ist auf die in Tabelle 5.1 angegebenen Informationen begrenzt. Andere Datenfelder in MS Project werden weder ausgelesen noch überschrieben. Insbesondere werden keine Ressourcendaten übertragen. Der Datenabgleich erfolgt per Knopfdruck innerhalb des 4D-BIM-Editors und bezieht sich auf den aktuell in MS Project geöffneten Terminplan. Ist MS Project nicht geöffnet oder kein Terminplan geladen, so wird die Anwendung gestartet und ein neuer Terminplan angelegt.
Grundlage für die Synchronisation von Daten ist eine eindeutige ID je Terminplanvorgang. Hierbei ergibt sich das Problem, dass auf die in MS Project verwendete eindeutige ID oder GUID[49] nur lesend zugegriffen werden kann. D.h., wird über die COM-

[48] http://jawinproject.sourceforge.net
[49] Eine GUID existiert erst in neueren MS Project Versionen. Zuvor wurde eine Ganzzahl als eindeutige ID verwendet.

Datenfeld in MS Project[50]	Beschreibung
UniqueID bzw. GUID	Eindeutige Nr. des Vorgangs (nur Lesezugriff)
OutlineNumber	Gliederungsnummer (PSP-Code) zur Beschreibung der Position des Vorgangs innerhalb der Terminplanhierarchie
Name	Name bzw. Beschreibung des Vorgangs
Start	Anfangstermin des Vorgangs
Finish	Endtermin des Vorgangs
Duration	Dauer des Vorgangs
Predecessors	Eine durch Semikolon getrennte Liste von Vorgängerbeziehungen
Successors	Eine durch Semikolon getrennte Liste von Nachfolgerbeziehungen
Number1..5	Mengenwerte (bis zu fünf verschiedene)
Number11	Geplante Kosten eines Gewerks in EUR
Text1..5	Beschreibung der Mengenwerte
Text6..10	Einheiten der Mengenwerte
Text16..21	eindeutige ID des gewählten Datensatzes der Aufwandswertedatenbank
OutlineCode1..5	Mengenart (Auswahl aus LV, Basis, -)
OutlineCode6	Berechnungsmethode (Auswahl aus Methode A-D)
OutlineCode7	Bezeichnung des Gewerks (Auswahl aus vordefinierter Liste)

Tabelle 5.1: Beim Abgleich mit MS Project berücksichtigte Datenfelder eines Vorgangs

Schnittstelle ein neuer Terminplanvorgang angelegt, so erhält dieser automatisch eine eindeutige ID, die weder vorgegeben noch im Nachhinein angepasst werden kann. Dieses stellt immer dann ein Problem dar, wenn über den 4D-BIM-Editor aus IFC ein Terminplan importiert wird. Um dieses Problem zu umgehen, muss in einem Datenfeld von MS Project mit Lese- und Schreibzugriff (z.B. *Text30*) eine im Projekt eindeutige ID verwaltet werden. Hierzu kann die in IFC für Terminplanvorgänge verwendete GUID genutzt werden. Wird andersherum beim Auslesen der Daten aus MS Project ein neuer Terminplanvorgang ohne eine solche ID angetroffen, so wird diese vom 4D-BIM-Editor neu erzeugt. Somit ist eine persistente Identifikation aller am Datenaustausch beteiligten Terminplanvorgänge sichergestellt. Beim Setzen der Vorgänger- und Nachfolgerbeziehungen sind die MS Project internen und die in IFC verwendeten Identifikatoren aufeinander abzubilden.
Ein weiteres Problem ergibt sich beim Synchronisieren des Anfangs- und Endtermins sowie der Vorgangsdauer. In MS Project, wie auch in anderen Terminplanungsprogrammen, entspricht die Vorgangsdauer in der Regel der *Arbeitszeit* zwischen Anfangs- und Endtermin. Die Definition der Arbeitszeit (Arbeitstage pro Woche, Arbeitsstunden pro

[50] Die Beschriftungen der Datenfelder (Spalten) können in MS Project für die Anzeige abweichend gewählt werden.

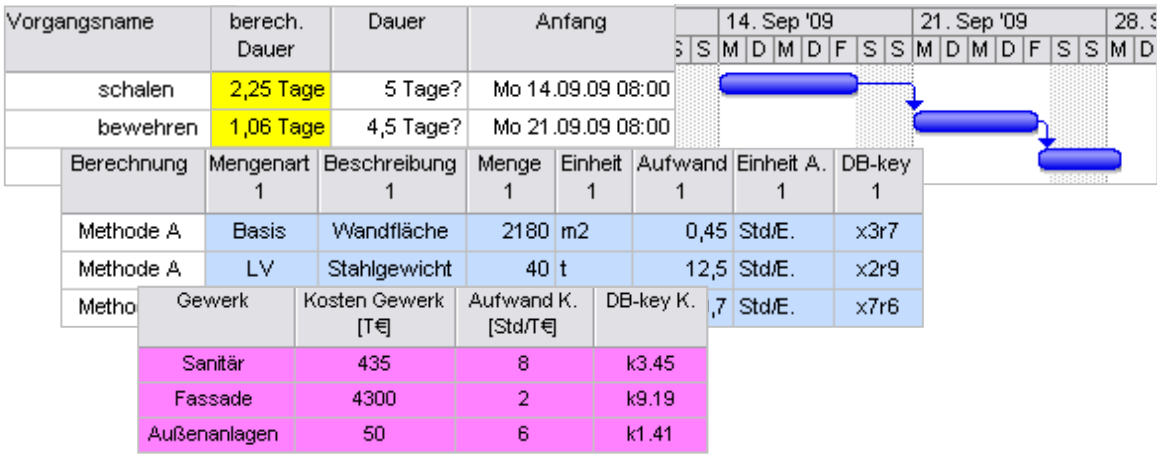


Abbildung 5.2: Benutzerdefinierte Felder in MS Project zur Berechnung der Dauer von Vorgängen auf Basis aus dem IFC-Modell extrahierter Daten

Tag, Feiertage usw.) erfolgt über vom Anwender konfigurierbare Kalender. Wie bereits in Kapitel 4.9.5 erwähnt, können innerhalb eines Terminplans mehrere verschiedene Kalender für Vorgänge und zugeordnete Ressourcen verwendet werden. Wird im Terminplanprogramm einer der drei Werte Start-, Endtermin, Vorgangsdauer verändert bzw. gesetzt, so wird einer der anderen beiden gemäß den Kalenderdaten automatisch angepasst. Somit ist ein Setzen aller drei Werte nur möglich, wenn sie mit dem verwendeten Kalender kompatibel sind. D.h., die Kalenderdaten müssen einmalig zwischen allen Projektbeteiligten abgestimmt und eingestellt oder bei jedem Datenaustausch mit übertragen werden. Im Rahmen dieser Arbeit wurde von Ersterem ausgegangen.
Die im 4D-BIM-Editor mit Hilfe der Verknüpfungssprache extrahierten Modelldaten (Basismengen, LV-Mengen, Kosten) sowie die dabei vom Nutzer eingegebenen Zusatzinformationen, wie die verwendete Berechnungsmethode (gemäß Kapitel 2.1.3) und die Art der genutzten Mengen (Basis- oder LV-Mengen), werden an MS Project übergeben. Zusätzlich werden den Mengenwerten zugeordnete Einheiten und Beschreibungen übermittelt. Bei Verwendung von Methode C werden die Bezeichnung des Gewerks und die zugehörigen Kosten übertragen (stets in EUR). Mit aus einer Datenbank ausgewählten Aufwandswerten wird aus diesen Modelldaten durch eine Formel innerhalb des Datenfelds *Dauer1* von MS Project automatisch ein Richtwert für die Vorgangsdauer berechnet. Die tatsächliche Vorgangsdauer kann vom Terminplaner hiervon abweichend festgelegt werden, um auf diese Weise Erfahrung und erweitertes Projektwissen einbringen zu können (siehe Abbildung 5.2).

5.1.3 Datenmodell, Konvertierung und Filter

Im BIM-Editor müssen Daten zwischen den Datenmodellen von MS Project, dem Workspace und der IFC ineinander konvertiert werden. Als Basis für diese Konvertierung wird ein generisches Datenmodell verwendet, das direkt der Modellbildung in Kapitel 4.4 entspricht. Daten eines Teilmodells werden hierbei in einem geschachtelten `HashMaps` verwaltet:

```
HashMap<String, HashMap<String, String» partialModel;
```

Jedes innere `HashMap` repräsentiert ein Objekt und enthält somit als *key* die Attributnamen und als *value* die zugehörigen Attributwerte. Das äußere `HashMap` repräsentiert das Teilmodell. Es enthält die Objekte als *value* und verwendet deren IDs als *key*. Alle Attributwerte werden in Textform gespeichert. Das generische Modell kann vom Workspace direkt verarbeitet werden und ist über Konverter an die COM- und IFC-Schnittstelle angebunden. Bei der Konvertierung von IFC in das generische Modell findet zusätzlich eine Filterung statt. D.h., es werden aus dem IFC-Modell nur diejenigen Daten importiert, die für die modellbasierte Terminplanung von Interesse sind und für die eine Versionierung erfolgen soll. Im Einzelnen handelt es sich hierbei um die in Tabelle 4.1 auf Seite 65 aufgeführten Informationen. Diese selektive Verarbeitung bzw. Versionierung der importierten Daten reduziert die Informationsmenge und dient damit sowohl der Übersichtlichkeit als auch der Performance der eingesetzten Softwarekomponenten. Insbesondere das Level der Versionierung ist anforderungsgerecht zu wählen. So ist z.B. die Versionierung jeder einzelnen Koordinate der Geometriedaten für die Anwendung in der Terminplanung nicht erforderlich und aufgrund der damit einhergehenden Informationsflut auch nicht gewünscht. Daher werden anstatt der vollständigen Geometriedaten der Bauteilobjekte vereinfachend lediglich deren Bounding-Boxen in das geometrische Datenmodell überführt. Dieses ist in Verbindung mit Mengeninformationen für das Erkennen von Planungsänderungen in der Regel ausreichend. Um jedoch den Zugriff auf die vollständigen Geometriedaten und die ursprüngliche Struktur der Daten jederzeit zu ermöglichen, wird für jede importierte Modellversion zusätzlich der Pfad zur Quelldatei hinterlegt.
Das generische Modell ist Grundlage für die weitere Bearbeitung im Rahmen der Terminplanung. Dabei wird es mit zusätzlichen Informationen angereichert. Bei einer anschließenden Konvertierung der Terminplan- und 4D-Simulationsdaten zurück nach IFC zwecks Weitergabe an andere Planungspartner (IFC-Export) erfolgt ebenfalls eine Filterung. Diese dient dazu, private von öffentlichen Daten zu trennen. Beispielsweise werden die zur Berechnung der Dauer von Vorgängen verwendeten Aufwandswerte nicht an externe Planungspartner weitergegeben. Die Bauteildaten werden beim Export aus der IFC-Datei des Imports übernommen, da hieran während der Terminplanung keine Änderungen vorgenommen werden. (Eine ggf. vorzunehmende Teilung von Geometrieobjekten erzeugt lediglich zusätzliche Objekte.)

5.1.4 Benutzeroberfläche

Der BIM-Editor stellt verschiedene Ansichten bzw. Eingabe- und Steuerelemente bereit (siehe Abbildung 5.3). Als Basiselemente für die Benutzeroberfläche wurden folgende Softwarebibliotheken verwendet (alle als open source verfügbar): VLDocking Frameworks[51], SwingX[52] und L2FProd[53]. Darauf aufbauend werden vom 4D-BIM-Editor folgende Fenster bereitgestellt:

- ein 3D/4D-Fenster mit verschiedenen Navigations- und Selektionsfunktionen, vordefinierten Standardansichten, Schnittebenen, einer Funktionalität zum interak-

[51] www.vlsolutions.com
[52] http://swinglabs.org
[53] www.l2fprod.com

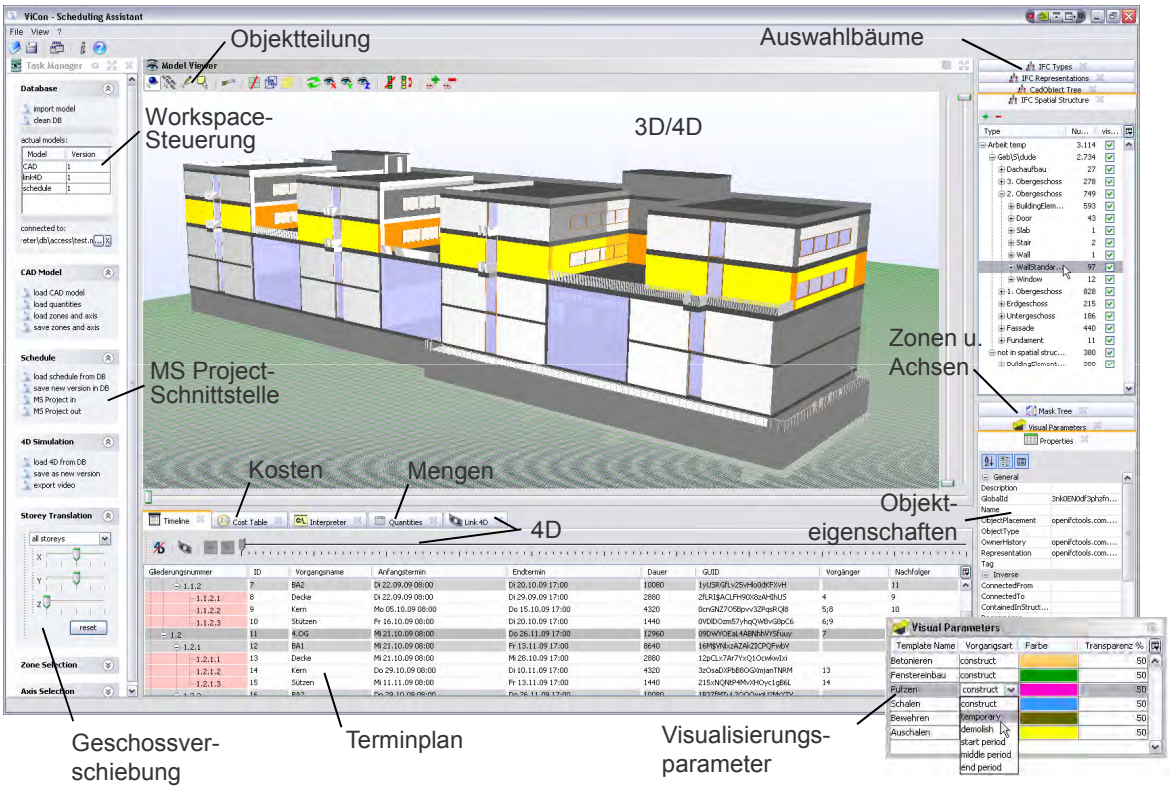


Abbildung 5.3: Oberfläche des entwickelten 4D-BIM-Editors

tiven Eingeben von Zonen und Achsen sowie zum Verschieben der Geschosse entsprechend Abbildung 2.5 auf Seite 26,

- mehrere Baumansichten (nach Gebäudestruktur, Bauteilart, Geometrierepräsentationsart) mit Funktionalitäten zum Selektieren bzw. Ein- und Ausschalten von Bauteilen,
- ein Fenster zur Anzeige von Eigenschaften eines selektierten Objektes,
- eine hierarchische Tabellenansicht des Terminplans mit einer Werkzeugleiste zum Erstellen und Steuern einer 4D-Simulation (kein Balkendiagramm des Terminplans),
- eine Tabellenansicht der existierenden Verknüpfungen zwischen Terminplan und Bauteilobjekten,
- ein Fenster zur Verwaltung von Templates für Visualisierungsparameter,
- ein Fenster für die Verwaltung von Zonen und Achsen,
- je eine Tabellenansicht für Mengen- und Kostenaufstellung,
- eine Steuerungsleiste mit integrierter Verwaltung des Workspaces sowie Steuerung der COM-Schnittstelle nach MS Project,
- eine Eingabemaske für Ausdrücke der Verknüpfungssprache.

Die 3D/4D-Visualisierung basiert auf Java3D[54]. Für die Interpretation von CSG-Geometrieinformationen der IFC wurden ein Boolean Modeller nach [Hubbard 1990] und ein zugehöriger Netzgenerator nach der Kettenmethode aus [Preparata u. Shamos 1985] implementiert. Informationen über die Interpretation von in IFC enthaltenen B-rep- bzw. 2D-Geometrieinformationen findet man in [Theiler 2008] [Theiler u. a. 2009] und [Theiler 2009]. Das Konzept für die Implementierung der 4D-Visualisierung ist in [Koschorrek u. a. 2008] erläutert. Die Anwendung der 4D-Funktionalität an einem Beispielprojekt ist in [Ester u. a. 2008] beschrieben.

5.2 Workspace

Der Workspace dient der Versionierung von Planungsdaten, deren Änderungen während der modellbasierten Terminplanung automatisch erkannt und dokumentiert werden sollen. Er besteht aus zwei Schichten: einem Versionsmanager und einem Datenspeicher.

5.2.1 Versionsmanager

Hierbei handelt es sich um eine Softwarekomponente zum Verwalten und Versionieren mehrerer Teilmodelle. Sie übernimmt die folgenden Aufgaben:

- Anlegen und Ändern des Datenbankschemas.
- Abspeichern eines neuen Teilmodells. Dieses beinhaltet das automatische Erkennen von Änderungen gegenüber der vorhergehenden Modellversion und das entsprechende Ergänzen der Änderungshistorie.
- Berechnung und Ausgabe der Daten eines Teilmodells in einer bestimmten Modellversion.
- Ausgabe von Informationen über die Änderungshistorie (bisher limitiert, weiterreichende Auswertungen können auf einfache Weise ergänzt werden).

Die Implementierung des Versionsmanagers erfolgte auf Basis der JDBC-Schnittstelle (Java Database Access) in der Klasse `VersionManager`. Diese stellt folgende Methoden zur Verfügung:

```
VersionManager(String driver, Sting url,
               String user, String password)
void createConnection()
void closeConnection()
ArrayList<String> getRegisteredModels()
printModel(HashMap<String, HashMap<String, String>> model)
void cleanDB()

String setModelName()
String getModelName()
```

[54] https://java3d.dev.java.net/

```
void createTables()
void clearTables()
void deleteTables()

storeModel(int prev, String[] infos,
               HashMap<String, HashMap<String, String>> model)
HashMap<String, HashMap<String, String>> getModel(int version)
String[] getModelInfos(int modelVersionNr)
String getCurrentModelVersion()
HashMap<String, String> getModelMembers()
String getCurrentObjectVersion(String objectId)
void getMVhistory(int src, int target,
               HashMap<String, ArrayList<Integer>> history)
```

Nach Übergabe der Verbindungsdetails für die als Datenspeicher verwendete Datenbank an den Konstruktor des `VersionManager`-Objektes und dem anschließenden Verbindungsaufbau mit der Methode `createConnection` besteht Zugriff auf den Workspace. Es können dann bereits übergeordnete Funktionen wie das Abfragen der Namen aller im Workspace gespeicherten Teilmodelle (`getRegisteredModels`) oder das Löschen (`cleanDB`) aller im Workspace gespeicherten Daten und Tabellen ausgeführt werden. Mit der Methode `printModel` kann zudem eine formatierte Ausgabe eines als Parameter übergebenen Teilmodells erfolgen.
Die Ausführung aller weiteren Funktionen erfolgt im Kontext eines einzelnen Teilmodells. Der Kontext muss daher zunächst mit der Methode `setModelName` gesetzt werden und kann später mit `getModelName` abgefragt werden. Soll ein zusätzliches Teilmodell in den Workspace aufgenommen werden, so muss das Datenbankschema um die dafür erforderlichen Tabellen erweitert werden. Dieses erfolgt durch die Methode `createTables`. Die Methode `deleteTables` entfernt im Gegensatz dazu ein Teilmodell komplett aus dem Workspace, wohingegen die Methode `clearTables` lediglich den Inhalt des Teilmodells und seine Historie löscht.
Der Import einer neuen Modellversion eines Teilmodells in den Workspace erfolgt mit der Methode `storeModel`. Ihr werden als Parameter die Nummer der als Vorgänger zu betrachtenden Modellversion, Zusatzinformationen wie ein Kommentar zur Beschreibung (z.B. „Update aufgrund Umplanung im 2.OG“) und der Pfad zur entsprechenden IFC-Datei sowie die zu speichernden Daten des Teilmodells in der generischen Modellstruktur des 4D-BIM-Editors übergeben. Die Verwendung der generischen Datenstruktur erlaubt es, alle Teilmodelle auf gleiche Weise abzubilden und damit alle Funktionen des Versionmanagers einheitlich anzuwenden. Zunächst wird der Zustand des Teilmodells in der angegebenen Vorgängerversion berechnet und dieser dann Objekt für Objekt und Attribut für Attribut mit dem zu speichernden Modellzustand verglichen. Werden dabei Änderungen erkannt, so werden die Modell- und Objektversionsgraphen gemäß Kapitel 4.8 um einen weiteren Blattknoten und die zugehörige Kante samt Bewertung ergänzt. Wurden keine Unterschiede zwischen den beiden Modellzuständen gefunden, so wird in Abweichung zum beschriebenem Konzept das neue Teilmodell nicht gespeichert und eine entsprechende Meldung am Bildschirm ausgegeben.
Die Methode `getModel` berechnet den Zustand des Teilmodells in der angegebenen

Modellversion und gibt ihn in der generischen Datenstruktur zurück. Sie wird von der Methode `storeModel` auch intern zur Berechnung der Vorgängerversion genutzt.
Die Methode `getModelInfos` gibt die für eine Modellversion gespeicherten Zusatzinformationen zurück. Mit `getCurrentModelVersion` kann die aktuelle Versionsnummer des Teilmodells erfragt werden, in deren Kontext die Anfrage gestellt wird. Die Methode `getModelMembers` gibt die IDs aller enthaltenen Objekte und deren zugehörige Versionsnummern zurück. Mit `getCurrentObjectVersion` kann die aktuelle Versionsnummer eines einzelnen Objektes abgefragt werden. Die Methode `getMVhistory` gibt für jedes Objekt in der Ausgangsmodellversion den Pfad im Objektversionsgraphen zurück, der zu der in der Zielmodellversion enthaltenen Objektversion führt. Sie dient einerseits dazu, dem Anwender einen Überblick über die Menge der erfolgten Änderungen zu verschaffen, und wird andererseits intern in der Methode `getModel` verwendet.
Der Versionsmanager unterstützt nur einen Single-User-Zugriff. D.h., eine Zusammenarbeit (gleichzeitiger Zugriff) mehrerer Terminplaner auf einem Workspace wurde nicht betrachtet. Um eine Zusammenarbeit zu ermöglichen, können mehrere Workspaces verwendet werden und die Zusammenführung der parallel erfolgten Änderungen im Projektdatenserver erfolgen. Private Daten können innerhalb einer Firma über den internen Server für die Terminplandaten und Ressourcen ausgetauscht werden. Durch ein geeignetes Workflowmanagement kann eine parallele Bearbeitung der gleichen Daten jedoch weitestgehend ausgeschlossen werden.

5.2.2 Datenspeicher

Für die Speicherung der Daten wurde aufgrund der weiten Verbreitung und der Möglichkeit zur Verwendung eines einheitlichen Schemas für alle Modellobjekte eine relationale Datenbank verwendet. Durch die Verwendung von JDBC (datenbankunabhängig) können zudem viele verschiedene relationale Datenbanksysteme als Datenspeicher für den Workspace verwendet werden[55]. Aufgrund von herstellerspezifischen SQL-Dialekten ist die Implementierung jedoch zu einem gewissen Anteil datenbankspezifisch. In der Umsetzung wurden bisher Implementierungen für die beiden Datenbanksysteme Java DB und MS Access erstellt. In der Oberfläche des 4D-BIM-Ediors lässt sich zusätzlich der Pfad zur Datenbank/Datei wählen, die als Workspace genutzt werden soll. So können für verschiedene Projekte unterschiedliche Datenbanken verwendet und die Daten getrennt gehalten werden.
Das Datenbankschema des Datenspeichers für den Workspace besteht aus sieben Tabellen je Teilmodell sowie einer Tabelle namens LM (loaded models), in der die Namen aller im Workspace enthaltenen Teilmodelle und deren aktuelle Versionsnummern registriert werden (siehe Abbildung D.1 im Anhang[56]). Die sieben Tabellennamen eines Teilmodelles ergeben sich aus dem Namen des Teilmodells und einem Suffix gemäß Tabelle 5.2.
Wird ein neues Teilmodell in den Workspace aufgenommen, so werden die entsprechenden sieben Tabellen neu angelegt und eine neue Zeile mit dem Namen des Teilmodells

[55] bzgl. verfügbarer Treiber siehe http://devapp.sun.com/product/jdbc/drivers
[56] Auf die Darstellung der zur Sicherung der referentiellen Integrität verwendeten Primär- und Fremdschlüssel wurde verzichtet.

Suffix	Bedeutung	enthaltene Daten
_MV	Modellversionen	Metainformationen zu jeder Modellversion
_OP	Objektpool	Attributbelegungen der aktuellen Objektversionen
_OS	Objektstatus	Kennzeichnung, ob ein Objekt Teil der aktuellen Modellversion ist
_DMV	Delta-Modellversion	Kanten des Modellversionsgraphens
_DMO	Delta-Modelloperation	Kantenbewertungen des Modellversionsgraphens
_DOV	Delta-Objektversion	Kanten der Objektversionsgraphen
_DOO	Delta-Objektoperation	Kantenbewertungen der Objektversionsgraphen

Tabelle 5.2: Tabellen zur Speicherung eines Teilmodells und seiner Änderungshistorie

und der Versionsnummer eins in die Tabelle LM eingefügt. Existierte vorher kein Teilmodell im Workspace, so wird die Tabelle LM ebenfalls angelegt. Beim Entfernen eines Teilmodells aus dem Workspace werden die zugehörigen sieben Tabellen gelöscht sowie der Eintrag aus der LM-Tabelle entfernt. Mit dem letzten Teilmodell wird auch die LM-Tabelle entfernt. Werden lediglich die Daten eines Teilmodells gelöscht, so werden die sieben Tabellen geleert und die zugehörige Modellversionsnummer in der Tabelle LM auf null gesetzt.

Abbildung des Referenzdatenbestands

Die Tabelle Θ_OP beinhaltet für jedes Objekt des Teilmodells Θ die Attributnamen und -werte der zuletzt gespeicherten Objektversion. Pro Objekt O existieren n Datensätze, wobei n der Anzahl der Attribute von O entspricht. In der Tabelle Θ_OS wird für jedes dieser Objekte dessen Versionsnummer hinterlegt und in der Spalte Status zusätzlich markiert, ob dieses Objekt Teil der aktuellen Modellversion ist (wahr/falsch). Bei jedem Import einer neuen Modellversion werden die neu enthaltenen Objekte zur Tabelle Θ_OP hinzugefügt sowie die Attribute geänderter Objekte angepasst. Gelöschte Objekte verbleiben unverändert in der Datenbank bestehen. Die Tabelle Θ_OS wird entsprechend aktualisiert.

Abbildung von Metainformationen

Zum Speichern von Metainformationen wird die Tabelle Θ_MV verwendet. Sie enthält für jede in den Workspace importierte Modellversion einen Datensatz, der den vom Nutzer angegebenen Kommentar, den Pfad zur Ursprungsdatei sowie das Datum und die Zeitangabe des Imports enthält.

Abbildung des Modellversionsgraphens

Für die Speicherung des Modellversionsgraphens werden die Tabellen Θ_DMV und Θ_DMO verwendet. Erstere enthält die Kanten des Graphens, wohingegen letztere die Kantenbewertungen beinhaltet. Dabei können pro Kante mehrere Bewertungen existieren. Die beiden Tabellen sind über die DMV_ID miteinander verknüpft. Eine

Kantenbewertung besteht aus der ID des betroffenen Objektes sowie der gelöschten (del_OV) und neu hinzugefügten (gen_OV) Versionsnummer. Wurde ein Objekt neu erzeugt oder gelöscht, so wird in der Kantenbewertung der Wert für del_OV bzw. gen_OV auf null gesetzt.

Abbildung der Objektversionsgraphen

Die Kanten aller Objektversionsgraphen werden in der Tabelle Θ_DOO gespeichert. Zur Zuordnung zu einem Objektversionsgraphen wird mit jeder Kante zusätzlich die zugehörige Objekt-ID hinterlegt. Die Kantenbewertungen der Objektversionsgraphen werden analog zum Modellversionsgraphen in der Tabelle Θ_DOV abgespeichert. Die Beziehung der beiden Tabellen wird über das Datenfeld DOV_ID hergestellt.

Beim ersten Import eines Teilmodells existiert keine Vorgängerversion im Workspace. Es brauchen daher keine Kanten in den Modell- bzw. die Objektversionsgraphen eingefügt zu werden. D.h., die Tabellen Θ_DMV, Θ_DMO, Θ_DOV und Θ_DOO bleiben nach dem ersten Import leer.
Alternativ kann eine Vorgängerversion mit der Versionsnummer null angenommen werden, die keine Objekte enthält. In diesem Fall wird in die Tabelle Θ_DMV und für jedes Objekt in die Tabelle Θ_DOV je eine Kante zwischen der Version null und eins eingetragen. In die Tabellen Θ_DMO und Θ_DOO werden die zugehörigen Kantenbewertungen eingetragen, die das Erstellen jedes einzelnen Objektes bzw. Attributes beschreiben. Durch dieses Vorgehen wird der vollständige Zustand der ersten Modellversion dauerhaft in den Versionsgraphen abgelegt. Somit steht ein zusätzlicher Referenzdatenbestand zur Verfügung, wodurch der maximale Berechnungsweg zum Restaurieren einer Modellversion halbiert wird.

5.3 Sprachinterpreter

5.3.1 Funktionsweise und Benutzerschnittstelle

Die Benutzerschnittstelle zur Eingabe von Verknüpfungsregeln mittels der in Kapitel 4.6.2 spezifizierten Sprache ist in Abbildung 5.4 auf der nächsten Seite dargestellt. Sie ist in den 4D-BIM-Editor integriert und besteht aus einer Kommandozeile für die Texteingabe, einer Tabelle mit Sprachbausteinen als Vorlagen zur Verwendung per Drag&Drop sowie je einem Ausgabebereich für Compilerinformationen und die Ergebniselemente. Die Sprachvorlagen sind zur besseren Übersicht in vier Bereiche (Spalten) unterteilt:

- Basics: Umfasst Sprachkonstrukte zum Erstellen und Schachteln von Mengenausdrücken (*set_terms*) inklusive Filtern sowie die zusätzlichen Schlüsselwörter *allValues* und *allAttributes*.
- Functions: Beinhaltet die Funktionen zur Weiterverarbeitung der erhaltenen Ergebnismenge (*functions*) bzw. zum Erstellen von Teilmodellen (*setCreationFunctions*).

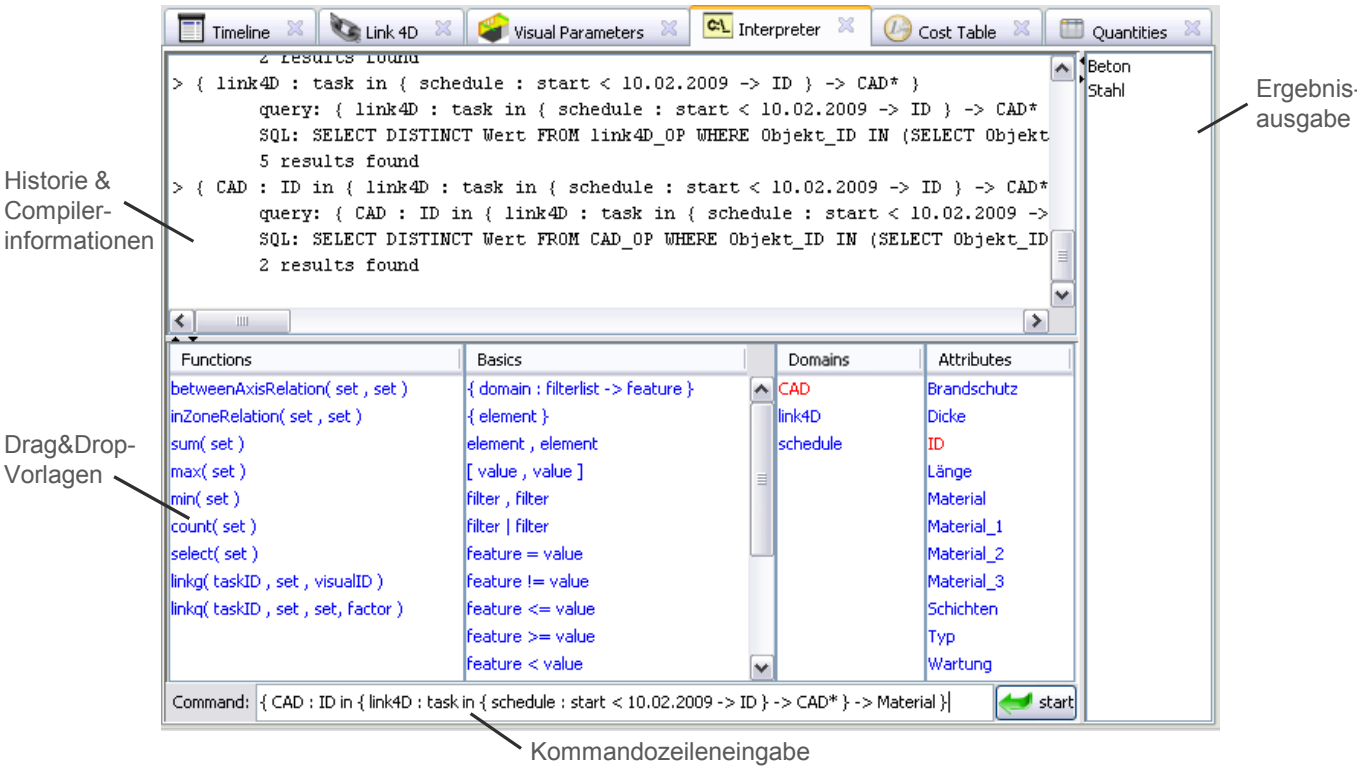


Abbildung 5.4: Userinterface des Sprachinterpreters

- Domains (live-Ansicht): Zeigt die im aktuell genutzten Workspace enthaltenen Teilmodelle (*identifiers*).
- Attributes (live-Ansicht): Zeigt alle im derzeit angewählten Teilmodell (rot markiert) vorhandenen Attributnamen (*features*).

Klickt der Anwender auf einen Attributnamen im Vorlagenbereich, so werden in der Ergebnisausgabe alle hierfür im Teilmodell existierenden Werte aufgelistet. Sie können ebenfalls per Drag&Drop in die Kommandozeile gezogen und als Vergleichswerte bei der Definition von Filtern verwendet werden. Bei der Nutzung von Drag&Drop erfolgt eine Wortersetzung, d.h., wird ein Sprachbaustein auf einem bereits innerhalb der Kommandozeile vorhandenen Wort fallen gelassen, so wird das Wort durch den Sprachbaustein ersetzt. Dieses erleichtert das Erstellen geschachtelter Mengenausdrücke. Zusätzlich erfolgt ein automatischer Leerzeichenausgleich. Innerhalb der Kommandozeile können zudem einzelne Worte (durch Doppelklick) bzw. Textbereiche (durch Markieren) ausgewählt und anschließend analog per Drag&Drop verwendet werden. Die Kommandozeile bietet zudem eine Historie der bisher eingegebenen Sprachausdrücke. Durch sie kann mit den Pfeiltasten der Tastatur navigiert werden.
Das Zusammenspiel von Benutzerobefläche, Makroprozessor und Sprachinterpreter ist in Abbildung 5.5 verdeutlicht. Zunächst wird vom Anwender ein Kommando oder Sprachausdruck eingegeben oder per Drag&Drop konstruiert. Sofort nach dem Starten der Verarbeitung wird die Eingabe dann in die Historie eingetragen und an den Makroprozessor weitergeleitet. Dieser untersucht die Eingabe auf Kommandos gemäß Tabelle 4.2 auf Seite 73 und führt diese aus. Im Fall eines erkannten Kommandos (wie z.B. der Definition eines Makros) erfolgt keine Weitergabe an den Sprachinterpreter. Handelt

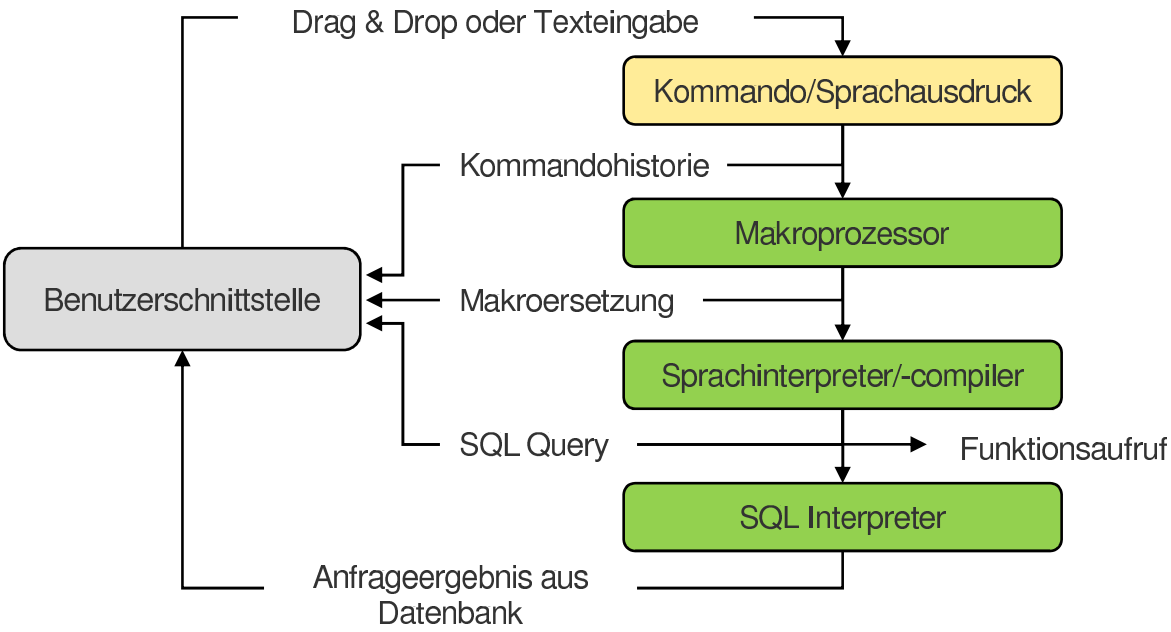


Abbildung 5.5: Prinzipielle Funktionsweise des Interpreters für die Verknüpfungssprache

es sich hingegen um einen Sprachausdruck, so erfolgt nach einer ggf. durchgeführten Makroersetzung die Weitergabe des expandierten Ausdrucks an den Sprachinterpreter bzw. -compiler, der das Ergebnis des Ausdrucks ermittelt. Dabei greift er sowohl auf den Workspace als auch auf Funktionalitäten des 4D-BIM-Editors zurück. Rein alphanumerische Anfragen werden direkt in SQL übersetzt. Für im Sprachausdruck enthaltene räumliche Anfragen (*setCreationFunctions*) erfolgt hingegen zunächst per Funktionsaufruf eine geometrische Auswertung im 4D-BIM-Editor. Die bei einer ggf. dabei durchgeführten Objektteilung neu entstandenen Objekte werden zum Teilmodell CAD hinzugefügt. Die IDs derjenigen CAD-Objekte, die die räumliche Anfrage erfüllen, werden in einer temporären Tabelle im Workspace hinterlegt. In die vom Sprachcompiler für alphanumerische Anfragen erzeugte SQL-Anweisung wird anschließend ein entsprechender Passus eingefügt, mit dem die Ergebnisinformationen der räumlichen Anfrage in der temporären Tabelle berücksichtigt werden. Nach vollständiger Übersetzung des Sprachausdrucks wird die SQL-Anfrage an die als Datenspeicher im Workspace verwendete Datenbank weitergeleitet und dort interpretiert. Somit wird eine räumliche Anfrage auf eine alphanumerische Datenbankanfrage mittels SQL zurückgeführt. Die Weiterverarbeitung der aus der SQL-Anfrage erhaltenen Elemente (mittels *functions*) erfolgt anschließend Java-seitig durch entsprechenden Programmcode.
Zur Information des Anwenders werden die Eingabedaten vor und nach der Makroersetzung, die vollständige Übersetzung in SQL, die erhaltene Ergebnismenge sowie Ergebnisinformationen einer ggf. verwendeten Funktion im oberen Bereich der Benutzerschnittstelle angezeigt.

Anmerkung: Die implementierte Benutzeroberfläche bietet bisher keine Möglichkeit der Speicherung und nachträglichen Bearbeitung eingegebener Verknüpfungsregeln. Da dieses jedoch Grundlage für eine spätere Reinterpretation bzw. Anpassung bei der Aktualisierung auf einen neuen Planungsstand ist, ist die Benutzeroberfläche entsprechend zu erweitern. Dieses ist für die Entwicklung eines erweiterten Demoszenarios im Rah-

men des InPro[57]-Projektes geplant. Dabei wird eine Speicherung einer oder mehrerer Verknüpfungsregeln im Kontext eines Terminplanvorgangs angestrebt.

5.3.2 Übersetzung in SQL

Die Übersetzung von Ausdrücken der Verknüpfungssprache in SQL erfolgt mit einem mittels JavaCC und Java Tree Builder (JTB)[58] auf Basis der im Anhang A angegebenen Grammatik erstellten Compilers. Dieser erstellt beim Parsen des Sprachausdrucks einen Syntaxbaum, der anschließend nach dem Visitor-Konzept[59] analysiert wird. Die Knoten des Syntaxbaums repräsentieren Sprachelemente (Terminale und Non-Terminale) der Eingabe. Sie werden vom Visitor in der Reihenfolge „depth first“, d.h. von unten nach oben und somit der Sprachausdruck von innen nach außen abgearbeitet. Je nach angetroffenem Sprachelement übernimmt der Visitor direkt die Übersetzung in SQL oder stößt zuvor eine geometrische Auswertung innerhalb des 4D-BIM-Editors an. Die Abbildung von Konstrukten der Verknüpfungssprache auf SQL wird nachfolgend erläutert. Zielsystem ist dabei MS Access. Nähere Informationen zu SQL und MS Access findet man in [Kline u. a. 2004], [Marsch u. Fritze 1995], [Viescas 1989] und [Roman 2002].

Mengenausdrücke

Die Datenbankabfagen des Sprachinterpreters zur Auswertung von *setCreationTerms* beziehen sich stets auf die aktuell geladenen Modellversionen im Workspace und damit in der FROM-Klausel der SQL-Anweisung auf die _OP-Tabelle eines betreffenden Teilmodells Θ. Daten der Versionshistorie werden im Rahmen der Sprachinterpretation nicht genutzt. Mit einer Unteranfrage (Subquery) wird der Status in der _OS-Tabelle auf den Wert „wahr“ geprüft und damit sichergestellt, dass nur Objekte berücksichtigt werden, die Teil der aktuellen Modellversion sind. Es ergibt sich daher folgendes Grundgerüst für die erzeugte SQL-Anweisung eines *setCreationTerms* (siehe auch Datenbankschema in Anhang D und dessen Beschreibung in Kapitel 5.2.2):

```
SELECT DISTINCT P1 FROM Θ_OP
WHERE Objekt_ID IN (SELECT Objekt_ID FROM Θ_OS WHERE STATUS=true)
[AND P2] [AND P3] [AND P4];
```

Die eckigen Klammern markieren optionale Abfrageteile. Die Parameter *P1* und *P4* legen fest, welche Daten gemäß der *extract*-Anweisung des Sprachausdrucks extrahiert werden. Der Parameter *P2* fragt Ergebnisinformationen von räumlichen Anfragen (*setCreationFunctions*) aus einer temporären Tabelle ab, und Parameter *P3* bildet geschachtelte Mengenausdrücke bzw. Filter der Eingabe ab.

Extraktion

Die drei von der Verknüpfungssprache zur Verfügung gestellten Extraktionsarten werden gemäß Tabelle 5.3 in SQL übersetzt.

[57] www.inpro-project.eu
[58] http://compilers.cs.ucla.edu/jtb/
[59] siehe Online-Dokumentation des Java Tree Builders und [Metsker u. Wake 2006], [Cooper 1998]

Extraktion	Sprachausdruck	P1	P4
alle Attributnamen	-> allAttributes	`Attributname`	-
alle Attributwerte	-> allValues	`Wert`	-
Werte eines einzelnen Attributs z.B. der ID	-> ID	`Wert`	`Attributname=ID`

Tabelle 5.3: Übersetzung der Extraktion in SQL

Räumliche Anfragen

Räumliche Auswertungen sind in die Verknüpfungssprache mittels der beiden *setCreationFunctions inZoneRelation(set_ term, set_ term)* und *betweenAxisRelation(set_ term, set_ term)* eingebettet. Beide tragen nach der räumlichen Auswertung auf Seite des 4D-BIM-Edtors die IDs der positiv getesteten, d.h. innerhalb der Anfragezonen bzw. zwischen den Anfrageachsen liegenden CAD-Objekte, in eine Hilfstabelle namens `tempTable` ein. Diese Informationen werden anschließend in der SQL-Anweisung zur Selektion von Informationen aus dem Teilmodell der CAD-Objekte verwendet. Um die Ergebnisse mehrerer räumlicher Anfragen eines Sprachausdrucks zu unterscheiden, werden die Ergebnisse zusammen mit einer Nummer für das Auftreten der räumlichen Anfrage innerhalb des Sprachausdrucks gespeichert. Die Tabelle `tempTable` besitzt daher die beiden Spalten `nr` (Datentyp Zahl) und `Objekt_ID` (Datentyp Text). Tabelle 5.4 zeigt die Abfrage des Ergebnisses einer räumlichen Anfrage in SQL (beispielhaft für Anfrage-Nr. 1).

Parameter	ersetzt durch
Θ`_OP`	`CAD_OP`
P2	`Objekt_ID IN (SELECT Objekt_ID FROM tempTable WHERE nr=1)`

Tabelle 5.4: Einbettung der Ergebnisse einer räumlichen Anfrage in SQL

Der Name des abgefragten Teilmodells Θ und der Inhalt von Parameter *P2* können im Prinzip je *setCreationFunction* individuell gesetzt werden. Somit ist eine Erweiterung der Verknüpfungssprache mit weiteren *setCreationFunctions* für zusätzliche Funktionalitäten leicht möglich (z.B. um Anfragen vom Typ „in Achse A“ zu ermöglichen).

Filter

Filter F_i werden über den Parameter *P3* in das Grundgerüst der SQL-Anfrage eingebettet. Mehrere Filter können in der Verknüpfungssprache mit und (,) bzw. oder (|) kombiniert werden. In SQL wird dieses mit `AND` bzw. `OR` wie folgt übersetzt:

und: *P3* = ... F_{n-1} `AND` F_n
oder[60]: *P3* = (F_1 ... F_{n-1} `OR` F_n)

[60] In SQL bindet der Operator `OR` schwächer als der Operator `AND`. Daher muss für jedes `OR` ein alle vorhergehenden Filter einschließendes Klammerpaar gesetzt werden, um eine äquivalente Logik zur Verknüpfungssprache zu erreichen.

Darüber hinaus können im Sprachausdruck zur logischen Kombination von Filtern Klammern verwendet werden. Diese werden in SQL an gleicher Stelle übernommen.

Jeder einzelne Filter F_i entspricht einer Unteranfrage. Für die Filter *defined* und *undefined* zur Überprüfung der Existenz bzw. des Fehlens eines Attributes hat diese die Form:

defined: `Objekt_ID IN ( SELECT Objek_ID FROM` Θ`_OP WHERE Attributname='`a_n`' )`

undefined: `Objekt_ID NOT IN ( SELECT Objek_ID FROM` Θ`_OP WHERE Attributname='`a_n`' )`

Für alle anderen von der Anfragesprache zur Verfügung gestellten Filter hat sie die folgende Form, wobei F_a die für den Filter spezifische SQL-Anweisungen enthält:

`Objekt_ID IN ( SELECT Objek_ID FROM` Θ`_OP WHERE Attributname='`a_n`' AND` F_a `)`

Um bei Vergleichsoperationen zuverlässige Ergebnisse zu erhalten, ist unmittelbar vor der Durchführung des Vergleichs eine Konvertierung der im Workspace als Text gespeicherten Attributwerte in deren ursprünglichen Datentyp erforderlich. Auf den Datentyp kann anhand der vom Nutzer im Sprachausdruck angegebenen Vergleichswerte geschlossen werden. SQL stellt für die Konvertierung spezielle Funktionen zur Verfügung. Deren Verwendung ist in den Tabellen 5.5 und 5.6 für die Filterung mittels Vergleichsoperatoren gezeigt.
Für die Filter *in* und *notin* kann zum Vergleich entweder eine *setElementList*, ein *setCreationTerm* oder ein Intervall verwendet werden. Die Übersetzung dieser Filter in SQL zeigt Tablle 5.7. Wird ein *setCreationTerm* zum Vergleich verwendet, so enthält F_a eine SQL-Anweisung entsprechend dem Grundgerüst (Schachtelung von Mengenausdrücken).

Datentyp	Sprachausdruck	F_a
Ganzzahl	a_n=3	`CINT(Wert)=3`
Fließkommazahl	a_n=3.2	`CDbl(Wert)=3.2`
Datum	a_n=01.09.2009	`CDate(Wert)=#09/01/2009#`
Text	a_n="Beton"	`Wert='Beton'`

Tabelle 5.5: Übersetzung des Operators = in SQL bei Verwendung verschiedener Datentypen (analog für <,>,<=,>=)

Datentyp	Sprachausdruck	F_a
Ganzzahl	a_n!=3	`CINT(Wert)<>3`
Fließkommazahl	a_n!=3.2	`CDbl(Wert)<>3.2`
Datum	a_n!=01.09.2009	`CDate(Wert)<>#09/01/2009#`
Text	a_n!="Beton"	`Wert<>'Beton'`

Tabelle 5.6: Übersetzung des Operators != in SQL bei Verwendung verschiedener Datentypen

Sprachkonstrukt	Beispiel	F_a
in Elementliste	a_n in {2.1,3.0,4.3}	`CDbl(Wert) IN (2.1,3.0,4.3)`
in Menge	a_n in *setCreationTerm*	`Wert IN (`*SQL setCreationTerm*`)`
nicht in Menge	a_n in *setCreationTerm*	`Wert NOT IN (`*SQL setCreationTerm*`)`
in Interval	a_n in [4, 9]	`CINT(Wert) BETWEEN 4 AND 9`
nicht in Interval	a_n `notin` [`04.02.2009, 11.02.2009`]	`CDate(Wert) NOT BETWEEN #02/04/2009# AND #02/11/2009#`

Tabelle 5.7: Übersetzung der Operatoren *in* und *notin* in SQL (jeweils analog für andere Datentypen)

Reguläre Ausdrücke

Bei der Angabe von Attributnamen und -werten können in der Verknüpfungssprache die folgenden Platzhalter verwendet werden:

* für eine unbestimmte Anzahl beliebiger Zeichen,
? für genau ein beliebiges Zeichen.

Deren Entsprechungen in SQL sind die Zeichen `%` und `_` . In Tabelle 5.8 ist die Verwendung von Platzhaltern in der Verknüpfungssprache und der SQL-Übersetzung gegenübergestellt.

Zweck der Nutzung	Sprachausdruck	SQL-Anweisung
pos. Filter für ein Attribut	a_n="*Beton*"	`Wert LIKE '%Beton%'`
neg. Filter für ein Attribut	a_n!="Vorgang?"	`Wert NOT LIKE 'Vorgang_'`
Existenz einer geordneten Attributgruppe	Material* defined	`Attributname LIKE 'Material%'`
Filtern einer geordneten Attributgruppe	Material*="Beton"	`Attributname LIKE 'Material%' AND Wert='Beton'`
Extraktion einer geordneten Attributgruppe	-> CAD_*	`Attributname LIKE CAD_%`

Tabelle 5.8: Übersetzung regulärer Ausdrücke in SQL

Beispiele

Zur Verdeutlichung der Umsetzung sind nachfolgend einige Ausdrücke der Verknüpfungssprache und die entsprechenden Übersetzungen in SQL aufgeführt:

Beispiel 1: Ausgabe der Materialien von CAD-Objekten ohne Schichtenaufbau

{ CAD : Schichten undefined -> Material }

```
SELECT DISTINCT Wert FROM CAD_OP
WHERE Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OS WHERE STATUS=true )
AND   Objekt_ID NOT IN ( SELECT Objekt_ID FROM CAD_OP
                         WHERE Attributname='Schichten' )
AND Attributname='Material';
```

Beispiel 2: Ausgabe der IDs aller Wände aus Beton

{CAD: Typ="Wand", Material="Beton" -> ID}

```
SELECT DISTINCT Wert FROM CAD_OP
WHERE Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OS WHERE STATUS=true )
AND   Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OP
                     WHERE Attributname='Typ'
                     AND Wert='Wand' )
AND   Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OP
                     WHERE Attributname='Material'
                     AND Wert='Beton' )
AND   Attributname='ID';
```

Beispiel 3: Ausgabe der IDs aller CAD-Objekte, die einen Schichtenaufbau mit 2 bis 4 Schichten besitzen, wovon eine aus Mauerwerk besteht

{ CAD : Schichten in [2 , 4] , Material* = "Mauerwerk" -> ID }

```
SELECT DISTINCT Wert FROM CAD_OP
WHERE Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OS WHERE STATUS=true )
AND   Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OP
                     WHERE Attributname='Schichten'
                     AND CINT(Wert) BETWEEN 2 AND 4 )
AND   Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OP
                     WHERE Attributname LIKE 'Material%'
                     AND Wert='Mauerwerk' )
AND   Attributname='ID';
```

Beispiel 4: Ausgabe der IDs aller Wände und Stützen innerhalb von Zone_1

{ inZoneRelation({ CAD : Typ = "Wand" | Typ = "Stütze" -> ID } ,
{ "Zone_1" }) : -> ID }

Zunächst werden die IDs der Wände und Stützen bestimmt:

```
SELECT DISTINCT Wert FROM CAD_OP
WHERE Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OS WHERE STATUS=true )
AND ( Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OP
                     WHERE Attributname='Typ'
                     AND Wert='Wand' )
OR    Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OP
                     WHERE Attributname='Typ'
                     AND Wert='Stütze' ) )
AND   Attributname='ID';
```

Mit den so bestimmten CAD-Objekten erfolgt die räumliche Auswertung im 4D-BIM-Editor. Das Ergebnis wird in die Tabelle `tempTable` geschrieben. Die Abfrage der Ergebnisses geschieht wie folgt:

```
SELECT DISTINCT Wert FROM CAD_OP
WHERE Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OS WHERE STATUS=true )
AND   Objekt_ID IN ( Select Objekt_ID FROM tempTable WHERE nr=1 )
AND   Attributname='ID';
```

Beispiel 5: Ausgabe der Materialien aller CAD-Objekte, die mit Terminplanvorgängen verknüpft sind, die vor dem 10.02.2009 beginnen

{ CAD : ID in { link4D : task in { schedule : start < 10.02.2009 -> ID } -> CAD* }
-> Material }

```
SELECT DISTINCT Wert FROM CAD_OP
WHERE Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OS WHERE STATUS=true )
AND   Objekt_ID IN ( SELECT Objekt_ID FROM CAD_OP
                     WHERE Attributname='ID'
                     AND Wert IN (
SELECT DISTINCT Wert FROM link4D_OP
WHERE Objekt_ID IN ( SELECT Objekt_ID FROM link4D_OS WHERE STATUS=true )
AND   Objekt_ID IN ( SELECT Objekt_ID FROM link4D_OP
                     WHERE Attributname='task'
                     AND Wert IN (
SELECT DISTINCT Wert FROM schedule_OP
WHERE Objekt_ID IN ( SELECT Objekt_ID FROM schedule_OS WHERE STATUS=true )
AND   Objekt_ID IN ( SELECT Objekt_ID FROM schedule_OP
                     WHERE Attributname='start'
                     AND CDate(Wert)<#02/10/2009# )
```

```
AND   Attributname='ID' ) )
AND   Attributname LIKE 'CAD%' ) )
AND   Attributname='Material';
```

5.3.3 Umsetzung von Funktionen

Die Ergebniselemente der SQL-Abfrage auf der Workspacesdatenbank werden in einem `HashSet<String>` namens `results` abgespeichert. Zur Weiterverarbeitung der Elemente stellt die Verknüpfungssprache mehrere Funktionen zur Verfügung. Deren Ausführung erfolgt Java-seitig gemäß Auflistung 5.1 bis 5.7.

Auflistung 5.1: Umsetzung der Funktion *max* in Java

```
double max;
try{
  max = Double.parseDouble(Collections.max(results,
    new Comparator<String>(){
      public int compare (String s1, String s2){
        Double d1 = Double.parseDouble(s1.replace(",", "."));
        Double d2 = Double.parseDouble(s2.replace(",", "."));
        return d1.compareTo(d2);
      }
    }
  ).replace(",", "."));
}
catch(NumberFormatException e){}
```

Auflistung 5.2: Umsetzung der Funktion *min* in Java

```
double min;
try{
  min = Double.parseDouble(Collections.max(results,
    new Comparator<String>(){
      public int compare (String s1, String s2){
        Double d1 = Double.parseDouble(s1.replace(",", "."));
        Double d2 = Double.parseDouble(s2.replace(",", "."));
        return d1.compareTo(d2);
      }
  }).replace(",", "."));
}
catch(NumberFormatException e){}
```

Auflistung 5.3: Umsetzung der Funktion *sum* in Java

```
double sum = 0.;
try{
  for(String s : results){
    sum += Double.parseDouble(s.replace(",", "."));
  }
}
catch(NumberFormatException e){}
```

Auflistung 5.4: Umsetzung der Funktion *count* in Java

```
int count = results.size();
```

Auflistung 5.5: Umsetzung der Funktion *Select* in Java

```
for(String ID : results){
  bimEditor.select(ID);
}
```

Auflistung 5.6: Umsetzung der Funktion *linkGeometrie* in Java

```
String taskID = p1.tokenImage;
String visualParameterID = p3.tokenImage;
bimEditor.linkGeometrie(taskID, results, visualParameterID);
//results enthält die IDs der zu verknüpfenden CAD-Objekte
```

Auflistung 5.7: Umsetzung der Funktion *linkQuantities* in Java

```
String taskID = p2.tokenImage;
String choosenPerformanceFactor = p4.tokenImage;
bimEditor.linkQuantities(taskID, results1, results2,
   choosenPerformanceFactror);
/* results1 enthält die IDs der zu verknüpfenden
   Mengenobjekte und results2 den Schlüssel für
   einen Eintrag in der Aufwandswertedatenbank */
```

5.4 Umsetzung räumlicher Anfragen

Die geometrischen Algorithmen aus Kapitel 4.7 wurden direkt auf den in Java3D für die Geometriebeschreibung verwendeten Datenstrukturen umgesetzt. Die Definition von Zonen und Achsen auf Ebenen erfolgt innerhalb des 4D-BIM-Editor interaktiv in der 3D-Ansicht. Dabei können für ein Modell mehrere Ebenen verwendet werden (z.B. eine je Geschoss). Sie werden ebenfalls interaktiv eingegeben und besitzen zur besseren Übersicht eine begrenzte Ausdehnung (siehe Abbildung 5.7). Bei der Eingabe von Zonen und Achsen wird für eine quasi zweidimensionale Eingabe in die Isometrieansicht der zugeordneten Ebene gewechselt. Die Normale einer Ebene entspricht der Extrusionsrichtung von Achsen und Zonen. Ebenen, Zonen und Achsen können vom Anwender in einer Baumansicht verwaltet und in ihrer Darstellung angepasst werden (Abbildung 5.7). Um eine Nutzung räumlicher Anfragen auch unabhängig von der Verknüpfungssprache zu ermöglichen, stehen im 4D-BIM-Editor entsprechende Steuerelemente für die Ausführung von Zonen- und Achsanfragen zur Verfügung. Ebenso steht ein Steuerelement für die Verschiebung von Bauwerksgeschossen entsprechend einer treppenförmigen Explosionsansicht zur Verfügung.
Abbildung 5.6 zeigt die Ergebnisse der zweidimensionalen Berechnung des Zwischenbereichs zweier Achsen für ausgewählte Fälle in einer zusätzlichen Testanwendung mit erweiterten Visualisierungsmöglichkeiten für Knoten, Kanten und Dreiecke. Gelb dargestellt sind Bereiche, die nach Prüfung nicht als Außentasche klassifiziert und entfernt wurden, da an ihrem Rand bzw. im Innern eine Kante der zweiten Achse gefunden wurde (siehe hierzu auch Kapitel 4.7.2).
Abbildung 5.8 zeigt die Durchführung einer Objektteilung für eine Deckenplatte.

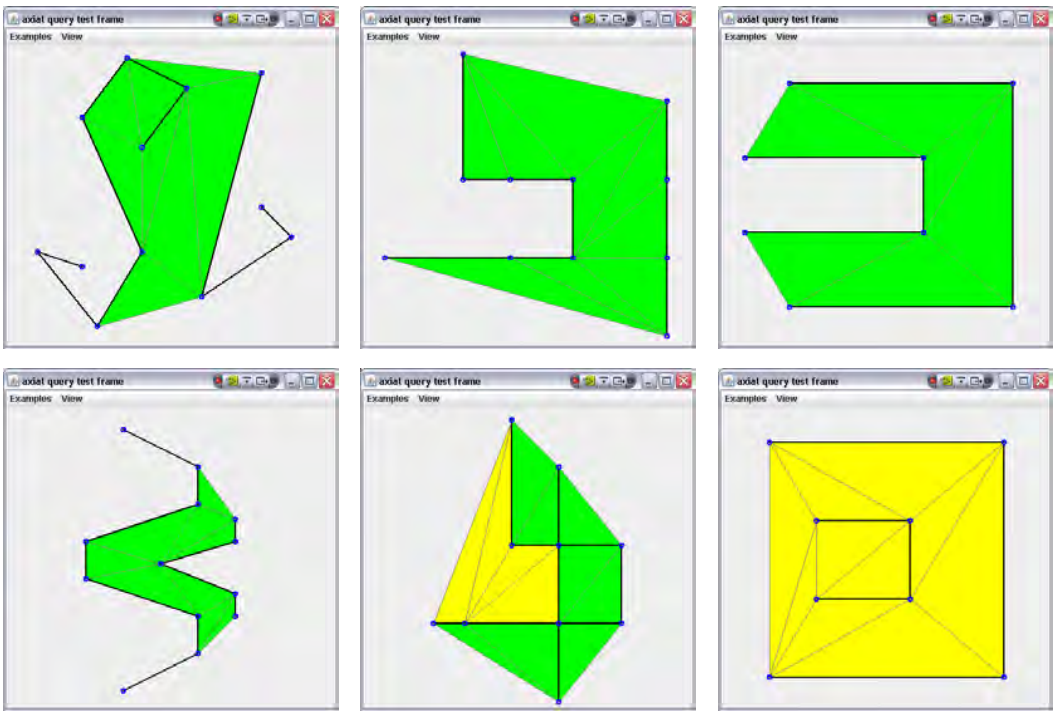


Abbildung 5.6: Berechnung des Zwischenbereichs zweier Achsen nach der relaxierten Definition.

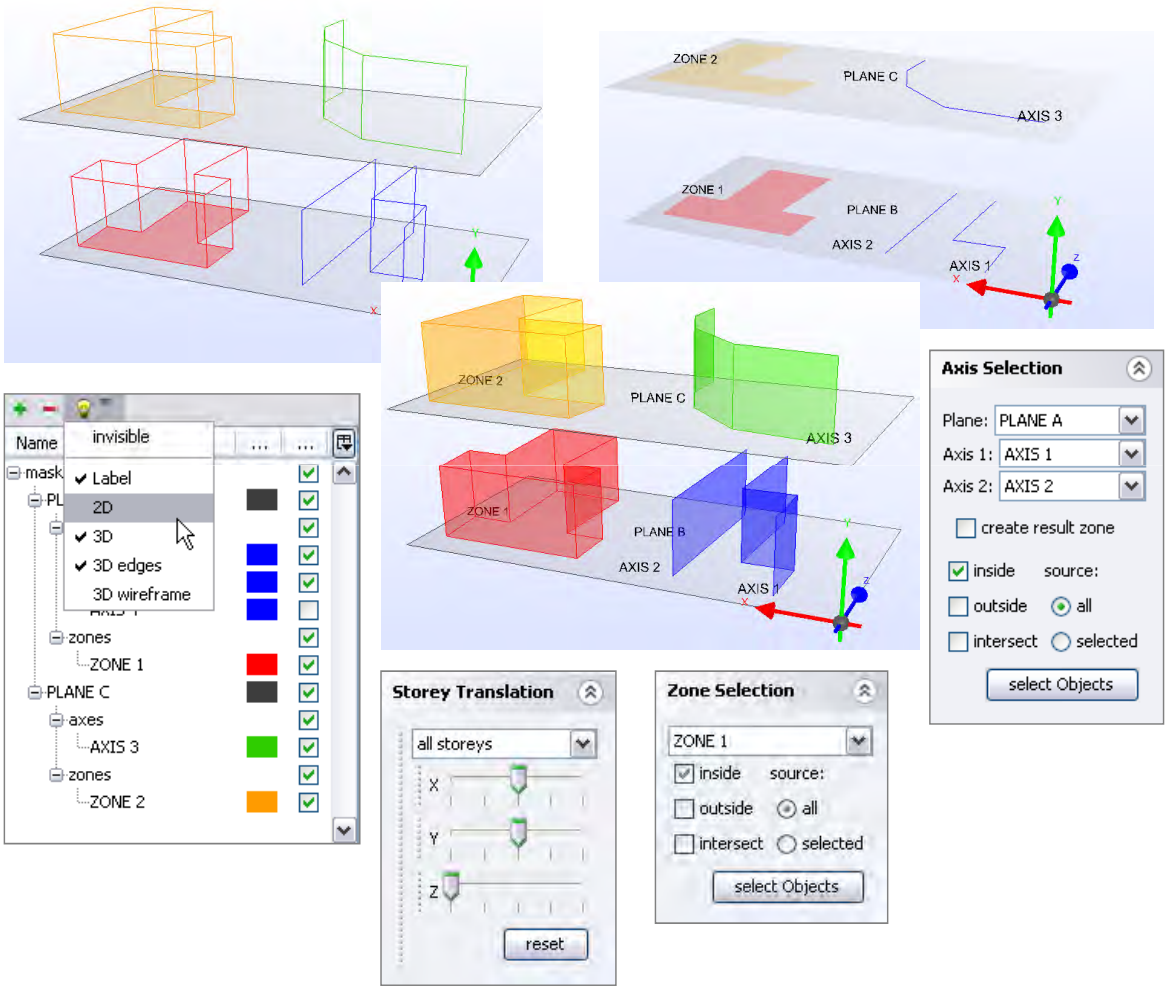


Abbildung 5.7: ebenenbezogene Eingabe und Verwaltung von Zonen und Achsen (verschiedene Darstellungsarten)

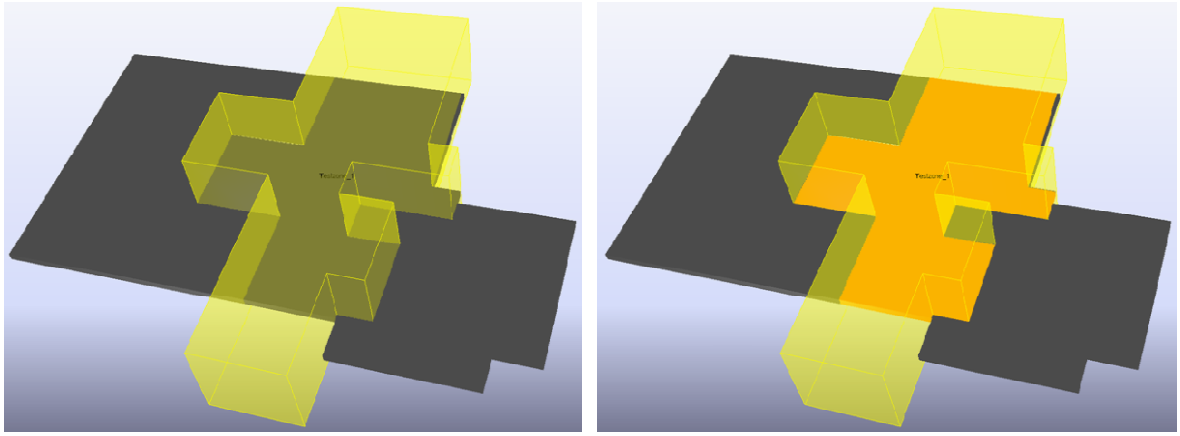


Abbildung 5.8: Objektteilung auf Basis einer Anfragezone (links: ungeteilte Deckenplatte, rechts: geteilte Deckenplatte mit selektiertem Innenteil)

5.5 Projekt und Paketstruktur

Das Softwarepaket wurde in folgende Module gegliedert:

1. **Java_Ifc_Toolbox**
   Lesen u. Schreiben von IFC-Dateien im STEP-Format, enthält die Java-Klassen der IFC-Objekte sowie die Klasse `IfcModel`
2. **Netzgenerator**
   Implementierung der eingeschränkten Triangulation nach der Kettenmethode aus [Preparata u. Shamos 1985]
3. **GM3D** (nutzt 2.)
   Geometriepaket mit Grundobjekten für die Visualisierung, enthält einen Boolean Modeller und Funktionen zum Extrudieren von Körpern aus Flächen.
4. **Java3D_Ifc_Loader** (nutzt 1., 3.)
   Interpretation der im IFC-Modell enthaltenen Geometrie und Überführung in einen Java3D-Szenengraphen
5. **WorkspaceManager**
   Versionierung und Datenbankkommunikation auf Basis von JDBC
6. **4D-BIM-Editor** (nutzt 1., 2., 3., 4., 5., 7.)
   3D-, Baum- und Tabellenansichten bzw. Editoren, 4D-Funktionalität, COM-Schnittstelle nach MS Project, Steuerung der anderen Pakete
7. **Sprachinterpreter** (nutzt 5.) Interpreter für die Verknüpfungssprache

Für alle genannten Pakete ist eine Bereitstellung zum Download im Internet unter www.openifctools.com geplant. Der Netzgenerator ist zudem für die Veröffentlichung als Applet durch die Universität Bonn unter www.geometrylab.de vorgesehen.

6 Beispiele und Demonstration

Die Umsetzung der vorgestellten Konzepte wird nachfolgend an vier Beispielen gezeigt. Zunächst werden das Prinzip sowie die verwendeten Daten detailliert an einem Minimalbeispiel erläutert. Die danach folgenden Praxisbeispiele konzentrieren sich auf die Anwendung der räumlichen Anfragen. Sie sollen einen Eindruck über Anwendungsfälle und die dabei auftretenden Antwortzeiten vermitteln.

6.1 Minimalbeispiel

In diesem Beispiel wird das Erstellen einer Wand betrachtet. Dem Terminplaner wird eine IFC-Datei zur Verfügung gestellt, die folgende Ausgangsinformationen enthält:

- die Geometrie der Wand,
- die Basismengen Breite, Länge, Höhe, Volumen, Seiten- und Aufstandsfläche,
- eine Kostenaufstellung für die vorgesehenen Arbeiten, getrennt in Rohbau- und Ausbauarbeiten.

Auf Basis der in der Kostenaufstellung aufgeführten Leistungen erstellt der Terminplaner im Terminplanungsprogramm (hier MS Project) zunächst eine Prozesskette, die er über die COM-Schnittstelle an den 4D-BIM-Editor übergibt. Er entscheidet sich aufgrund der Abmaße der Wand und der auf der Baustelle zur Verfügung stehenden Ressourcen dafür, den Rohbau in drei Bauabschnitten zu erstellen. Der Ausbau wird jedoch als ein Vorgang betrachtet. Im 4D-BIM-Editor erstellt er entsprechend eine Ebene mit vier Achsen, die für die Unterteilung der Wand in einen 4m und zwei 3m lange Abschnitte genutzt werden. Zusätzlich erzeugt er zwei Sätze von Visualisierungsparametern für die 4D-Simulation (Rohbau=blau, Ausbau=rot). In der Berechnungsformel für die Dauer der Terminplanvorgänge verwendet er bei den Rohbauvorgängen das Volumen des jeweiligen Wandabschnitts und für den Ausbauvorgang die Gesamtausbaukosten. Für die Erstellung der Verknüpfungsregeln verwendet er drei vorher definierte Makros:
Definition der Macros:

```
#Link4D(t,a1,a2,v):=
     linkg("t",{betweenAxisRelation({CAD:->ID},
     {"AXIS a1","AXIS a2"}):->ID},"v")
#LinkVol(t,a1,a2,f):=
     linkq("t", sum({BaseQ: CADid in {betweenAxisRelation(
     {CAD:->ID},{"AXIS a1","AXIS a2"}):->ID}->GrossVolume}),{},f)
#LinkCosts(t,p,f):=
     linkq("t",{Costs:pos=2->value},{},f)
```

Alle drei Makros weisen Parameter auf. Das Makro `#Link4D` verknüpft den Terminplanvorgang t mit den CAD-Objekten zwischen den Achsen *a1* und *a2*. Es wird dabei der Visualisierungsparameter v verwendet. Die Anfrage in Makro `#LinkVol` erstreckt sich über zwei Teilmodelle, um den Terminplanvorgang t mit der Summe des Volumens aller CAD-Objekte, die zwischen den Achsen *a1* und *a2* liegen, zu verknüpfen. Als Aufwandswert wird dabei der Faktor f verwendet[61]. Die CAD-Objekte und Basismengen liegen im Workspace jeweils in einem eigenen Teilmodell vor. Die Basismengen (Teilmodell *BaseQ*) sind über das Attribut *CADid* den CAD-Objekten (Teilmodell *CAD*) zugeordnet. Das Makro `#LinkCosts` weist dem Terminplanvorgang t den zur Positionsnummer p gehörenden Kostenwert zu.[62]
Mit Hilfe der Makros werden die Verknüpfungen wie folgt erstellt:

```
#Link4D(1.1, 7, 8,Rohbau)
#Link4D(1.2, 8, 9,Rohbau)
#Link4D(1.3, 9,10,Rohbau)
#Link4D(  2, 7,10,Ausbau)

#LinkVol(1.1, 7, 8, 2.5)
#LinkVol(1.2, 8, 9, 2.5)
#LinkVol(1.3, 9,10, 2.5)
#LinkCosts(2, 2, 0.02)
```

Es wurden die Aufwandswerte 2,5 Std/m3 Beton für den Rohbau und 0,02 Std/€ für den Ausbau angenommen. Das Ergebnis der Berechnung der Vorgangsdauer wird intern auf volle Arbeitstage (à 8 Std.) gerundet. Durch die Ausführung dieser Regeln erfolgt Folgendes:

- die Wand wird geteilt,
- die Mengen der Teilwände werden berechnet[63],
- die Dauer der Terminplanvorgänge wird aktualisiert,
- die 4D-Simulation wird erstellt.

Die Ergebnisse zeigt Abbildung 6.1. Anschließend wird das IFC-Modell mit den neuen Informationen (geteilte Wandobjekte inkl. Mengen, Terminplan, 4D) ergänzt und der Projektleitung bzw. Baustelle zur Verfügung gestellt. Die Auflistungen 6.1 bis 6.6 zeigen Auszüge aus der IFC-Datei, die insgesamt 390 Zeilen (Objekte) enthält. Ergeben sich im Planungsverlauf Änderungen, z.B. in der Wandlänge, so muss der Terminplaner

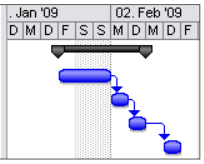
[61] Der dritte Parameter der Funktion *linkq* dient normalerweise der Auswahl eines Vorschlagswertes für den Aufwand aus einer Datenbank. In diesem Beispiel wurde darauf verzichtet und daher der Parameter zu {} (leere Menge) gesetzt.

[62] Da keine Funktion *linkc* implementiert wurde, die Kostenwerte mit Terminplanvorgängen verknüpft, wird stattdessen *linkq* verwendet. Kosten und Aufwandswert stehen dadurch jedoch innerhalb von MS Project in den Spalten für Mengen.

[63] Die Berechnung von Mengen ist im 4D-BIM-Editor bisher nicht implementiert. Die Mengenwerte der Teilobjekte wurden daher im Workspace manuell ergänzt.

MS Project

	Vorgangsname	Dauer	Anfang	Ende	Vorgänger	Nachfolger	Text30
1	**– Rohbau Wand**	**3 Tage?**	**Fr 30.01.09 08:00**	**Mi 04.02.09 00:00**			**1DkjVBCLyNHRTVleohdiP3**
2	Betonierabschnitt 1	1 Tag?	Fr 30.01.09 08:00	Mo 02.02.09 00:00		3	31O4VabBWVH87F4y9KdyXp
3	Betonierabschnitt 2	1 Tag?	Mo 02.02.09 08:00	Di 03.02.09 00:00	2	4	3lfXUajup2HytZuAenSRYf
4	Betonierabschnitt 3	1 Tag?	Di 03.02.09 08:00	Mi 04.02.09 00:00	3	5	3ov5vzv5ydHCqV3Wb_mZ9Y
5	Ausbauarbeiten	1 Tag?	Do 05.02.09 08:00	Fr 06.02.09 00:00	4		34h$C_1nWAGPc9UJs3092M



4D-BIM-Editor

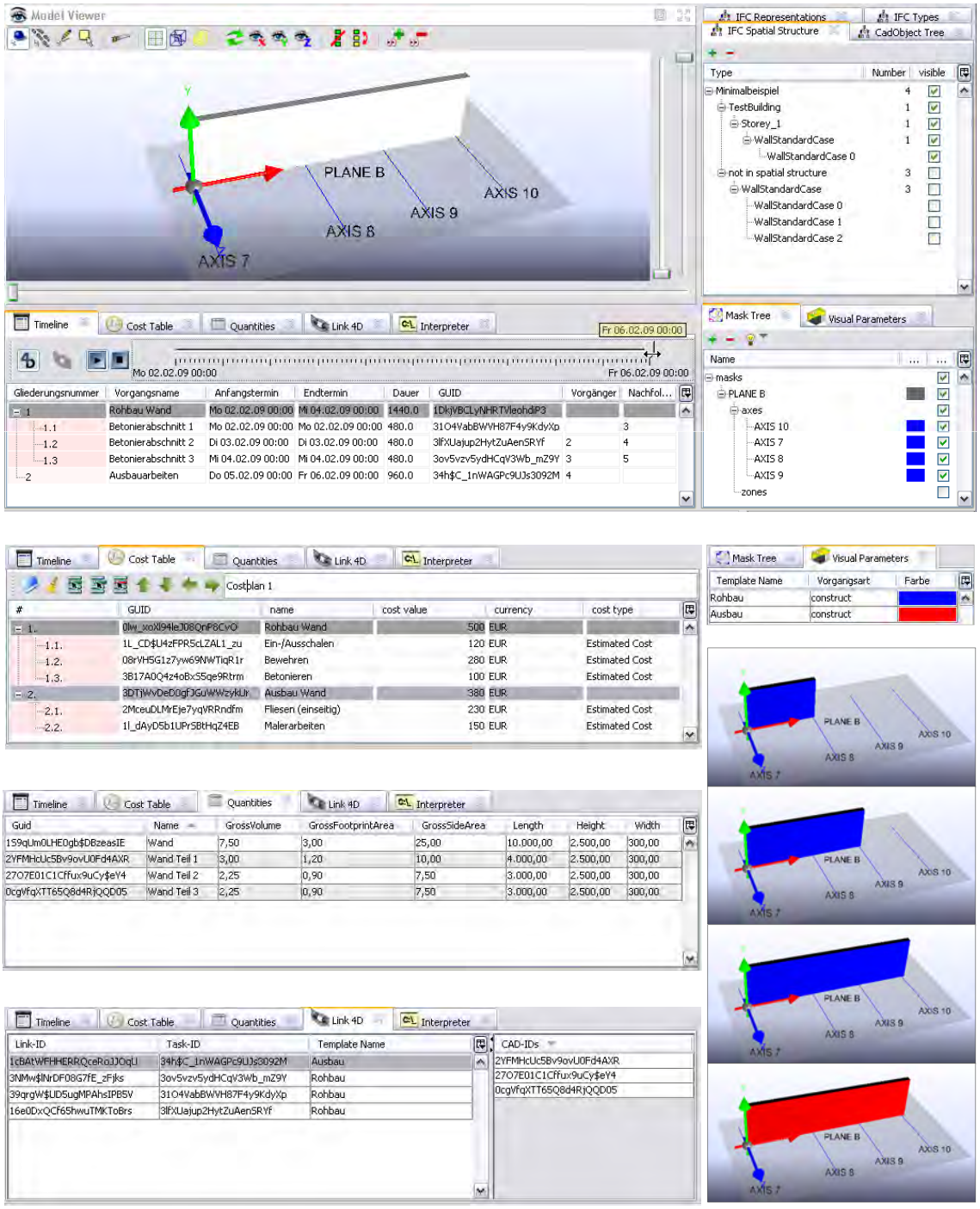


Abbildung 6.1: Minimalbeispiel: Wanderstellung in mehreren Bauabschnitten

lediglich die Lage der Achsen anpassen und die Verknüpfungsregeln neu interpretieren lassen, um Terminplan und 4D-Simulation zu aktualisieren. Änderungen in der Ausbauqualität haben zudem über die Kosten Einfluss auf die geplante Dauer des Ausbaus. Die Verknüpfungssprache kann auch nachträglich zum Analysieren des verknüpften Modells verwendet werden. So können z.B. auf der Baustelle beim Auftreten von Lieferverzögerungen für bestimmte Fliesen alle Vorgänge ausgegeben werden, die mit Mengen oder Kosten dieses Fliesentyps verknüpft sind und so die Auswirkungen auf den Terminplan abgeschätzt werden.

Auflistung 6.1: Abbildung des Terminplans in IFC

```
#257= IFCCALENDARDATE(4,2,2009);
#258= IFCCALENDARDATE(4,2,2009);
#259= IFCSCHEDULETIMECONTROL('3QmBjKrR7JJBfsxsUWGxGw',#252,'Ausbauarbeiten',$,$,$,$,
      $,#260,$,$,$,#261,960.0,$,$,$,$,$,$,$,$);
#260= IFCCALENDARDATE(5,2,2009);
#261= IFCCALENDARDATE(6,2,2009);
#262= IFCRELASSIGNSTASKS('0BkRfy0ha8JPCGz1Pqk92T',#252,'Betonierabschnitt 1',$
      ,(#263),.PROCESS.,#264,#267);
#263= IFCTASK('3104VabBWVH87F4y9KdyXp',#252,'Betonierabschnitt 1','1.1',$,'2','
      NOTDEFINED',$,.F.,$);
#264= IFCWORKSCHEDULE('0Spn8nAZh2JyjyPMR0pYb4',#252,$,'4D work schedule',$,'not
      known',#265,$,'4D modelling',$,$,#266,$,$,$);
#265= IFCCALENDARDATE(6,2,2009);
#266= IFCCALENDARDATE(6,2,2009);
#267= IFCSCHEDULETIMECONTROL('31UziW2swBISmy_zRiz$Ex',#252,'Betonierabschnitt 1',$,$
      ,$,$,$,#268,$,$,$,#269,480.0,$,$,$,$,$,$,$,$,$);
#268= IFCCALENDARDATE(2,2,2009);
#269= IFCCALENDARDATE(2,2,2009);
#270= IFCCALENDARDATE(2,2,2009);
#271= IFCTASK('3lfXUajup2HytZuAenSRYf',#252,'Betonierabschnitt 2','1.2',$,'3','
      NOTDEFINED',$,.F.,$);
#272= IFCRELSEQUENCE('2$_3qPuwjeJRP2OdEXihZk',#252,$,$,#271,#273,0.0, .FINISH_FINISH
      .);
#273= IFCTASK('3ov5vzv5ydHCqV3Wb_mZ9Y',#252,'Betonierabschnitt 3','1.3',$,'4','
      NOTDEFINED',$,.F.,$);
#274= IFCCALENDARDATE(3,2,2009);
#275= IFCSCHEDULETIMECONTROL('3_UwGVsWDpJe5rAuATcMMJ',#252,'Betonierabschnitt 2',$,$
      ,$,$,$,#276,$,$,$,#274,480.0,$,$,$,$,$,$,$,$,$);
#276= IFCCALENDARDATE(3,2,2009);
#277= IFCCALENDARDATE(4,2,2009);
#278= IFCRELASSIGNSTASKS('0CXqfbHj$lGP32jd1n0Otn',#252,'Betonierabschnitt 2',$
      ,(#271),.PROCESS.,#264,#275);
#279= IFCTASK('34h$C_1nWAGPc9UJs3092M',#252,'Ausbauarbeiten','2',$,'5','NOTDEFINED',
      $,.T.,$);
#280= IFCRELNESTS('1k5AJbg8qLJ8HD1lMBlQuT',#252,$,$,#281,(#273,#271,#263));
#281= IFCTASK('1DkjVBCLyNHRTVleohdiP3',#252,'Rohbau Wand','1',$,'1','NOTDEFINED',$,.
      T.,$);
#282= IFCRELASSIGNSTASKS('1qNrurDdQLIh4lKjfvgBRw',#252,'Rohbau Wand',$,(#281),.
      PROCESS.,#264,#283);
#283= IFCSCHEDULETIMECONTROL('3vUBICPLTnH8WpKQEVfb72',#252,'Rohbau Wand',$,$,$,$,$
      ,#270,$,$,$,#277,1440.0,$,$,$,$,$,$,$,$,$);
#284= IFCRELASSIGNSTASKS('2Ps66jjj2LKeHEDIotvGQJ',#252,'Ausbauarbeiten',$,(#279),.
      PROCESS.,#264,#259);
#285= IFCRELSEQUENCE('2ptX8_LhIYKvOQK6mx$Rt0',#252,$,$,#263,#271,0.0,.FINISH_FINISH
      .);
#286= IFCRELASSIGNSTASKS('1pYdf1HhBqGQhGFtJ7VTC7',#252,'Betonierabschnitt 3',$
      ,(#273),.PROCESS.,#264,#251);
#287= IFCRELSEQUENCE('3aU5Oh2ewwJ8FVd5MTvGP6',#252,$,$,#273,#279,0.0,.FINISH_FINISH
      .);
```

Auflistung 6.2: Abbildung der Verknüpfungen zwischen Original- und Teilobjekten und für die 4D-Simulation in IFC

```
#69= IFCWALLSTANDARDCASE('2YFMHcUc5Bv9ovU0Fd4AXR',#2,'Wand Teil 1',$,$,#30,#31,$);
#115= IFCWALLSTANDARDCASE('27O7E01C1Cffux9uCy$eY4',#2,'Wand Teil 2',$,$,#80,#81,$);
#161= IFCWALLSTANDARDCASE('0cgVfqXTT65Q8d4RjQQD05',#2,'Wand Teil 3',$,$,#126,#127,$)
      ;
#235= IFCWALLSTANDARDCASE('1S9qUm0LHE0gb$DBzeasIE',#211,'Wand',$,$,#236,#223,$);
```

```
#184= IFCRELNESTS('0J$yGtHBD12v72y4qFNR01',#2,'Splitting of wall #235',$
      ,#235,(#161,#69,#115));

#288= IFCRELASSIGNSTOPROCESS('0J$yGtHBD12v72y4qFRR90',#252,$,$,(#69),.PRODUCT.,#263,
      $);
#289= IFCRELASSIGNSTOPROCESS('0J$yGtHBD12v72y4qFRR91',#252,$,$,(#115),.PRODUCT
      .,#271,$);
#290= IFCRELASSIGNSTOPROCESS('0J$yGtHBD12v72y4qFRR92',#252,$,$,(#161),.PRODUCT
      .,#273,$);
#291= IFCRELASSIGNSTOPROCESS('0J$yGtHBD12v72y4qFRR99',#252,$,$,(#161,#69,#115),.
      PRODUCT.,#279,$);
```

Auflistung 6.3: Abbildung eines Satzes von Visualisierungsparametern für die 4D-Simulation in IFC

```
#292= IFCRELDEFINESBYPROPERTIES('2K5FvWlVz8BewqgTQUfMYL',#252,'Template 1',$
      ,(#273,#271,#263),#293);
#293= IFCPROPERTYSET('09O00kRDDDnuATp7E65iLQ',#252,'4DVisualizationParameter','
      Rohbau',(#295,#299,#297,#294,#298,#296));
#294= IFCPROPERTYSINGLEVALUE('Visualization',$,IFCBOOLEAN(.T.),$);
#295= IFCPROPERTYSINGLEVALUE('RGBElementColour_Red',$,IFCINTEGER(0),$);
#296= IFCPROPERTYSINGLEVALUE('RGBElementColour_Green',$,IFCINTEGER(0),$);
#297= IFCPROPERTYSINGLEVALUE('RGBElementColour_Blue',$,IFCINTEGER(255),$);
#298= IFCPROPERTYSINGLEVALUE('RGBElementColour_Transparency',$,IFCINTEGER(50),$);
#299= IFCPROPERTYENUMERATEDVALUE('ConstructionType',$,(IFCLABEL('CONSTRUCT')),#300);
#300= IFCPROPERTYENUMERATION('ConstructionTypeEnumeration',(IFCLABEL('CONSTRUCT'),
      IFCLABEL('TEMPORARY'),
      IFCLABEL('DEMOLISH'),IFCLABEL('START_PERIOD'),IFCLABEL('MIDDLE_PERIOD'),
            IFCLABEL('END_PERIOD')),$);
```

Auflistung 6.4: Abbildung der Basismengen in IFC

```
#309= IFCRELDEFINESBYPROPERTIES('1sb$t0hX988ObgwlqB1gJj',#2,$,$,(#69),#310);
#310= IFCELEMENTQUANTITY('0L_Y4oJAH1WRBP983w3v_J',#2,'BaseQuantities',$,$
      ,(#314,#311,#316,#312,#313,#315));
#311= IFCQUANTITYLENGTH('Height',$,$,2500.0);
#312= IFCQUANTITYLENGTH('Length',$,$,4000.0);
#313= IFCQUANTITYLENGTH('Width',$,$,300.0);
#314= IFCQUANTITYAREA('GrossFootprintArea',$,$,1.2);
#315= IFCQUANTITYAREA('GrossSideArea',$,$,10.0);
#316= IFCQUANTITYVOLUME('GrossVolume',$,$,3.0);
```

Auflistung 6.5: Abbildung der Kostenaufstellung in IFC

```
#341= IFCCOSTSCHEDULE('2OP0ozhN9ESACznc31O2fC',#2,$,$,$,$,$,$,$,$,$,'Costplan 1',.
      COSTPLAN.);
#342= IFCRELSCHEDULESCOSTITEMS('2mYyoRm458iguzkbrZ8g5G',#2,$,$,(#343,#348),$,#341);

#343= IFCCOSTITEM('0lw-xoXl94leJ0iQnp8CvO',#2,'1.','Rohbau Wand',$);
#344= IFCRELNESTS('0poIWgRnz2YvDQx3QbV3PS',#2,$,$,#343,(#345,#346,#347));
#345= IFCCOSTITEM('1L_CD$U4zFPR5cLZAL1_zu',#2,'1.1.','Ein-/Ausschalen',$);
#346= IFCCOSTITEM('08rVH5G1z7yw69NWTiqR1r',#2,'1.2.','Bewehren',$);
#347= IFCCOSTITEM('3B17A0Q4z4oBxS5qe9Rtrm',#2,'1.3.','Betonieren',$);

#348= IFCCOSTITEM('3DTjWvDeD0gfJGuWWzykUr',#2,'2.','Ausbau Wand',$);
#349= IFCRELNESTS('2AoAoLqAX42A2HhU43nnQO',#2,$,$,#348,(#351,#350));
#350= IFCCOSTITEM('2MceuDLMrEje7yqVRRndfm',#2,'2.1.','Fliesen (einseitig)',$);
#351= IFCCOSTITEM('1l_dayD5b1UPrSBtHqZ4EB',#2,'2.2.','Malerarbeiten',$);

#352= IFCRELASSOCIATESAPPLIEDVALUE('3D6sXOCgj2IR9S5QCZBM4s',#2,$,$,(#345),#353);
#353= IFCCOSTVALUE($,$,IFCMONETARYMEASURE(120.0),$,$,$,'Estimated Cost',$);
#354= IFCRELASSOCIATESAPPLIEDVALUE('2fJpwNpJr4ARMRt3IkuZol',#2,$,$,(#346),#355);
#355= IFCCOSTVALUE($,$,IFCMONETARYMEASURE(280.0),$,$,$,'Estimated Cost',$);
#356= IFCRELASSOCIATESAPPLIEDVALUE('0GkKjvHIv7NOXHY_fqRI4W',#2,$,$,(#347),#357);
#357= IFCCOSTVALUE($,$,IFCMONETARYMEASURE(100.0),$,$,$,'Estimated Cost',$);

#358= IFCRELASSOCIATESAPPLIEDVALUE('39zlhnLajDKvyq1fdkhp$X',#2,$,$,(#350),#359);
#359= IFCCOSTVALUE($,$,IFCMONETARYMEASURE(230.0),$,$,$,'Estimated Cost',$);
#360= IFCRELASSOCIATESAPPLIEDVALUE('2HoT_bITD1sxj9id__lRUx',#2,$,$,(#351),#361);
#361= IFCCOSTVALUE($,$,IFCMONETARYMEASURE(150.0),$,$,$,'Estimated Cost',$);
```

Auflistung 6.6: Abbildung des Achsrasters in IFC

```
#362= IFCGRID('1tTSgyyDb2j8r9EiHHGhi$',#2,$,$,$,#363,#365,(#374,#378,#382,#386)
      ,(#390),$);
#363= IFCLOCALPLACEMENT(#20,#364);
#364= IFCAXIS2PLACEMENT3D(#370,#368,#369);
#365= IFCPRODUCTDEFINITIONSHAPE($,$,(#366));
#366= IFCSHAPEREPRESENTATION(#4,'FootPrint','GeometricCurveSet',(#367));
#367= IFCGEOMETRICCURVESET((#389,#381,#377,#373,#385));
#368= IFCDIRECTION((0.0,0.0,1.0));
#369= IFCDIRECTION((0.0,1.0,0.0));
#370= IFCCARTESIANPOINT((0.0,0.0,0.0));

#371= IFCCARTESIANPOINT((-5000.0,0.0));
#372= IFCCARTESIANPOINT((5000.0,0.0));
#373= IFCPOLYLINE((#371,#372));
#374= IFCGRIDAXIS('A1',#373,.T.);

#375= IFCCARTESIANPOINT((-5000.0,-4000.0));
#376= IFCCARTESIANPOINT((5000.0,-4000.0));
#377= IFCPOLYLINE((#375,#376));
#378= IFCGRIDAXIS('A2',#377,.T.);
...
```

6.2 Praxisbeispiele

Beispiel A:
In diesem Beispiel werden zwecks Planung des Ausbaus eines Bürogebäudes zur Unterteilung des Untergeschosses in Mietbereiche Achsen verwendet. Abbildung 6.2 zeigt Auszüge aus der 2D- und 3D-Planung sowie den berechneten Zwischenbereich von Achse B und C zur Adressierung des Mietbereichs II.2. Die gemessene Rechenzeit für die Berechnung des Zwischenbereichs beträgt eine Millisekunde[64]. Die daraus erzeugte dreidimensionale Zone ist Grundlage für die Objektauswahl. Abbildung 6.3 zeigt die Ergebnisse von drei Anfragen. Im oberen Bild wurden zunächst alle Raumobjekte selektiert, die vollständig innerhalb des Mietbereichs II.2 liegen. Zu erkennen ist, dass der Flur, der sich über Mietbereich II.1 und II.2 erstreckt, nicht selektiert wurde. Daraufhin wurde die Anfrage mit aktivierter Objektteilung wiederholt. Das Ergebnis ist im mittleren Bild dargestellt. Im unteren Bild ist das Ergebnis der Selektion aller Trockenbauwände innerhalb des Mietbereichs II.2 gezeigt. Alle tragenden, den Mietbereich begrenzenden Wände werden von der Auswahlzone geschnitten und wurden daher nicht selektiert. Die angegebenen Rechenzeiten beinhalten das Ausführen der geometrischen Algorithmen für alle Objekte im gezeigten Geschoss sowie die Selektion der positiv getesteten Objekte.

[64] Zeiten unter Windows für Intel Core 2 Duo T9400 2.53Ghz, 3GB RAM

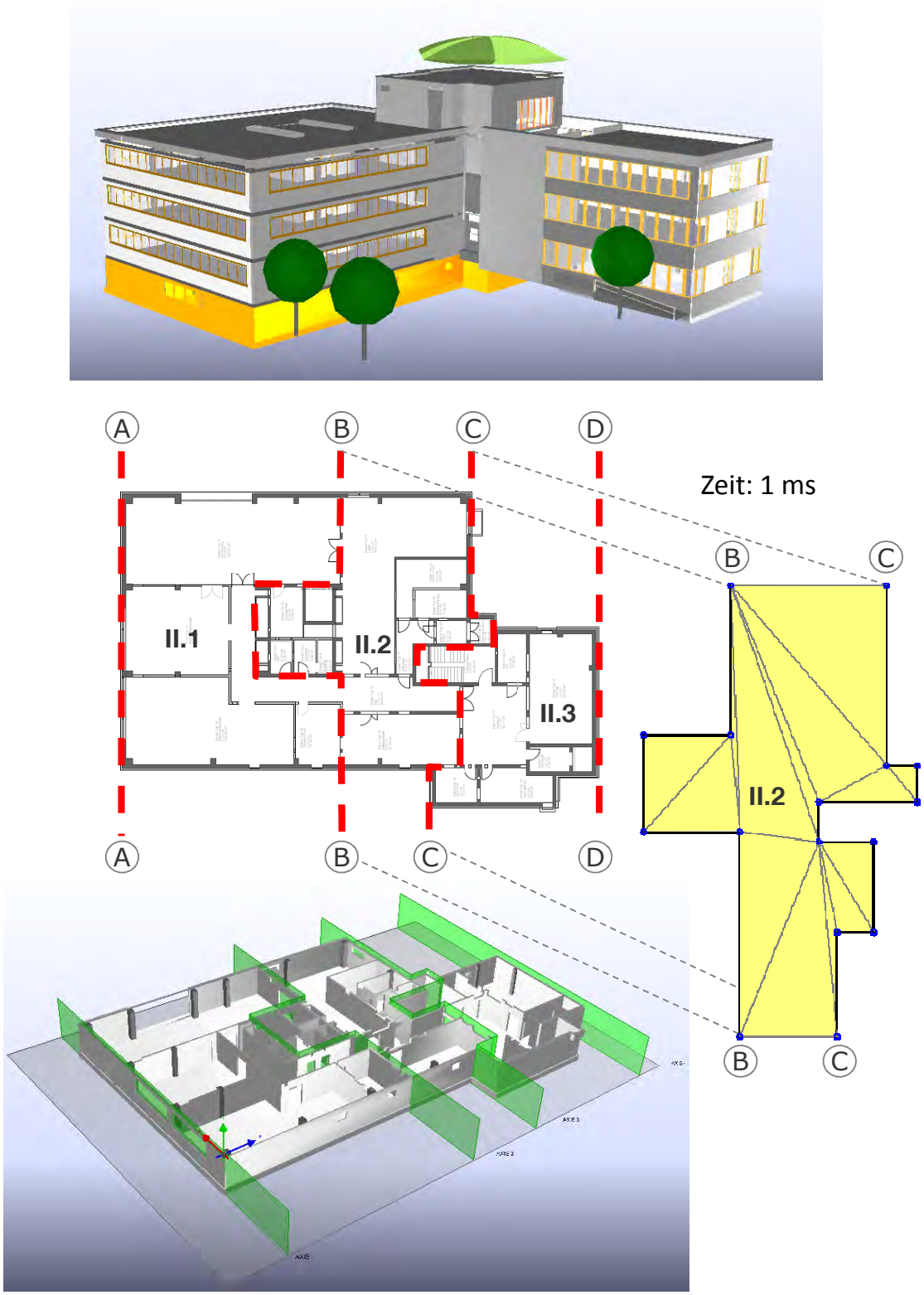


Abbildung 6.2: Beispiel A: Bestimmung des Zwischenbereichs zweier Achsen als Grundlage für die Objektselektion

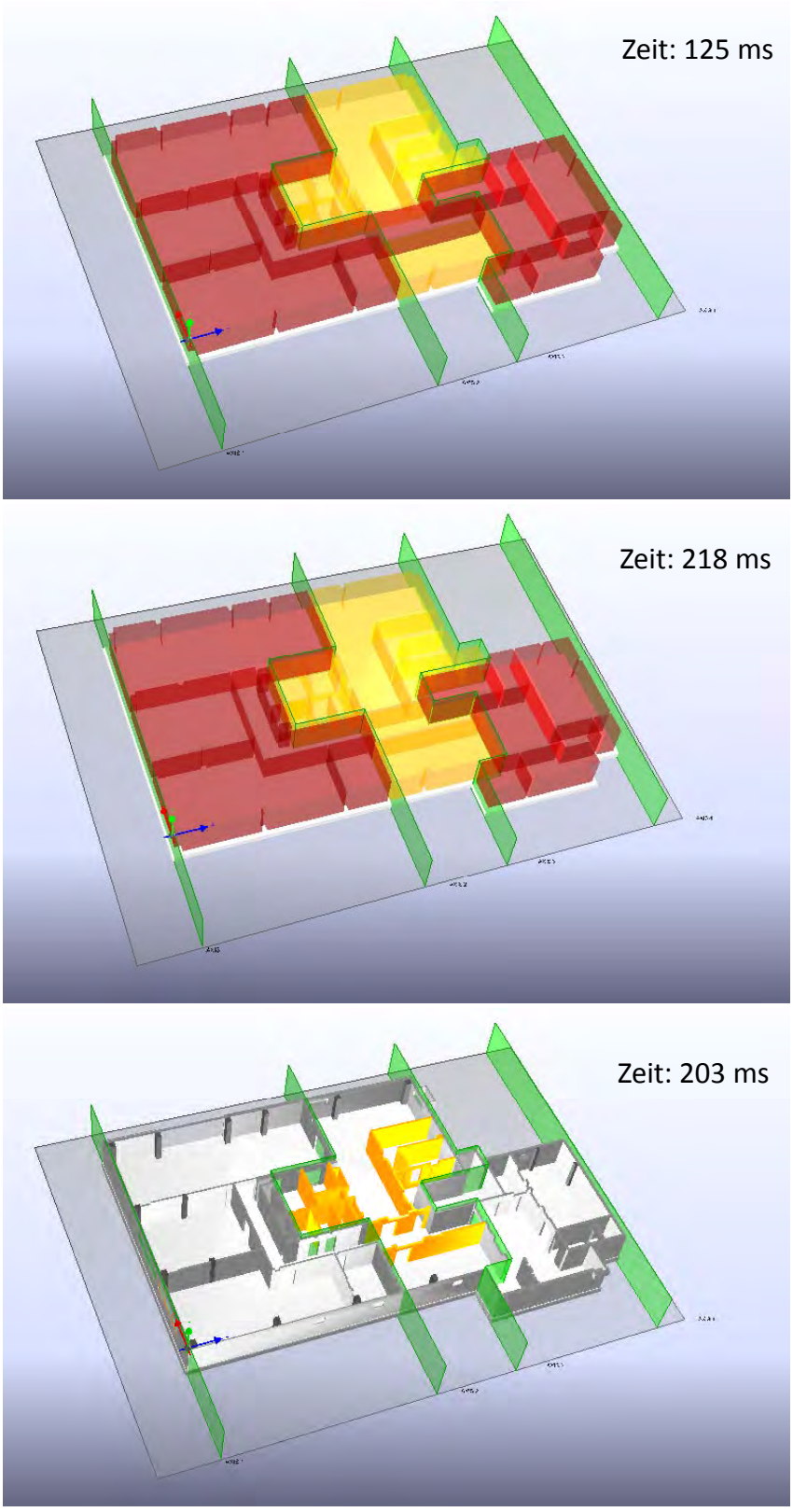


Abbildung 6.3: Beispiel A: Selektion von Objekten im Mietbereich II.2
Oben: Räume ohne Objektteilung, Mitte: Räume mit Objektteilung, Unten: Trennwände

Beispiel B: Eine Deckenplatte soll aus technischen Anforderungen und aufgrund der zur Verfügung stehenden Ressourcen in mehrere Bauabschnitte eingeteilt werden. Dabei sollen die Arbeitsfugen entlang tragender Wände und Unterzüge des darunter liegenden Geschosses geführt werden. Abbildung 6.4 zeigt die Situation. Für die Berechnung der auf die einzelnen Bauabschnitte entfallenden Betonmengen und zur Erstellung einer 4D-Simulation werden vom Terminplaner fünf Zonen definiert und mit ihnen nacheinander die Deckenplatte geteilt. Abbildung 6.5 zeigt die definierten Zonen und die geteilte Deckenplatte. Eine Verwendung von Achsen zur Teilung wäre aufgrund der T-förmigen Fuge nicht ohne Weiteres möglich gewesen.

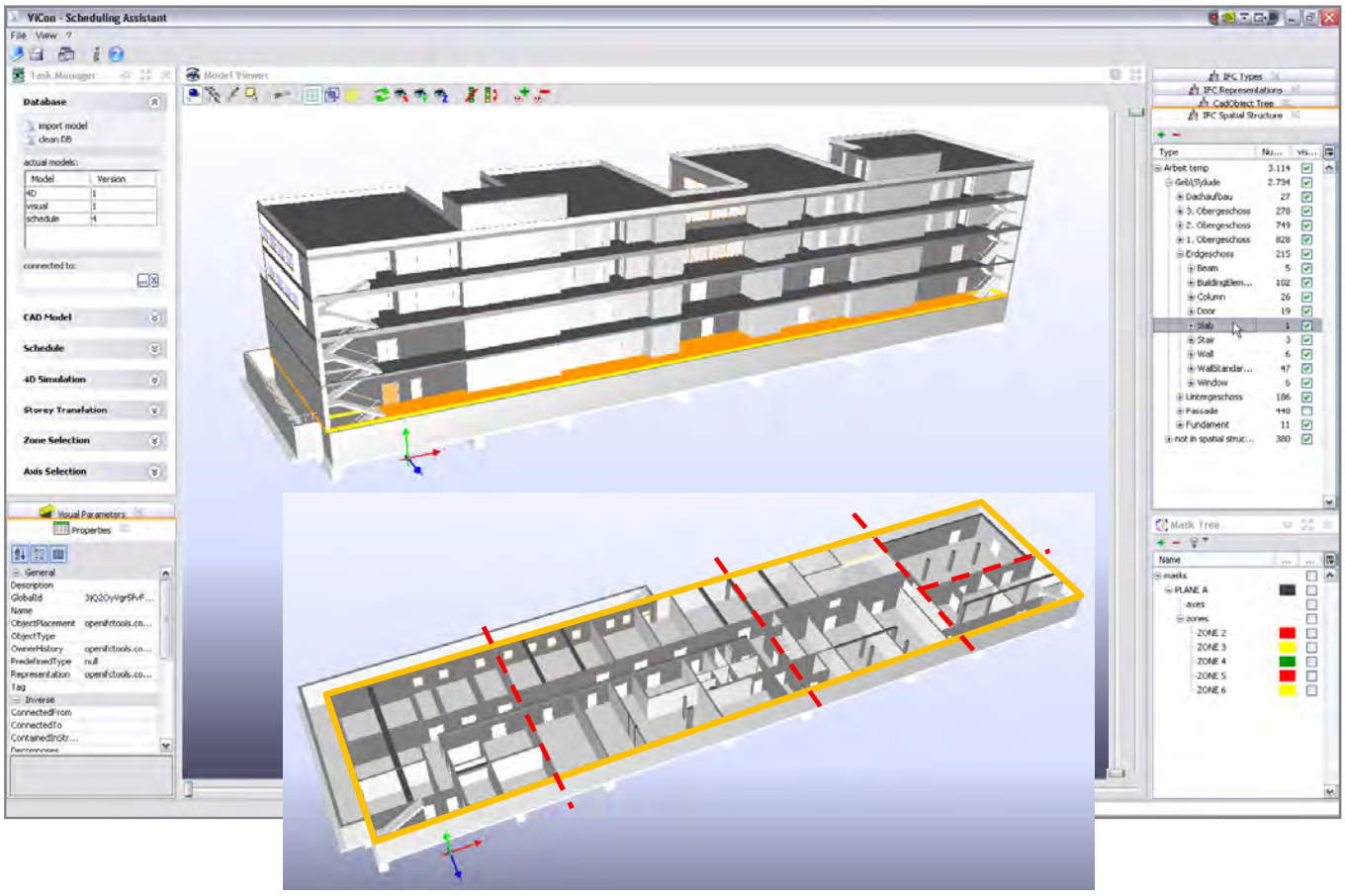


Abbildung 6.4: Beispiel B: Einteilung einer Deckenplatte in Bauabschnitte

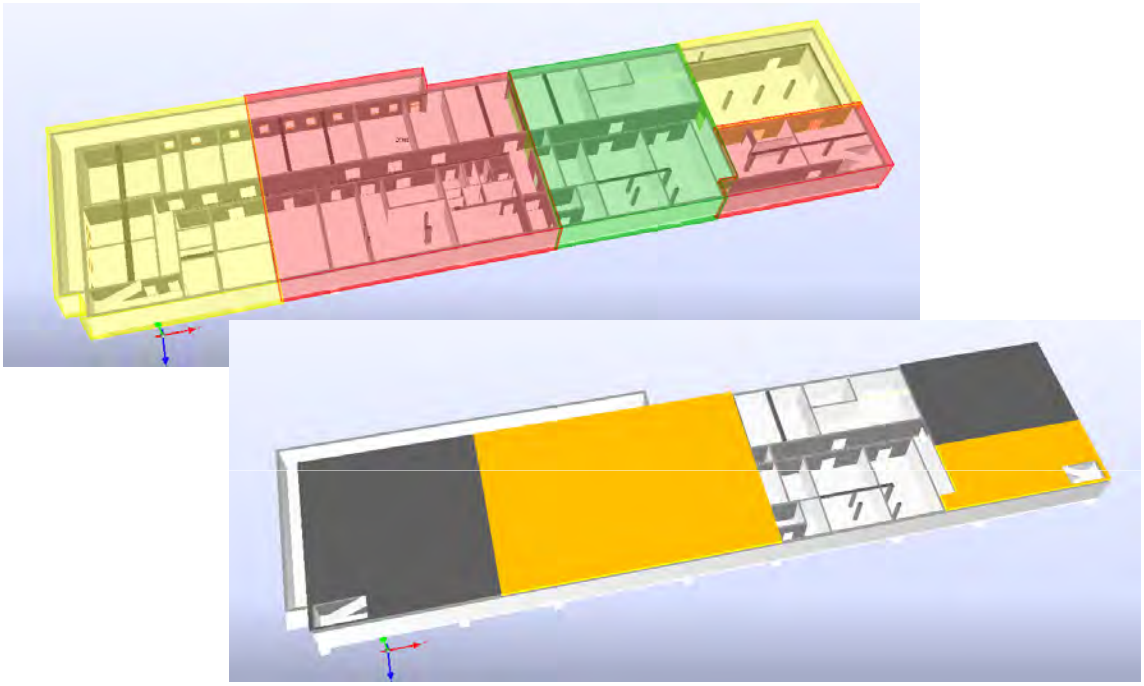


Abbildung 6.5: Beispiel B: Teilung der Deckenplatte mittels Zonen entsprechend der Bauabschnitte

Beispiel C: Bei einem Hochhausbau erfolgt die Rohbauerstellung für alle Geschosse nach einem regelmäßigem Schema. Der Grundriss des Gebäudes wird mit Hilfe von drei Achsen in einen nördlichen und einen südlichen Bauabschnitt unterteilt. Zur Adressierung der Kernbereiche werden zudem zwei Zonen verwendet. Zonen und Achsen werden für jedes Geschoss auf getrennten Ebenen definiert. Abbildung 6.6 zeigt vier Geschosse des Gebäudes, die Definition von Zonen und Achsen in der treppenförmigen Explosionsansicht sowie einen Auszug aus dem Terminplan. Die Erstellung der Deckenplatte, Kernwände und Stützen erfolgt jeweils abwechselnd für die beiden Bauabschnitte. Der Innenausbau erfolgt mit einem Nachlauf von zwei Geschossen für beide Bauabschnitte gleichzeitig (im Bild nicht zu sehen). Als Basis für die Erstellung einer 4D-Simulation werden vom Terminplaner Makros definiert. Die Makros stellen Schablonen für die Verknüpfungsregeln dar. Dabei werden die Eigenschaften Objekttyp (*type*) und Geschoss (*storey*) sowie die zuvor angelegten Zonen und Achsen zur Filterung der zu verknüpfenden Objekte verwendet. Als Parameter werden die Nummer bzw. der PSP-Code des Vorgangs (*ta*), die Geschossbezeichnung (*sy*) sowie die Nummern der jeweiligen Achsen (*a1*,*a1*) und Zone (*zn*) übergeben. Folgende Makros werden definiert:

```
#LinkDecke(ta,sy,a1,a2):=
    linkg("ta",{betweenAxisRelation(
    {CAD:type="slab",storey="sy"->ID},
    {"AXIS a1","AXIS a2"}):->ID},"Rohbau")
#LinkStützen(ta,sy,a1,a2):=
    linkg("ta",{betweenAxisRelation(
    {CAD:type="column",storey="sy"->ID},
    {"AXIS a1","AXIS a2"}):->ID},"Rohbau")
```

```
#Kern(sy,zn):=
    {inZoneRelation({CAD:type="wall",storey="sy"->ID},
    {"ZONE zn"}):->ID}
#LinkKern(ta,sy,zn):=
    linkg("ta",#Kern("sy","zn"),"Rohbau")
#LinkInnenwände(ta,sy,zn1,zn2):=
    linkg("ta",{CAD:type="wall",storey="sy",
    ID notin #Kern("sy","zn1"),
    ID notin #Kern("sy","zn2")->ID},"Ausbau")
```

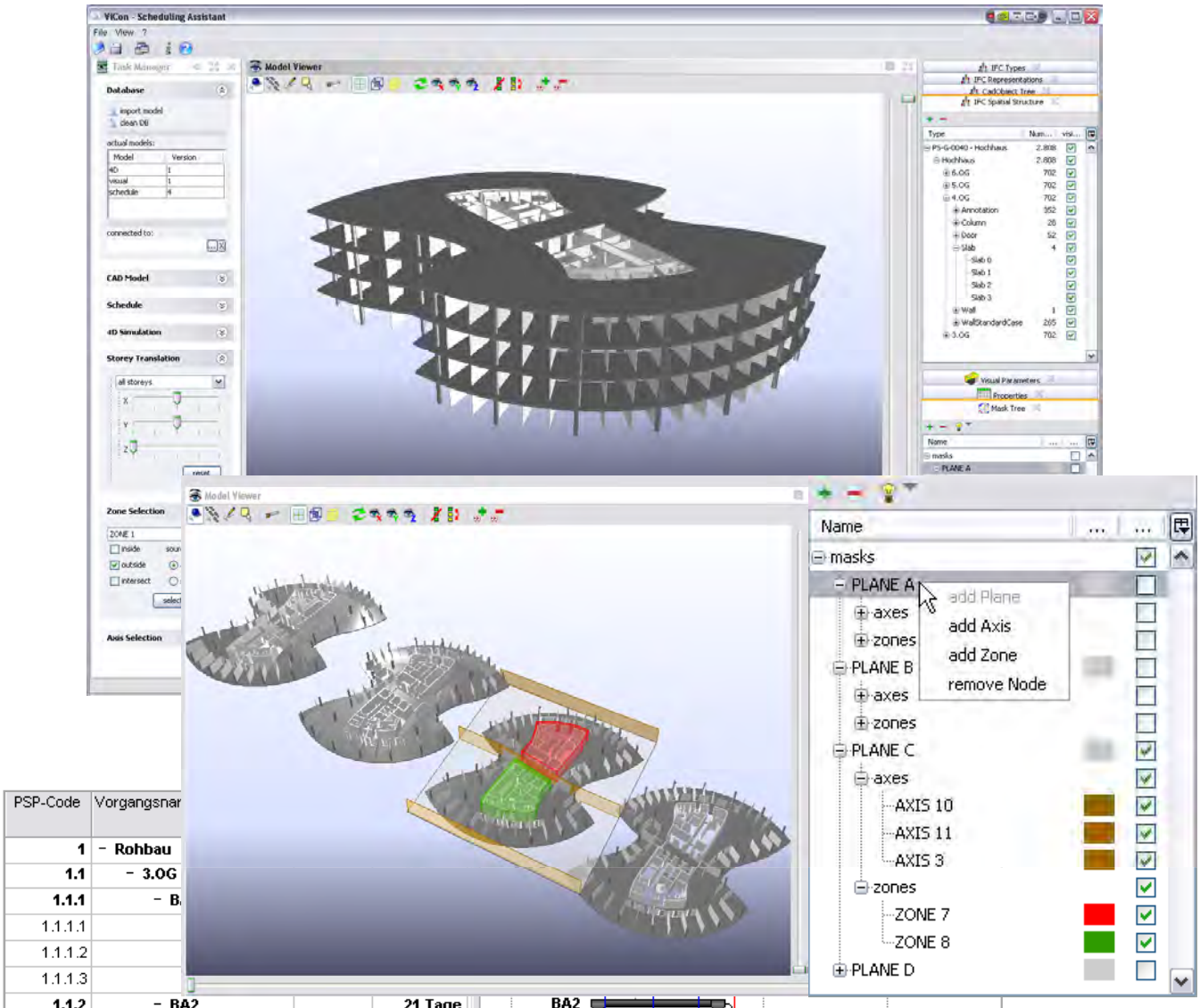
Die Makros sind projektunabhängig und können auch für andere Hochhausbauten genutzt werden. Eine Anpassung an das spezifische Projekt erfolgt lediglich durch die übergebenen Parameter und die verwendeten Zonen und Achsen.
Die eigentliche Verknüpfung der Vorgänge erfolgt beispielhaft für das 4.OG wie folgt:

```
4.OG BA1:
   #LinkDecke("1.2.1.1","4.OG",3,10)
   #LinkKern("1.2.1.2","4.OG",7)
   #LinkStützen("1.2.1.3,"4.OG",3,10)

4.OG BA2:
   #LinkDecke("1.2.2.1","4.OG",3,11)
   #LinkKern("1.2.2.2","4.OG",8)
   #LinkStützen("1.2.2.3"','4.OG",3,11)

   #LinkInnenwände("2.2.3","4.OG",7,8)
```

Das Resultat der Verknüpfungen ist in Abbildung 6.7 dargestellt. Die Verknüpfung für die anderen Geschosse erfolgt analog. Die deutliche Zeitersparnis eines solchen regelbasierten Verknüpfens bei Hochhausbauten wurde bereits in [Tulke u. Hanff 2007] beschrieben. Dabei standen jedoch noch keine räumlichen Anfragen bzw. eine Objektteilung zur Verfügung.
Das Kopieren und Anpassen der Verknüpfungsregeln (Makroaufrufe) für die einzelnen Geschosse könnte im Zuge einer Weiterentwicklung zudem automatisiert werden. Die Werte der Parameter *ta*, *sy*, *zn*, *a1*, *a2* könnten direkt aus den innerhalb der Terminplanstruktur vorhandenen Informationen abgeleitet werden, da ein direkter Zusammenhang aus Geschoss, Bauabschnitt und verwendeten Zonen und Achsen besteht. Der Name des zu wählenden Makros könnte bei einer formalisierten Namensgebung zudem direkt aus dem Vorgangsnamen abgeleitet werden. Um die Arbeitsweise des Terminplaners jedoch nicht durch zu hohe Anforderungen an die Terminplanstruktur und Namensgebung zu beeinflussen, könnte eine Ableitung der Verknüpfungsregeln alternativ auf Basis einer Attributierung der Terminplanvorgänge beruhen. Attributierungen des Terminplans werden oftmals ohnehin schon zwecks Filterung vorgenommen.
Um abschließend einen Eindruck von den Antwortzeiten der verwendeten räumlichen Anfragen bzw. geometrischen Algorithmen zu vermitteln, sind in Abbildung 6.8 und 6.9 die Ergebnisse bei der Auswertung aller Objekte der vier gezeigten Geschosse angegeben.



PSP-Code	Vorgangsnar		
1	**- Rohbau**		
1.1	**- 3.OG**		
1.1.1	**- B**		
1.1.1.1			
1.1.1.2			
1.1.1.3			
1.1.2	**- BA2**		**21 Tage**
1.1.2.1	Decke	4	6 Tage
1.1.2.2	Kern	5;8	9 Tage
1.1.2.3	Stützen	6;9	3 Tage
1.2	**- 4.OG**	**7**	**27 Tage**
1.2.1	**- BA1**		**18 Tage**
1.2.1.1	Decke		6 Tage
1.2.1.2	Kern	13	9 Tage
1.2.1.3	Stützen	14	3 Tage
1.2.2	**- BA2**		**21 Tage**
1.2.2.1	Decke	13	6 Tage
1.2.2.2	Kern	14;17	9 Tage
1.2.2.3	Stützen	15;18	3 Tage

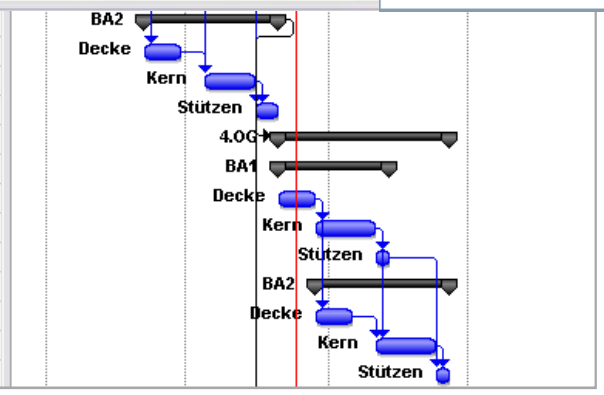


Abbildung 6.6: Beispiel C: Modell, Terminplan und definierte Zonen für ein Hochhausprojekt

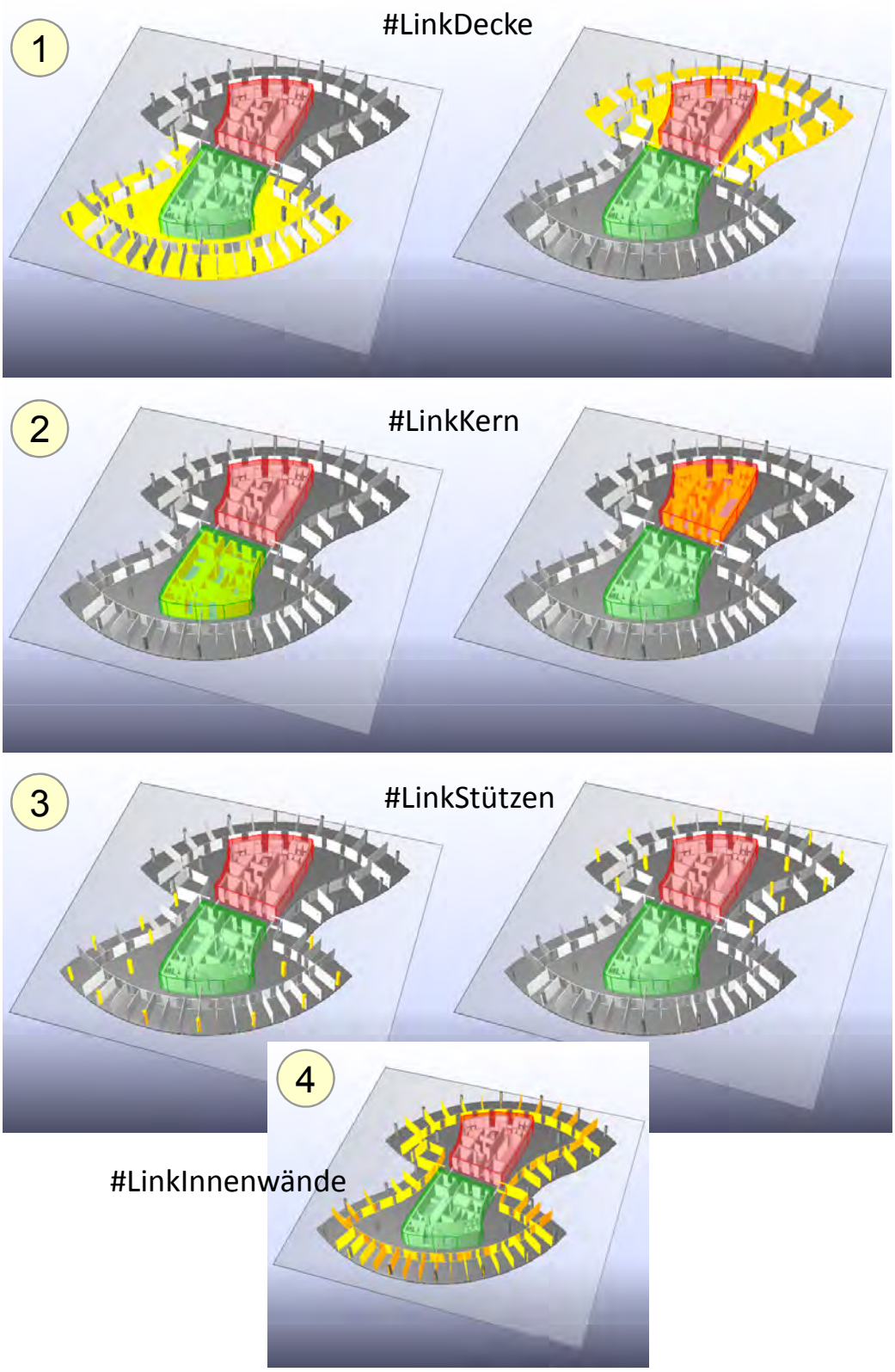
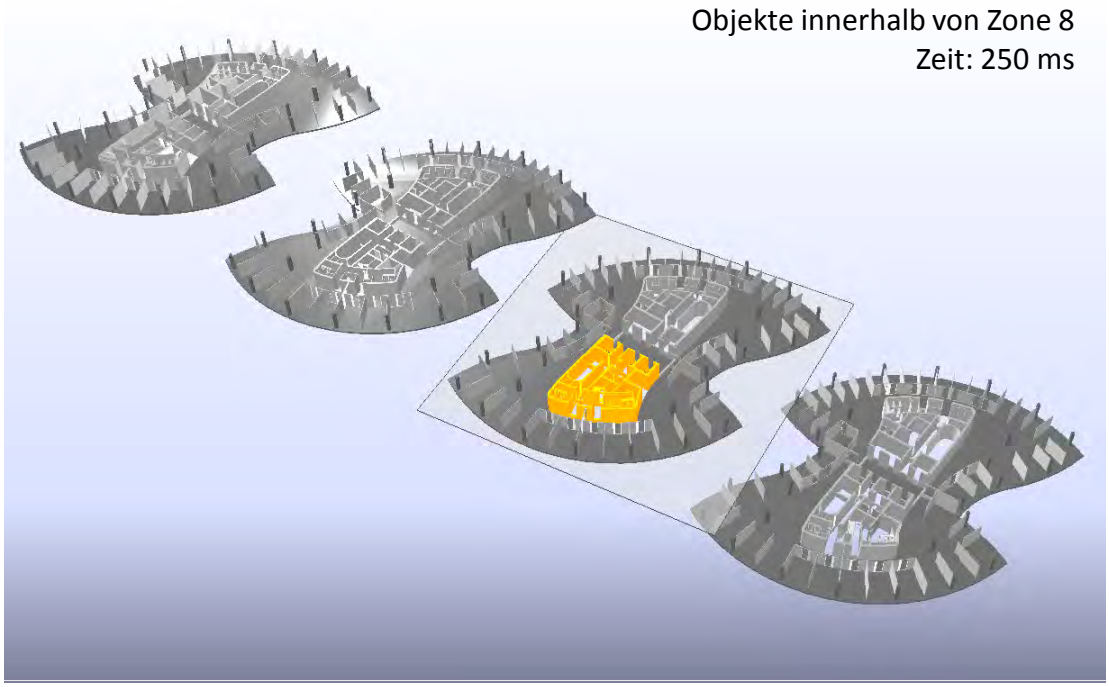


Abbildung 6.7: Beispiel C: Ergebnis der regelbasierten Verknüpfung jeweils je Bauabschnitt getrennt



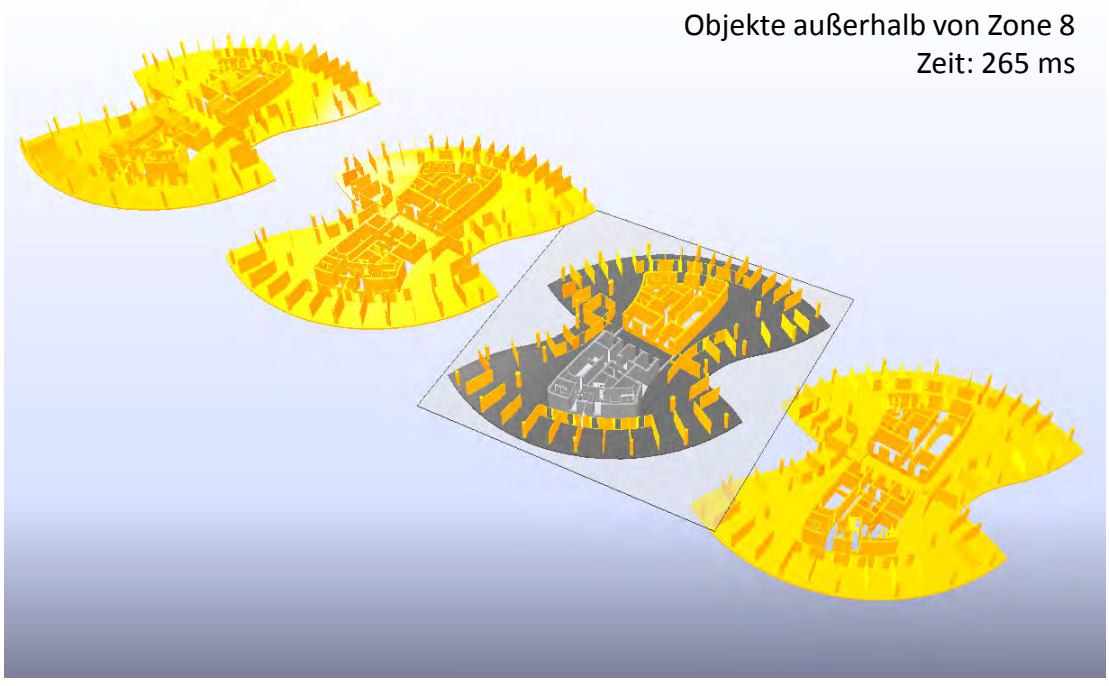
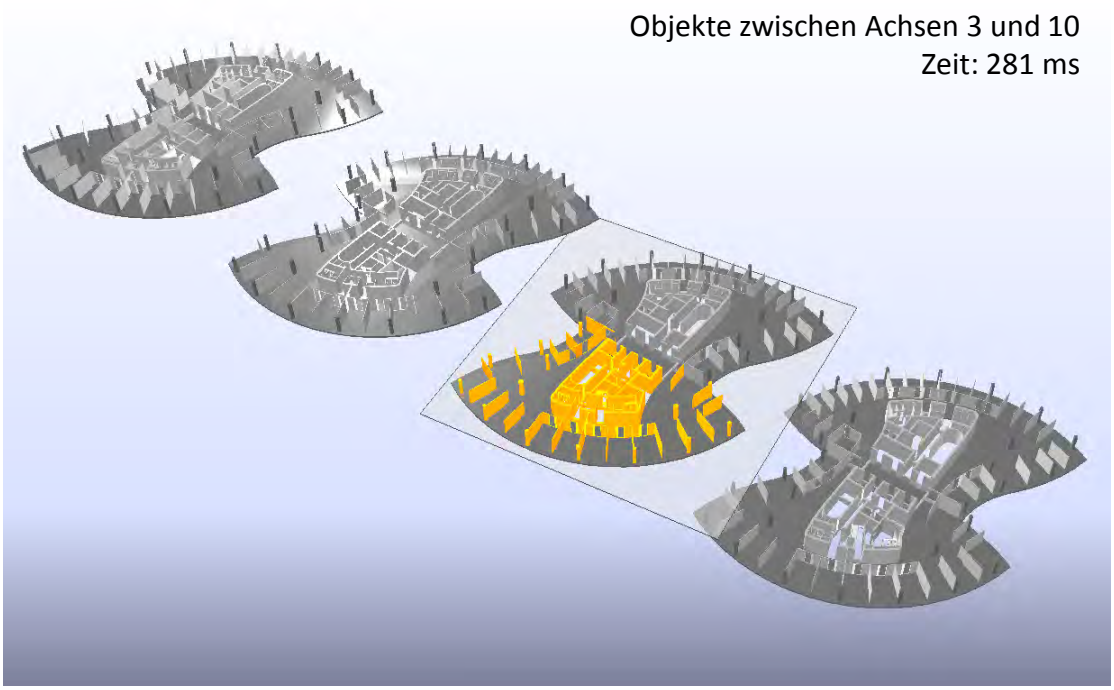


Abbildung 6.8: Antwortzeiten bei der Selektion mittels einer Zone bei insgesamt 2808 geprüften Objekten (Zeiten unter Windows für Intel Core 2 Duo T9400 2.53Ghz, 3GB RAM)



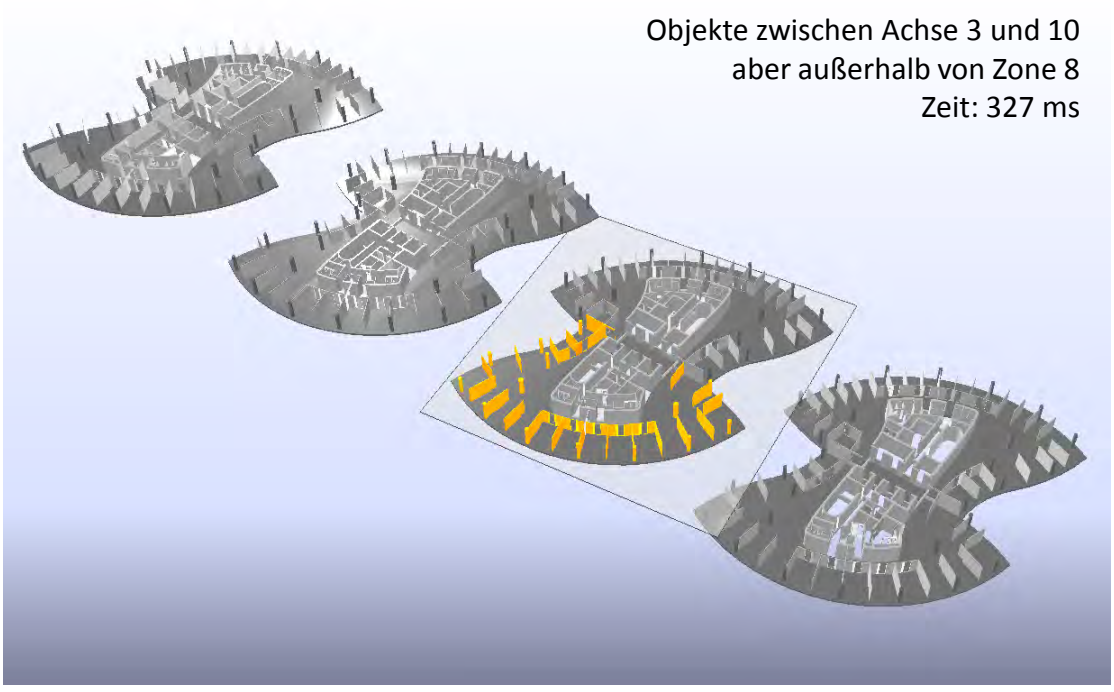


Abbildung 6.9: Antwortzeiten bei der Selektion mittels Achsen und Zonen bei insgesamt 2808 geprüften Objekten (Zeiten unter Windows für Intel Core 2 Duo T9400 2.53Ghz, 3GB RAM)

6.3 Erweitertes Demonstrationsszenario

Um das Gesamtkonzept auch im Zusammenspiel mit verschiedenen Softwarepaketen weiter zu evaluieren, ist im Rahmen des InPro-Projektes für Mai 2010 ein Demonstrationsszenario gemäß Abbildung 6.10 geplant. Ein CAD-Modell inklusive des vom Architekten vorgegebenen Achsrasters und zugehörigen Basis- bzw. LV-Mengen eines Beispielprojektes werden als Ausgangsdaten auf einem IFC-basierten Modellserver hinterlegt. Ebenso eine Aufstellung des Kostenbudgets. Der zu demonstrierende Prozess entspricht den in Abbildung 4.4 auf Seite 48 aufgeführten Schritten 2-6 und wird von zwei Akteuren ausgeführt. Akteur A repräsentiert den Terminplaner, der weitgehende Änderungen an den Terminplanungs- und 4D-Simulationsdaten vornimmt. Akteur B repräsentiert hingegen das Projektmanagement oder einen Nachunternehmer, der zwar als Vorschlag Datumsänderungen bzw. das Einfügen von Untervorgängen, nicht jedoch das Ändern der Verknüpfungen der 4D-Simulation bzw. der Objektgranularität vornehmen kann.
Von Akteur A wird der im Rahmen dieser Arbeit entwickelte Prototyp verwendet. Von Akteur B hingegen MS Project mit einem von TNO[65] auf Basis der IFCEngineDLL[66] entwickelten Plug-in für den IFC Im- und Export sowie einem externen Anwendungsfenster mit 4D-Simulations-Funktionalität.
Im Rahmen der Vorbereitung dieses Demonstrationsszenarios ist eine kontinuierliche Verbesserung sowie Dokumentation der entwickelten Softwarekomponenten geplant.

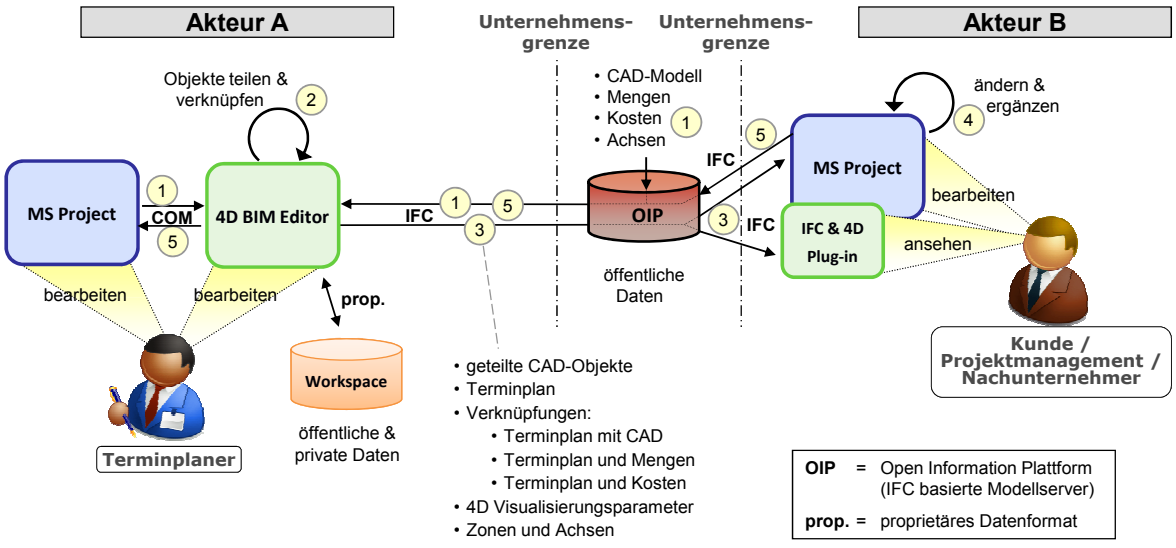


Abbildung 6.10: Demoszenaro

[65] www.tno.nl
[66] www.ifcbrowser.com

7 Abschließende Bemerkungen

7.1 Zusammenfassung

Ziel dieser Arbeit war die Erstellung und Erprobung eines Konzeptes für die nahtlose Integration der Terminplanung in die sich derzeit bei der Planung von Bauprojekten etablierende, modellbasierte Arbeitsweise. Dabei sollte insbesondere aus Sicht des Terminplaners gegenüber seiner derzeitigen, weitestgehend autonomen Arbeitsweise ein Mehrwert entstehen, der eine Umstellung rechtfertigt. Mit der innerhalb der Terminplanung derzeit noch üblichen Beschränkung einer Modellnutzung auf das Erstellen von 4D-Simulationen konnte zwar in der Vergangenheit bereits hinsichtlich der Fehlerkontrolle und des tieferen Verständnisses des geplanten Bauablaufs ein gewisser Mehrwert geschaffen werden, jedoch erzeugt die Erstellung und mehrfache Aktualisierung von 4D-Simulationen im Verlauf einer iterativen Planung noch einen unverhältnismäßig hohen Zusatzaufwand. Zudem entsteht für den Terminplaner während der Terminplanerstellung (seiner eigentlichen Kernaufgabe) durch eine nachträgliche Nutzung einer 4D-Simulation kein unmittelbarer Nutzen. Somit blieb die Anwendung dieser Technik in der Praxis bisher die Ausnahme.
Im Rahmen dieser Arbeit wurde daher nach einer detaillierten Analyse der derzeitigen Arbeitsweise von Terminplanern sowie existierenden Randbedingungen ein Konzept entwickelt, das eine Verbesserung des Aufwand-Nutzen-Verhältnisses der modellbasierten Arbeitsweise und damit eine praxisgerechtere Integration der Terminplanung erreicht.
Ein *zusätzlicher Nutzen* wird dadurch generiert, dass, neben der bisher üblichen Verwendung der Geometrie zur Erstellung von 4D-Simulationen, auch weitere Informationen aus einem Bauwerksinformationsmodell mit Terminplanvorgängen verknüpft und als Grundlage für die Berechnung der Dauer von Vorgängen bereits während der Terminplanerstellung genutzt werden. Für die Berechnung der Vorgangsdauer wurden dabei in Abhängigkeit vom Reifegrad der Planung vier Methoden betrachtet. Je nach verwendeter Methode erfolgt die Berechnung auf Grundlage von Basismengen (groben Raum- und Flächendaten), Mengen des Leistungsverzeichnisses (detaillierten Bauteilmengen) oder dem Kostenbudget eines Gewerks sowie mit Hilfe von Aufwandswerten. Um die bisher übliche Neuermittlung dieser Eingangsinformationen während der Terminplanung zu vermeiden und stattdessen deren Wiederverwendung aus einem gemeinschaftlich im Projekt erstellten Bauwerksinformationsmodell zu ermöglichen, wurde ein Systementwurf unter Verwendung eines zentralen Modellservers und eines lokalen Workspace vorgestellt. Dieser Systementwurf erlaubt gleichzeitig den Austausch und die Verteilung von Terminplan- und 4D-Simulationsdaten unter den Projektbeteiligten und ermöglicht die Trennung der Datenhaltung von öffentlichen und privaten Daten. Um den Systementwurf trotz der im Bauwesen von Projekt zu Projekt wechselnden Zusammensetzung von Planungspartnern flexibel realisieren zu können, wurde für die Kommunikation mit

dem Modellserver das produktneutrale Datenformat IFC gewählt und für alle auszutauschenden Daten eine Abbildung in dieses Datenmodell vorgenommen.
Zur *Verringerung des Aufwands* für die Gewinnung der zuvor genannten Informationen bzw. die Erstellung und Aktualisierung der entsprechenden Verknüpfungen wurde als Kern dieser Arbeit eine Verknüpfungssprache spezifiziert. Diese ermöglicht die präzise Auswahl von Teilinformationen eines Bauwerksinformationsmodells mittels einer Kombination aus attributbasierten und räumlichen Filterkriterien auf mehreren miteinander in Beziehung stehenden Teilmodellen. Somit wird beispielsweise auch eine räumliche Auswahl von Mengen des Leistungsverzeichnisses ermöglicht. Für die räumliche Filterung werden dabei in der Ebene frei definierte und anschließend extrudierte Zonen und Achsen verwendet, die gleichzeitig der Einteilung des Bauwerksmodells in Bauabschnitte dienen und, wenn erforderlich, als Grundlage für eine automatische Objektteilung verwendet werden. Die zugehörigen geometrischen Algorithmen werden in der Arbeit ausführlich dargestellt. Zusätzlich werden von der Verknüpfungssprache Funktionen zur Weiterverarbeitung der zuvor ausgewählten Teilinformationen im Sinne einer Verknüpfungserstellung oder weiteren Berechnung bereitgestellt.
Mit Hilfe der Sprache können für Terminplanvorgänge auf einheitliche Art Regeln zur Verknüpfung mit Mengen-, Kosten- und Geometriedaten des Bauwerksinformationsmodells formuliert werden. Die regelbasierte Verknüpfung hat dabei eine wesentliche Verringerung des Verknüpfungsaufwands zur Folge. Im Rahmen der Reaktion auf Änderung in den zugrunde gelegten Planungsdaten reicht zudem oftmals eine Reinterpretation dieser Regeln aus, um den Terminplan und die zugehörige 4D-Simulation zu aktualisieren. Nur bei neu hinzugefügten Objekten, die bisher von keiner Regel erfasst werden, oder bei Änderung der Planungsstrategie des Terminplaners wird eine Anpassung von Verknüpfungsregeln erforderlich.
Um eine weitere Effizienzsteigerung bei der Erstellung von Verknüpfungen zu erreichen, wurde in den Interpreter der Verknüpfungssprache eine Funktionalität zur Definition und Verwendung von parametrisierten Abkürzungen (Macros) integriert. Hierdurch ist es möglich, Sprachausdrücke (Regeln) als Schablonen zu speichern und in leicht abgewandelter Form mehrfach zu verwenden (z.B. für verschiedene Geschosse).
Um Änderungen in Planungsdaten für den Terminplaner nachvollziehbar zu gestalten, wurde zudem ein Konzept für die Versionierung der lokal im Workspace gespeicherten Daten entwickelt. Damit wird eine gezielte Analyse von Planungsänderungen möglich, wodurch der Terminplaner zusätzlich bei der Aktualisierung des Terminplans unterstützt wird. Zudem erfolgt durch die Versionierung eine Dokumentation des Planungsverlaufs.
Insgesamt wurde der Aufwand für das Erstellen und Aktualisieren der Verknüpfungen somit durch die Kombination aus einer regelbasierten Verknüpfung und der Versionierung der Planungsdaten auf ein Minimum reduziert. In Verbindung mit dem erhöhten Mehrwert aus der Wiederverwendung von Modelldaten für die Berechnung der Dauer von Terminplanvorgängen und der zugehörigen Infrastruktur aus dem Systementwurf wurde somit eine tiefere Integration der Terminplanung in die modellbasierte Arbeitsweise erreicht und damit die Zusammenarbeit im Projekt insgesamt verbessert.

Schwierigkeiten ergaben sich lediglich im Zusammenhang mit räumlichen Anfragen bei der Definition und Berechnung des Zwischenbereichs zweier Achsen. Hierfür konnte keine mathematische Definition gefunden werden, die für alle erdenklichen Konfigura-

tionen unmittelbar dem intuitiven Verständnis entspricht. Es wurde daher eine Variante gewählt, die für in der Bauplanung gängige Konfigurationen sinnvolle Ergebnisse liefert und eine Berechnung des Zwischenbereichs auf relativ einfache Weise ermöglicht.
Bei der Modellbildung und Umsetzung der Verknüpfungssprache war zudem eine Datenstruktur zu wählen, die einerseits ein einheitliches Konzept für die Filterung von Daten unterschiedlicher Teilmodelle auf Attributbasis sowie die Integration räumlicher Anfragen erlaubt und andererseits als Grundlage für eine einheitliche Versionierung auf Objektebene dient. Dieses konnte mit dem in der Arbeit vorgestellten Datenbankschema für eine relationale Datenbank und die Zurückführung von Ausdrücken der Verknüpfungssprache auf SQL-Anfragen erreicht werden.
Es konnte zudem festgestellt werden, dass die Abbildung der im Rahmen der Terminplanung verarbeiteten Daten in IFC2x3 bereits weitestgehend möglich ist. Frei platzierbare Zonen und Kalenderinformationen des Terminplans sind jedoch erst in der kommenden Version IFC2x4 abbildbar.

7.2 Bewertung der Ergebnisse

Das vorgestellte Konzept zielt auf die Verbesserung des Nutzen-Aufwand-Verhältnisses für die Einbeziehung von Bauwersksinformationsmodellen in die Terminplanung ab. Dabei werden jedoch bestehende Arbeitsweisen und gängige Softwarekomponenten nicht vollständig ersetzt, sondern mit zusätzlichen Komponenten und Funktionalitäten auf eine Weise ergänzt, die existierende Randbedingungen berücksichtigt. Hierdurch ist eine schnelle Akzeptanz in der Praxis zu erwarten.
Die spezifizierte Verknüpfungssprache stellt zudem ein mächtiges und gleichzeitig flexibles Werkzeug für die regelbasierte Erstellung von Verknüpfungen zwischen Informationen eines Bauwerksinformationsmodells dar. Sie kann dabei auch für andere Anwendungsfälle außerhalb der Terminplanung genutzt und dafür entsprechend erweitert werden. Die Auswertungsgeschwindigkeit der vorgestellten geometrischen Algorithmen ist für übliche Modellgrößen komfortabel. Es bieten sich jedoch weitere Optimierungsmöglichkeiten durch die Verwendung raumpartitionierender Datenstrukturen. Durch die Einbeziehung der Objektteilung in die Sprachfunktionalität wird zudem ein ansonsten daraus resultierendes Hindernis im Prozessablauf behoben und der Terminplaner in die Lage versetzt, das Architekturmodell auf einfache Weise in ein Modell für die Bauausführung zu überführen.
Die direkte Formulierung von Regeln in der formalen Syntax der Verknüpfungssprache ist für den Endanwender trotz der in der Testanwendung zur Verfügung gestellten Drag&Drop-Vorlagen sehr abstrakt und erfordert daher eine gewisse Einarbeitung. Eine marktfähige Anwendung könnte diese Hürde jedoch durch eine verfeinerte grafische Benutzerschnittstelle senken, die die Bildung von Sprachausdrücken ohne Einschränkung der Funktionalität vor dem Anwender weitestgehend verbirgt.
Die Verwendung eines Modellservers als zentrale Plattform für den Datenaustausch in einem Projekt bietet im Rahmen der Terminplanung den Zugriff auf wesentliche Eingangsinformationen für die Terminplanerstellung und dient damit der Wiederverwendung von Daten. Gleichzeitig ermöglicht er die Zusammenarbeit an der Terminplanerstellung beteiligter Projektpartner und vereinfacht die Kommunikation im Rahmen des zeitlichen Projektmanagements.

Sowohl der konzeptionelle Entwurf als auch die prototypische Umsetzung und Erprobung einer in die modellbasierte Arbeitsweise integrierten Terminplanung wurden im Rahmen dieser Arbeit umfänglich behandelt und können im Ergebnis als abgeschlossen betrachtet werden. Für die Einführung in der Praxis ist jedoch noch eine kommerzielle Umsetzung durch einen im Bauwesen bereits etablierten Softwareanbieter erforderlich.

7.3 Ausblick

Die Einzelkomponenten des entwickelten Konzeptes wurden in Java umgesetzt. Sie werden kurzfristig inklusive Dokumentation als open source unter der Internetdomain www.openifctools.de veröffentlicht und damit als Basis für weitere Forschung im Bereich der Verwendung von Bauwerksinformationsmodellen und IFC bereitgestellt. Mögliche Erweiterungen im Bereich der Terminplanung und damit Ausgangspunkte für eine weitere Entwicklung könnten sein:

- die Unterstützung von Achsanfragen der Form „in Achse“,
- die Implementierung eines Punktfangs zur Anwenderunterstützung bei der Eingabe von Zonen und Achsen,
- die Implementierung der Neuermittlung von Mengen für geteilte CAD-Objekte,
- die Übertragung der vorgestellten geometrischen Algorithmen auf eine Geometriebeschreibung mit NURBS (Non-Uniform Rational B-Splines),
- die Festlegung eines verbindlichen Farbschemas für die Visualisierung von Bauprozessen,
- die weitere Anwenderunterstützung durch das automatische Generieren von Regeln als Vorschlag. Dieses könnte auf Basis der Terminplanhierarchie und einer Attributierung/Klassifizierung von Vorgängen erfolgen, die üblicherweise bereits zur Filterung von Terminplanvorgängen verwendet wird.

Anpassungen der Softwarekomponenten und der Abbildung in IFC sind auch im Rahmen der in Kapitel 6.3 beschriebenen und für den Frühsommer 2010 geplanten Demonstration der Ergebnisse des InPro[67]-Projektes zu erwarten. Hierfür ist auch die Anbindung an den IFC-Modellserver der Firma Eurostep[68] (Produktname Share-A-Space) vorgesehen. Als zweite Endanwendersoftware ist das im Rahmen von InPro von TNO entwickelte IFC-Plug-in für MS Project vorgesehen, das auch eine Visualisierung von 4D-Simulationen bieten wird.
Die Firma Synchro hat im Zusammenhang mit dieser Arbeit und InPro bereits eine IFC-Schnittstelle für ihr Produkt Project Constructor implementiert[69], die das Lesen und Schreiben von CAD-, Terminplan- und 4D-Informationen ermöglicht. Auch eine (bedingt nützliche) Funktionalität zur Teilung von Objekten ist bereits verfügbar. Weitere Anpassungen und Erweiterungen sind im weiteren Projektverlauf auch hier zu

[67] www.inpro-project.eu
[68] www.eurostep.com
[69] siehe Newsletter unter www.synchroltd.com/news/NA-20090612175657

erwarten.
Die Firma ViCo Software[70] hat zudem kürzlich inoffiziell angekündigt, im Frühjahr 2010 eine neue Version ihres Produktes ViCo Office bzw. ViCo 5D Presenter herauszubringen. Diese soll eine Funktionalität enthalten, die dem in dieser Arbeit vorgestellten Konzept der Einteilung des Modells in Bauabschnitte mittels Zonen entspricht. Die Eingabe polygonal umrandeter Zonen soll, wie ebenfalls in dieser Arbeit vorgestellt, in Ebenen erfolgen. Ein Punktfang zur Unterstützung des Anwenders bei der Eingabe ist ebenfalls geplant. Die Objektteilung mittels Zonen soll auch die Neuberechnung zugehöriger Mengeninformationen beinhalten. Des Weiteren sind Mengen die Grundlage für die Bestimmung der Dauer von Vorgängen im Terminplanungsprogramm ViCo Control. Seit dem Produktstart von ViCo Office im April 2009 wird zudem nach Angaben von ViCo Software eine Objektversionierung für die in der Objektdatenbank der Firma Versant[71] gespeicherten Projektdaten geboten. Die Implementierung der hier vorgestellten, regelbasierten Verknüpfung sowie der Adressierung von Objekten auf Basis eines Achsrasters sind jedoch nicht geplant. Auch die Verwendung eines Modellservers steht nicht im Focus der Entwicklung, gleichwohl eine IFC-Schnittstelle zumindest für die CAD-Daten bereits existiert. Eine Anbindung einer Aufwandswertesammlung wird in der neuen Version lediglich indirekt durch die Übernahmen von Daten aus einem Musterprojekt unterstützt werden.

Es bleibt zu wünschen, dass die im Rahmen dieser Arbeit entwickelten Konzepte bald von weiteren Firmen aufgegriffen werden und somit in dieser oder leicht abgewandelter Form in die tägliche Planungspraxis im Bauwesen aufgenommen werden.

[70] www.vicosoftware.com
[71] www.versant.com

Anhang

A Grammatik der Verknüpfungssprache in erweiterter Backus-Naur-Form

Auflistung 1: Grammatik der Verknüpfungssprache: Terminale

```
<DEFAULT> SKIP : {
  " "
| "\r"
| "\t"
}

<DEFAULT> TOKEN : {
  <EOL: "\n">
}

<DEFAULT> TOKEN : {
  <LPARENT: "(">
| <RPARENT: ")">
| <LPARENT_SET: "{">
| <RPARENT_SET: "}">
| <LPARENT_INTERVAL: "[">
| <RPARENT_INTERVAL: "]">
}

<DEFAULT> TOKEN : {
  <EQUALS_FILTER: "=">
| <NOT_EQUALS_FILTER: "!=">
| <GREATER_FILTER: ">">
| <GREATER_EQUALS_FILTER: ">=">
| <LESS_FILTER: "<">
| <LESS_EQUALS_FILTER: "<=">
| <DEFINED_FILTER: "defined">
| <NOT_DEFINED_FILTER: "undefined">
| <CONTAINED_IN_FILTER: "in">
| <NOT_CONTAINED_IN_FILTER: "notin">
| <AND_SIGN: ",">
| <OR_SIGN: "|">
| <EXTRACTION: "->">
}

<DEFAULT> TOKEN : {
  <ALL_ATTRIBUTES: "allAttributes">
| <ALL_VALUES: "allValues">
}

<DEFAULT> TOKEN : {
  <BETWEEN_AXIS: "betweenAxisRelation">
| <IN_ZONES: "inZoneRelation">
```

```
44  }

46  <DEFAULT> TOKEN : {
      <SUM: "sum">
48  | <MAX: "max">
    | <MIN: "min">
50  | <COUNT: "count">
    | <SELECT: "select">
52  | <LINK_GEOMETRIE: "linkg">
    | <LINK_QUANTITIES: "linkq">
54  }

56  <DEFAULT> TOKEN : {
      <INTEGER_LITERAL: (["+","-"])? ["0"-"9"] (["0"-"9"])*>
58  | <FLOATING_POINT_LITERAL:
        (["+","-"])? ((["0"-"9"])+ "." (["0"-"9"])* (<EXPONENT>)?
60    | "." (["0"-"9"])+ (<EXPONENT>)? | (["0"-"9"])+ <EXPONENT>)
      >
62  | <#EXPONENT: ["e","E"] (["+","-"])? (["0"-"9"])+>
    | <STRING_LITERAL:
64      "\"" (~["\"","\\","\n","\r"]
      | "\\" ["n","t","b","r","f","\\","\'","\""])+ "\""
66    >
    | <DATE_LITERAL:
68      <DATE_ONLY_LITERAL> (" " <TIME_ONLY_LITERAL>)?>
    | <#DATE_ONLY_LITERAL:
70      ("0" ["1"-"9"]
      | ["1"-"2"] ["0"-"9"]
72    | "3" ["0"-"1"]) "." ("0" ["1"-"9"]
      | "1" ["0"-"2"]) "." ["0"-"9"]["0"-"9"]["0"-"9"]["0"-"9"]
74          >
    | <#TIME_ONLY_LITERAL:
76      ("0" ["0"-"9"]
      | "1" ["0"-"9"]
78    | "2" ["0"-"3"]) ":" ("0" ["0"-"9"]
      | ["1"-"5"] ["0"-"9"])
80    >
    }
82

    <DEFAULT> TOKEN : {
84    <IDENTIFIER: <LETTER> (<LETTER> | ["0"-"9"])*>
    | <#LETTER:
86     ["a"-"z","A"-"Z","_","$","\u00e4","\u00c4","\u00f6",
       "\u00d6","\u00fc","\u00dc","?","*"]
88    >
    }
```

Auflistung 2: Grammatik der Verknüpfungssprache: Nichtterminale

```
90
start ::=                 function ( <EOL> | <EOF> )
92                      | set_term ( <EOL> | <EOF> )
                        | ( <EOL> | <EOF> )
94
function ::=              function_sum
96                      | function_max
                        | function_min
98                      | function_count
                        | function_select
100                     | function_linkGeometrie
                        | function_linkQuantities
102
function_sum ::=        <SUM> <LPARENT> set_term <RPARENT>
104
function_max ::=        <MAX> <LPARENT> set_term <RPARENT>
106
function_min ::=        <MIN> <LPARENT> set_term <RPARENT>
108
function_count ::=      <COUNT> <LPARENT> set_term <RPARENT>
110
function_select ::=     <SELECT> <LPARENT> set_term <RPARENT>
112
function_linkGeometrie ::=
114                     <LINK_GEOMETRIE> <LPARENT>
                        <STRING_LITERAL> "," set_term ","
                        <STRING_LITERAL> <RPARENT>

116 function_linkQuantities ::=
                        <LINK_QUANTITIES> <LPARENT>
                        <STRING_LITERAL> "," set_term ","
                        set_term "," <FLOATING_POINT_LITERAL>
                        <RPARENT>
118
set_term ::=              setCreationTerm
120                     | setElementList

122 setElementList ::=    <LPARENT_SET> element
                        ( "," element )* <RPARENT_SET>
124
setCreationTerm ::=     <LPARENT_SET>
126                     ( <IDENTIFIER> | setCreationFunction )
                        ":" ( filterList )? <EXTRACTION>
128                     ( feature | <ALL_ATTRIBUTES> |
                        <ALL_VALUES> )
                        <RPARENT_SET>
130
setCreationFunction ::=
```

```
132                           betweenAxisRelation
                            | inZoneRelation
134
    betweenAxisRelation ::=
136                         <BETWEEN_AXIS> <LPARENT>
                            set_term "," set_term <RPARENT>
138
    inZoneRelation ::=      <IN_ZONES> <LPARENT>
140                         set_term "," set_term <RPARENT>

142 filterList ::=          filterTerm ( <AND_SIGN> filterTerm |
                            <OR_SIGN> filterTerm )*
144
    filterTerm ::=            equalsFilter
146                         | notEqualsFilter
                            | greaterFilter
148                         | greaterEqualsFilter
                            | lessFilter
150                         | lessEqualsFilter
                            | definedFilter
152                         | notDefinedFilter
                            | containedInFilter
154                         | notContainedInFilter
                            | (<LPARENT> filterList <RPARENT> )
156
    equalsFilter ::=        feature <EQUALS_FILTER> valueTerm
158
    notEqualsFilter ::=     feature <NOT_EQUALS_FILTER> valueTerm
160
    greaterFilter ::=       feature <GREATER_FILTER> valueTerm
162
    greaterEqualsFilter ::=
164                         feature <GREATER_EQUALS_FILTER>
                            valueTerm

166 lessFilter ::=          feature <LESS_FILTER> valueTerm

168 lessEqualsFilter ::=    feature <LESS_EQUALS_FILTER> valueTerm

170 definedFilter ::=       feature <DEFINED_FILTER>

172 notDefinedFilter ::=    feature <NOT_DEFINED_FILTER>

174 containedInFilter ::= feature <CONTAINED_IN_FILTER>
                            ( set_term | interval )
176
    notContainedInFilter ::=
178                         feature <NOT_CONTAINED_IN_FILTER>
                            ( set_term | interval )
```

```
180
    feature ::=             <IDENTIFIER>
182
    interval ::=            <LPARENT_INTERVAL> element <AND_SIGN>
184                         element <RPARENT_INTERVAL>

186 valueTerm ::=           element

188 element ::=               <INTEGER_LITERAL>
                            | <FLOATING_POINT_LITERAL>
190                         | <DATE_LITERAL>
                            | <STRING_LITERAL>
```

B Grammatik des Makroprozessors in erweiterter Backus-Naur-Form

Auflistung 3: Grammatik des Makroprozessors: Terminale

```
TOKEN_MGR_DECLS : {
        int macroLevel=0; // used to count nested macros
}

<DEFAULT> SPECIAL : {
  <MACRO_START: "#"> {macroLevel++;}: MACRO
}

<DEFAULT> TOKEN : {
  <ANY: ~["#"]>
}

<DEFAULT> TOKEN [IGNORE_CASE] : {
  <LIST_COMMAND: "listMacros">
}

<DEFAULT> TOKEN : {
  <ASSIGNMENT: ":=">
}

<MACRO> SKIP : {
  " "
| "\r"
| "\t"
| "#"{macroLevel++;}
}

<MACRO> TOKEN : {
  <LPARENT: "(">
| <RPARENT: ")"> {if(--macroLevel==0) SwitchTo(DEFAULT);}
| <COMMA: ",">
}

<MACRO> TOKEN : {
  <INTEGER_LITERAL: (["+","-"])? ["0"-"9"] (["0"-"9"])*>
| <FLOATING_POINT_LITERAL:
    (["+","-"])? ((["0"-"9"])+ "." (["0"-"9"])* (<EXPONENT>)?
  | "." (["0"-"9"])+ (<EXPONENT>)? | (["0"-"9"])+ <EXPONENT>)
  >
| <#EXPONENT: ["e","E"] (["+","-"])? (["0"-"9"])+>
| <STRING_LITERAL:
    "\"" (~["\"","\\","\n","\r"]
```

```
    | "\\" ["n","t","b","r","f","\\","\'","\""])+ "\""
    >
| <DATE_LITERAL:
    <DATE_ONLY_LITERAL> (" " <TIME_ONLY_LITERAL>)?>
| <#DATE_ONLY_LITERAL:
    ("0" ["1"-"9"]
    | ["1"-"2"] ["0"-"9"]
    | "3" ["0"-"1"]) "." ("0" ["1"-"9"]
    | "1" ["0"-"2"]) "." ["0"-"9"]["0"-"9"]["0"-"9"]["0"-"9"]
        >
| <#TIME_ONLY_LITERAL:
    ("0" ["0"-"9"]
    | "1" ["0"-"9"]
    | "2" ["0"-"3"]) ":" ("0" ["0"-"9"]
    | ["1"-"5"] ["0"-"9"])
    >
}

<MACRO> TOKEN : {
  <IDENTIFIER: <LETTER> (<LETTER> | ["0"-"9"])*>
| <#LETTER:
   ["a"-"z","A"-"Z","_","$","\u00e4","\u00c4","\u00f6",
   "\u00d6","\u00fc","\u00dc","?","*"]
  >
}
```

Auflistung 4: Grammatik des Makroprozessors: Nichtterminale

```
start ::=                 command <EOF>
                        | macro_definition
                        | query

query ::=               ( <ANY> | macro_call )* <EOF>

macro_call ::=          <IDENTIFIER> <LPARENT>
                        ( parameter_value )? ( <COMMA>
                        parameter_value )*
                        <RPARENT>

macro_definition ::=    <IDENTIFIER> <LPARENT>
                        ( parameter_name )? ( <COMMA>
                        parameter_name )*
                        <RPARENT> <ASSIGNMENT> ( <ANY> |
                        macro_call )* <EOF>

parameter_name ::=      <IDENTIFIER>
```

```
86 parameter_value ::=      <INTEGER_LITERAL>
                          | <FLOATING_POINT_LITERAL>
88                        | <DATE_LITERAL>
                          | <STRING_LITERAL>
90                        | macro_call

92 command ::=            <LIST_COMMAND>
```

C Datenabbildung in IFC

Die nachfolgenden Abbildungen dienen der Verdeutlichung der gewählten Datenmodellierung in IFC.

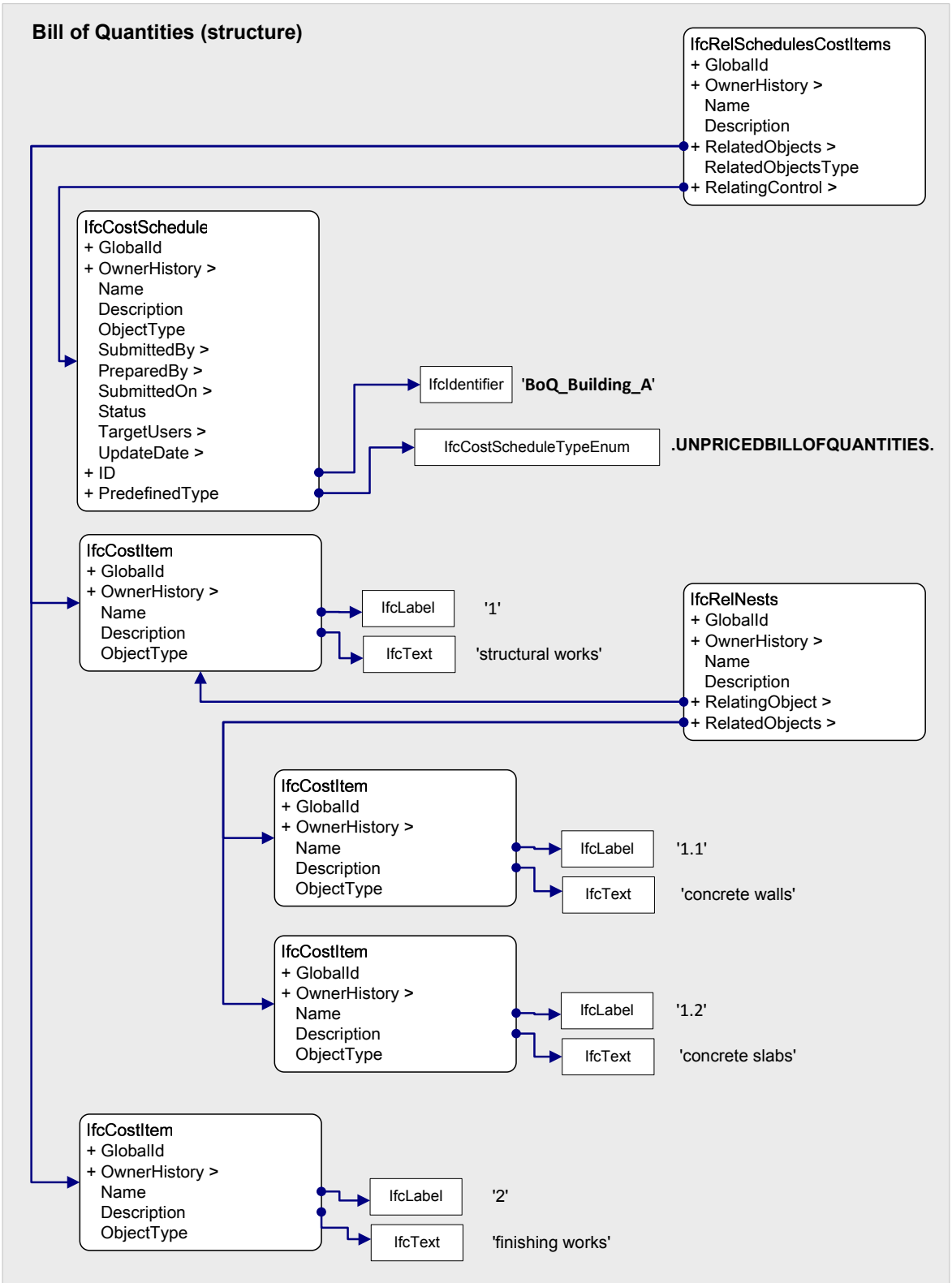


Abbildung C.1: Abbildung der LV-Struktur in IFC

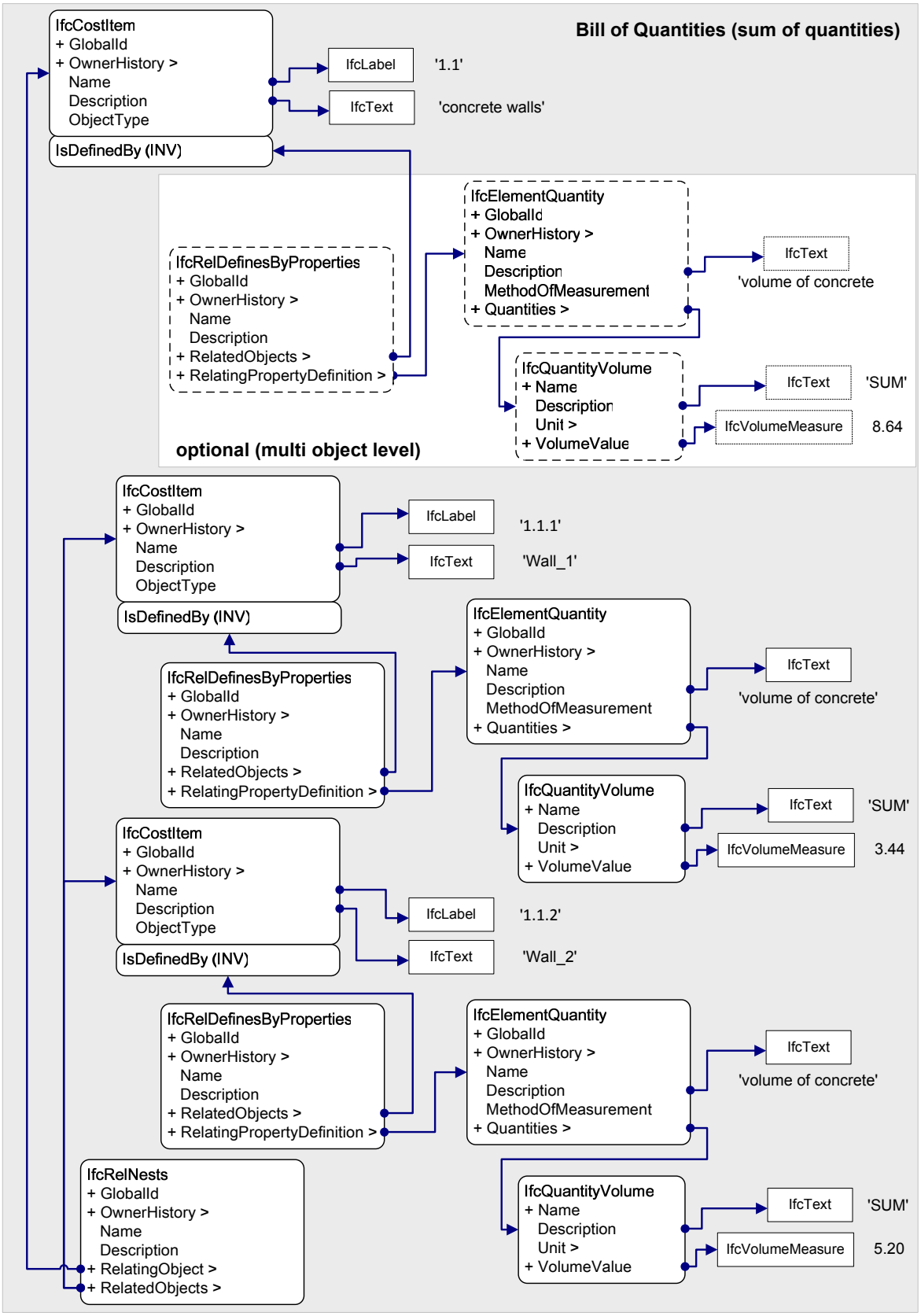


Abbildung C.2: LV-Einzelmengen und Summenbildung in IFC

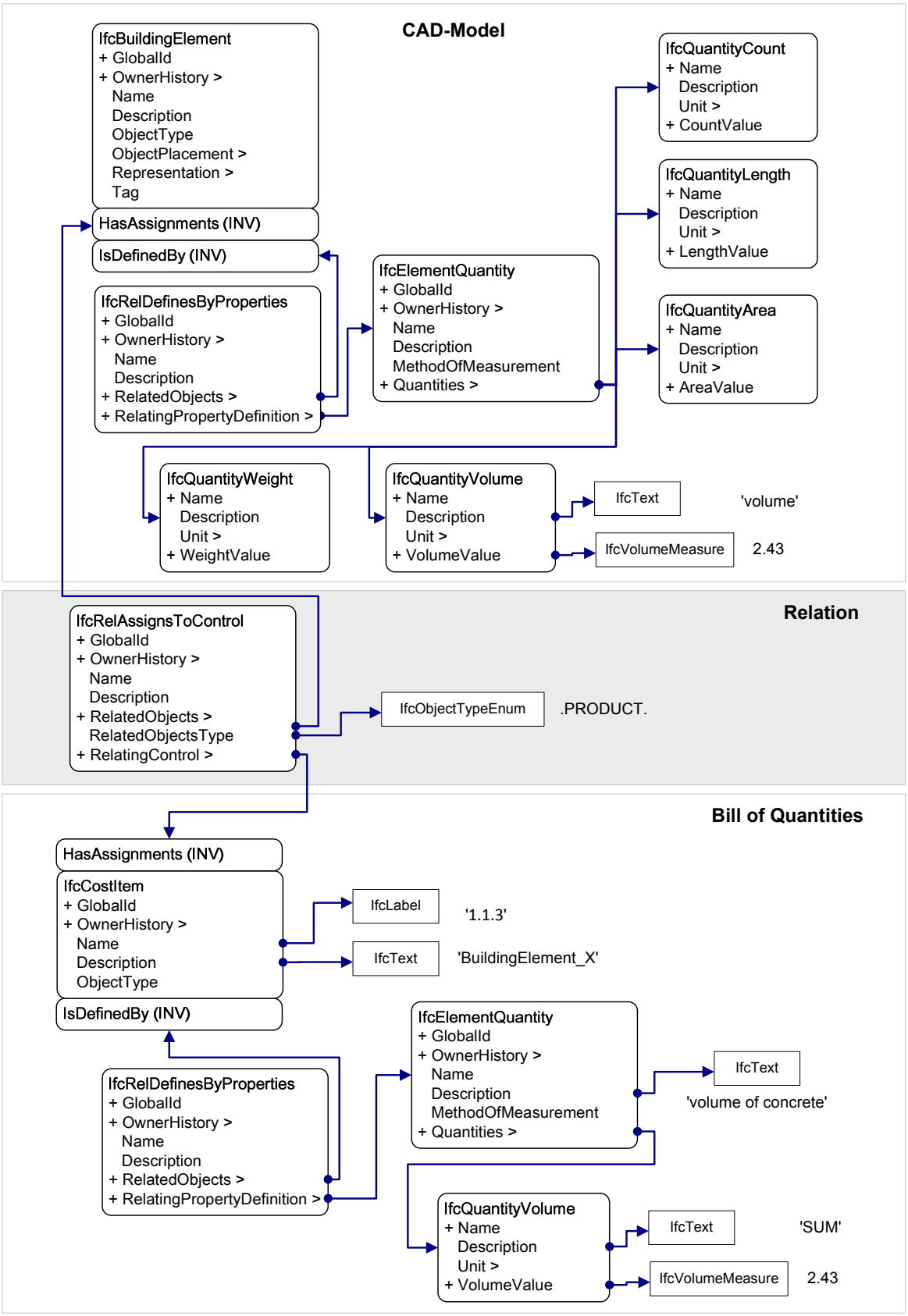


Abbildung C.3: Verknüpfung von LV-Mengen und CAD-Objekten in IFC

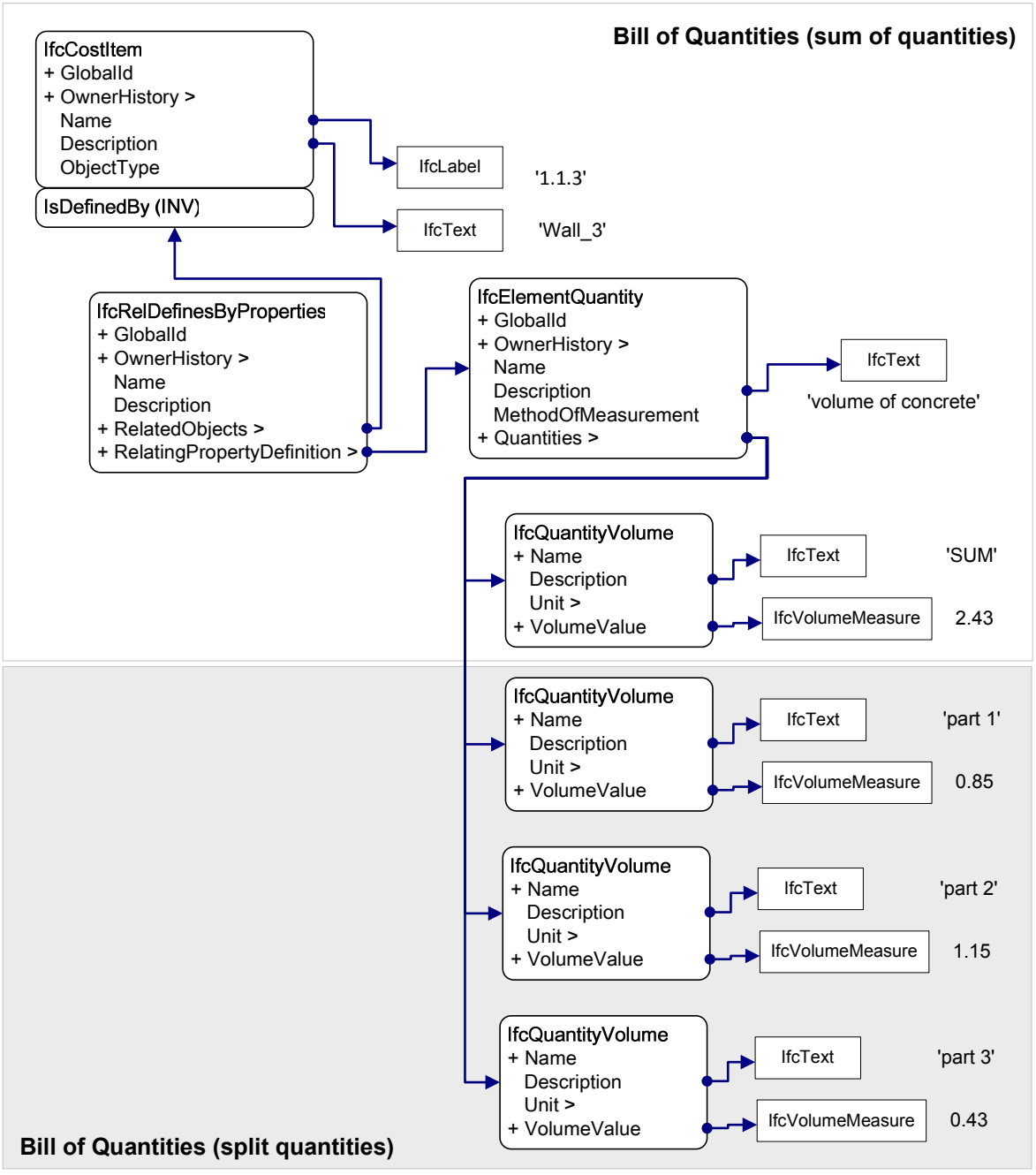


Abbildung C.4: Verfeinerte LV-Mengen infolge Objektteilung in IFC

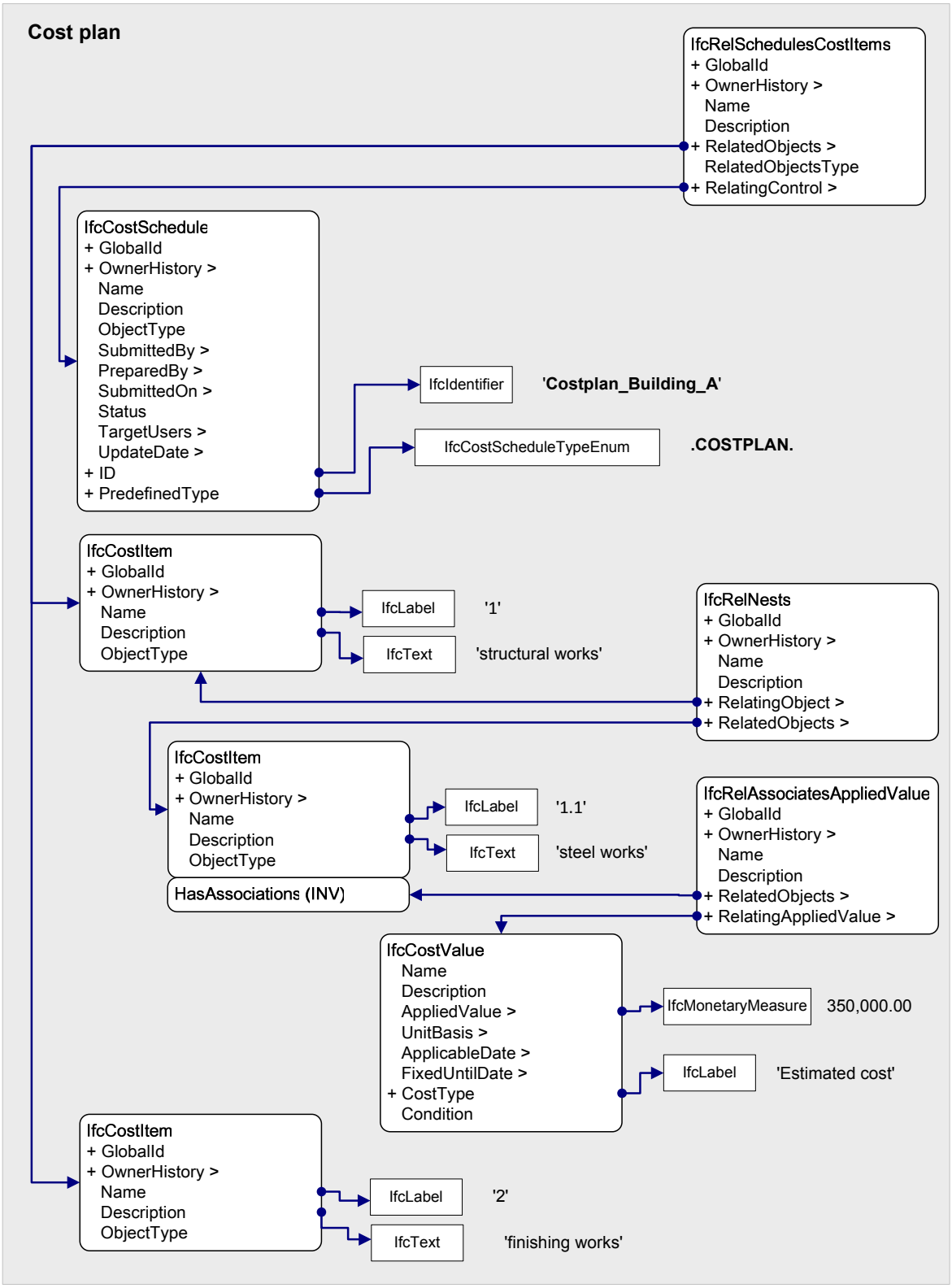


Abbildung C.5: Abbildung grober Kostendaten in IFC

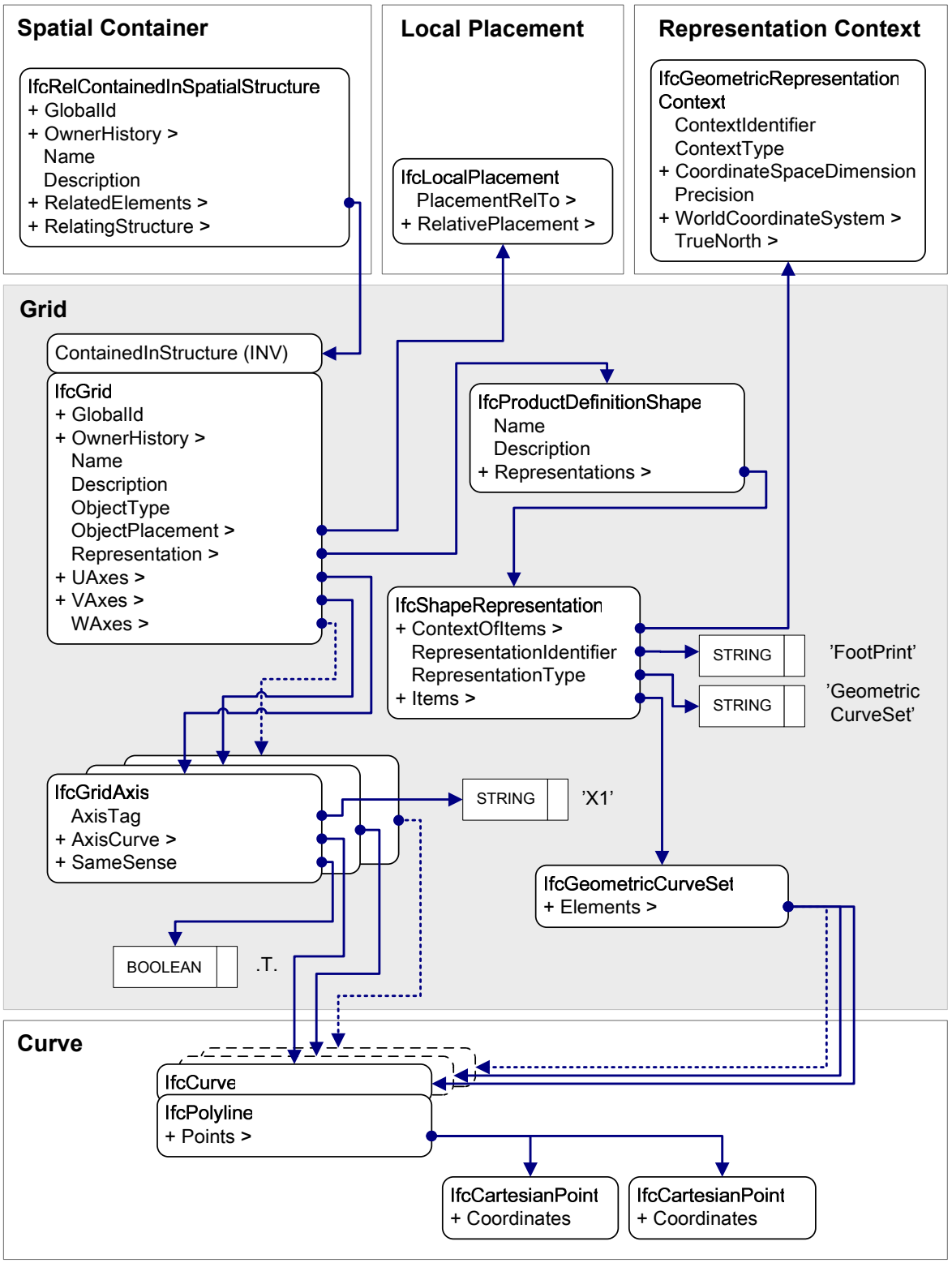


Abbildung C.6: Abbildung eines Achsrasters in IFC

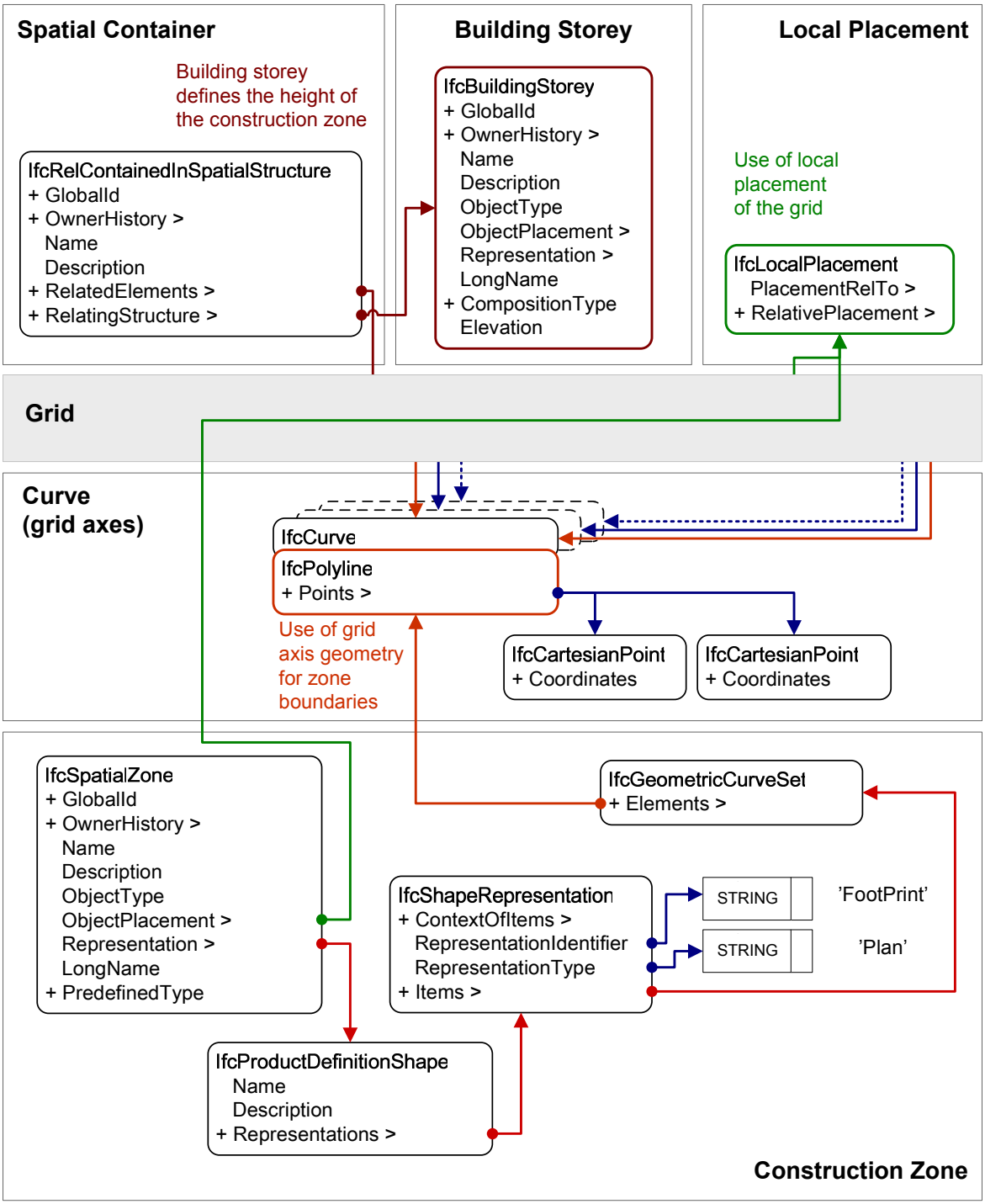


Abbildung C.7: Definition von Zonen auf Basis eines Achsrasters in IFC

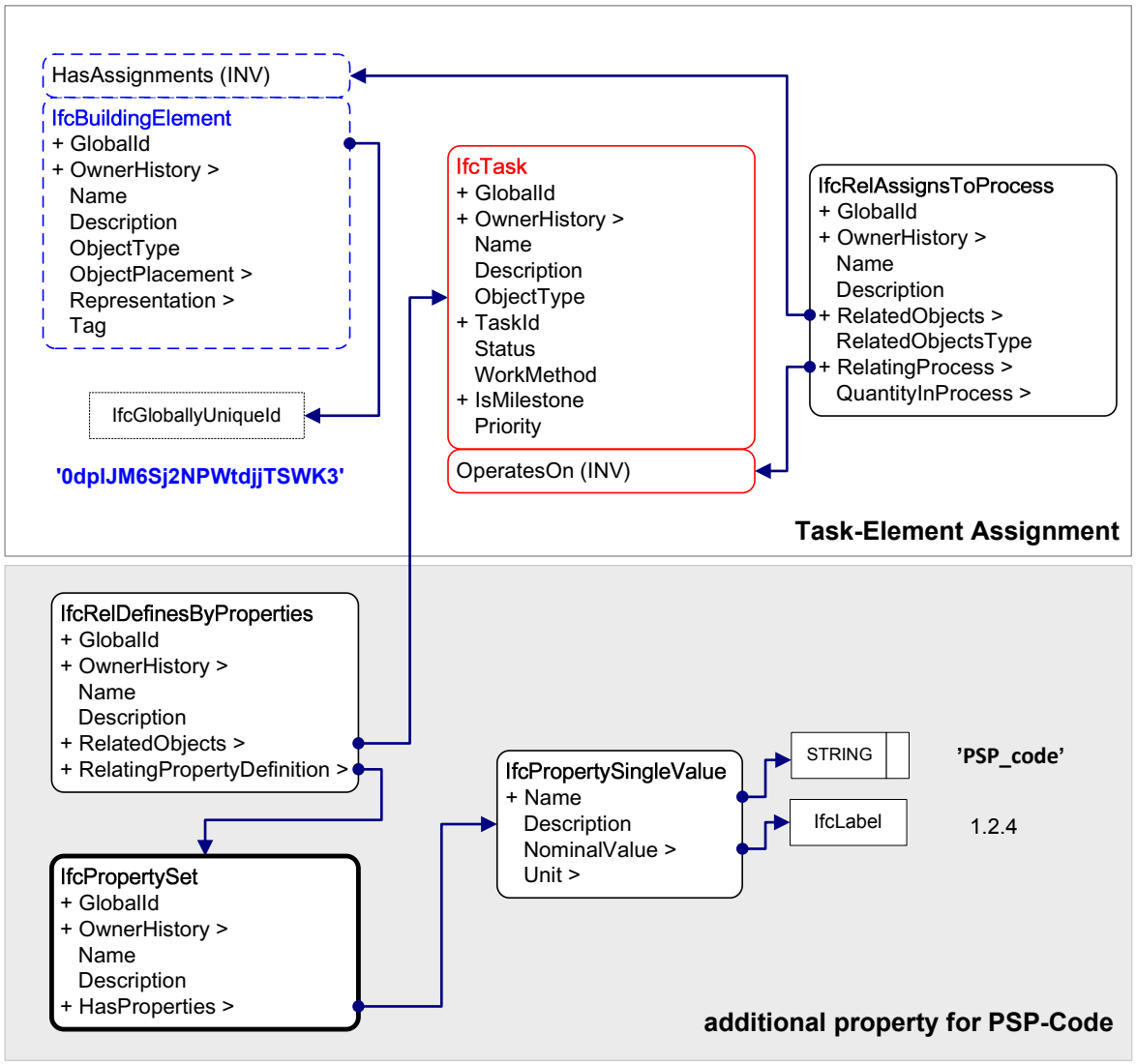


Abbildung C.8: oben: Verknüpfung von Terminplanvorängen mit Bauteilobjekten des CAD-Modells in IFC, unten: Abbildung eines Projektstrukturplan-Codes von Terminplanvorgängen in IFC

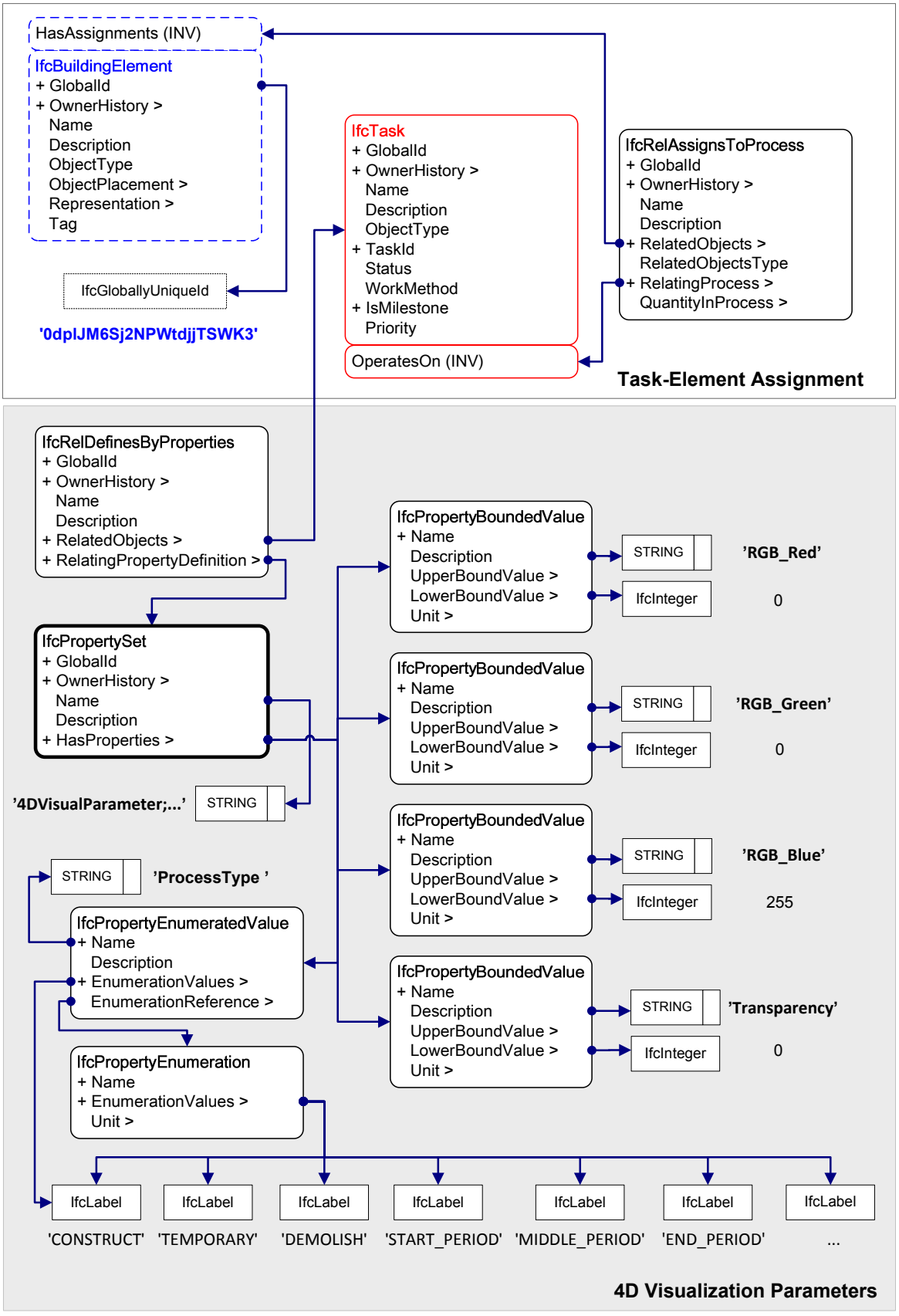


Abbildung C.9: Abbildung von 4D Visualisierungsparametern in IFC

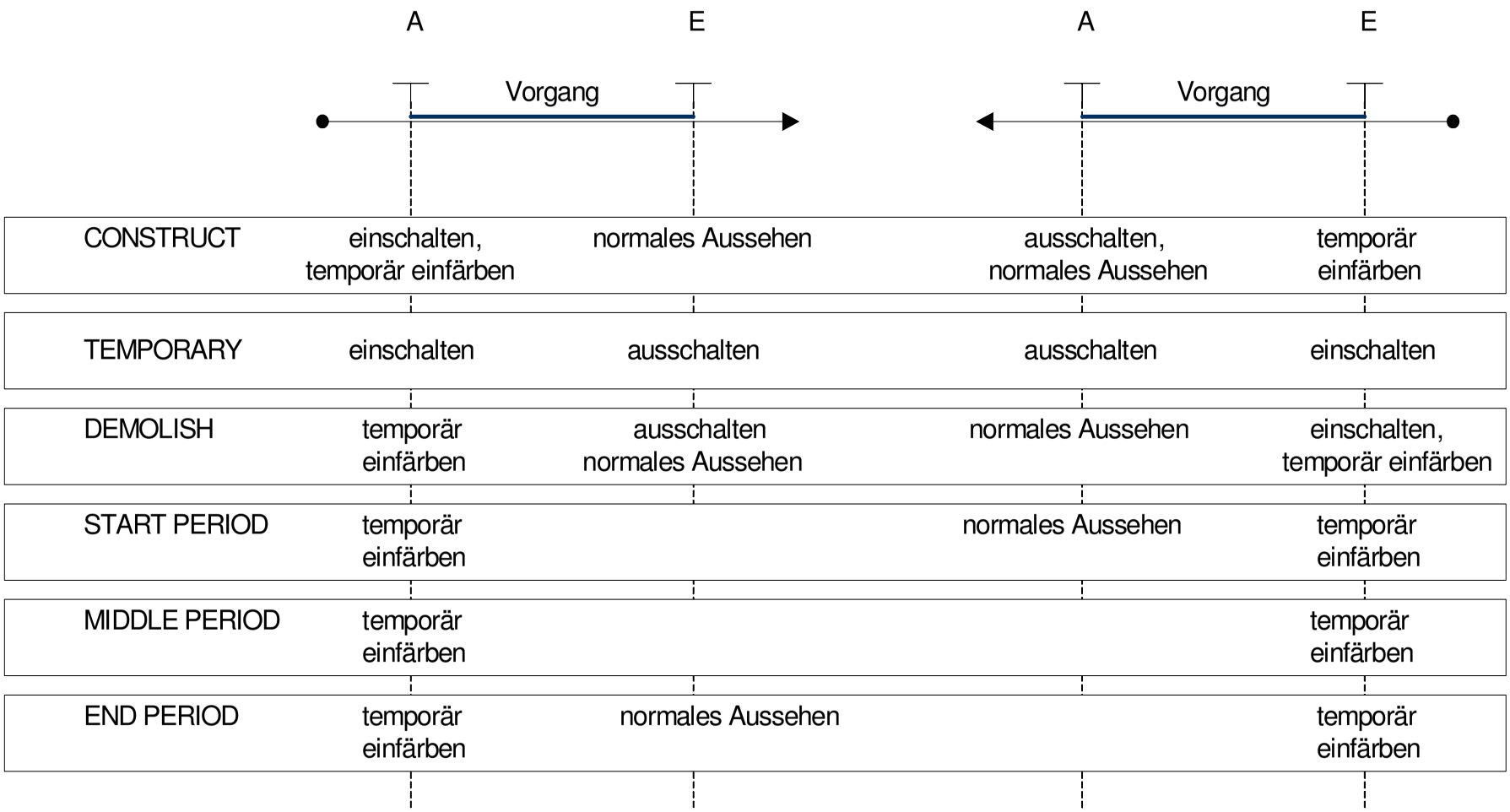


Abbildung C.10: Veränderung der Darstellung von Bauteilobjekten zum Anfang (A) und Ende (E) eines Vorgangs in Abhängigkeit der Zeitlaufrichtung und des Prozesstyps (aus [Koschorrek u. a. 2008])

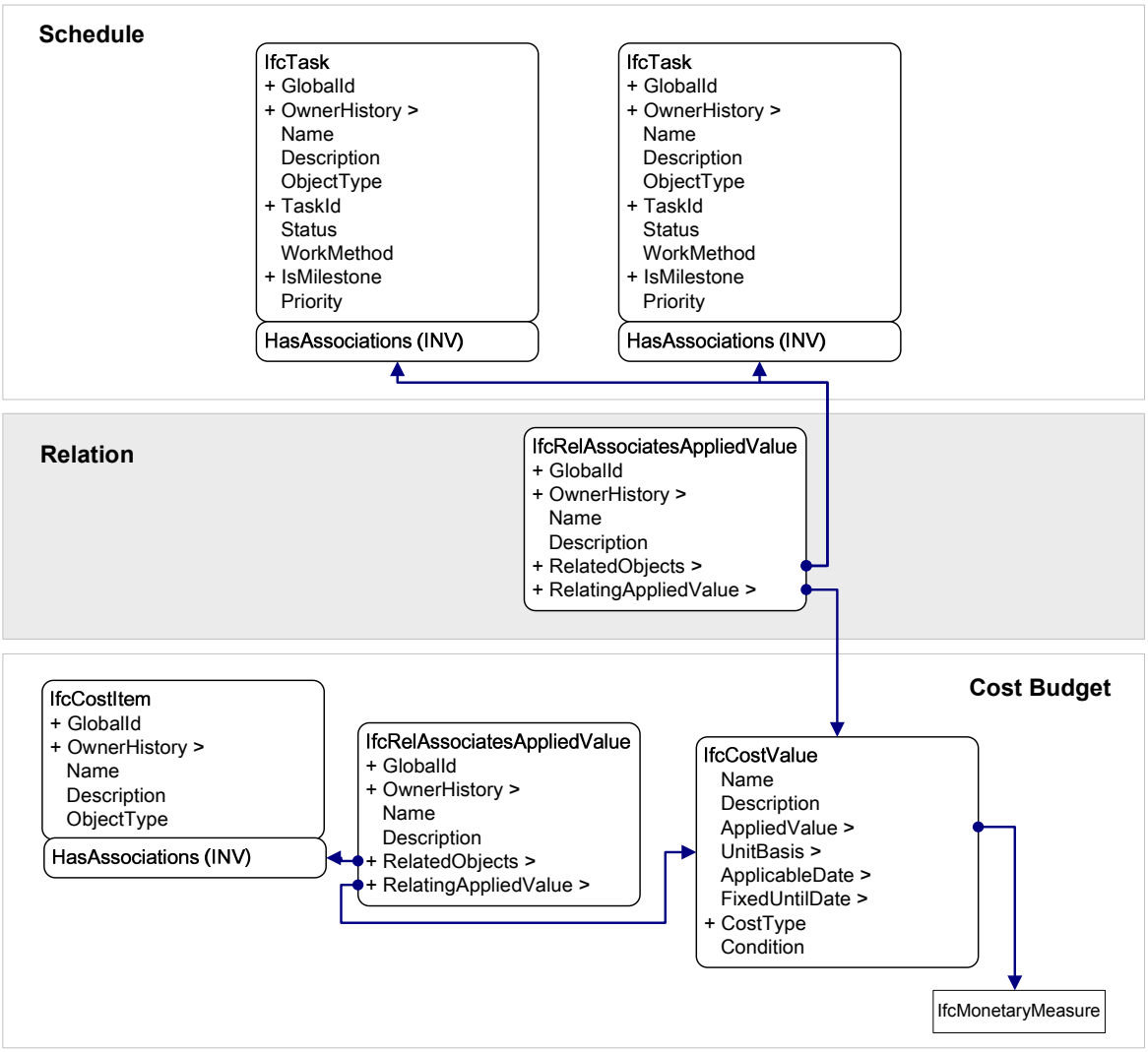


Abbildung C.11: Direkte Zuordnung eines Kostenwerts zu ein oder mehreren Terminplanvorgängen in IFC

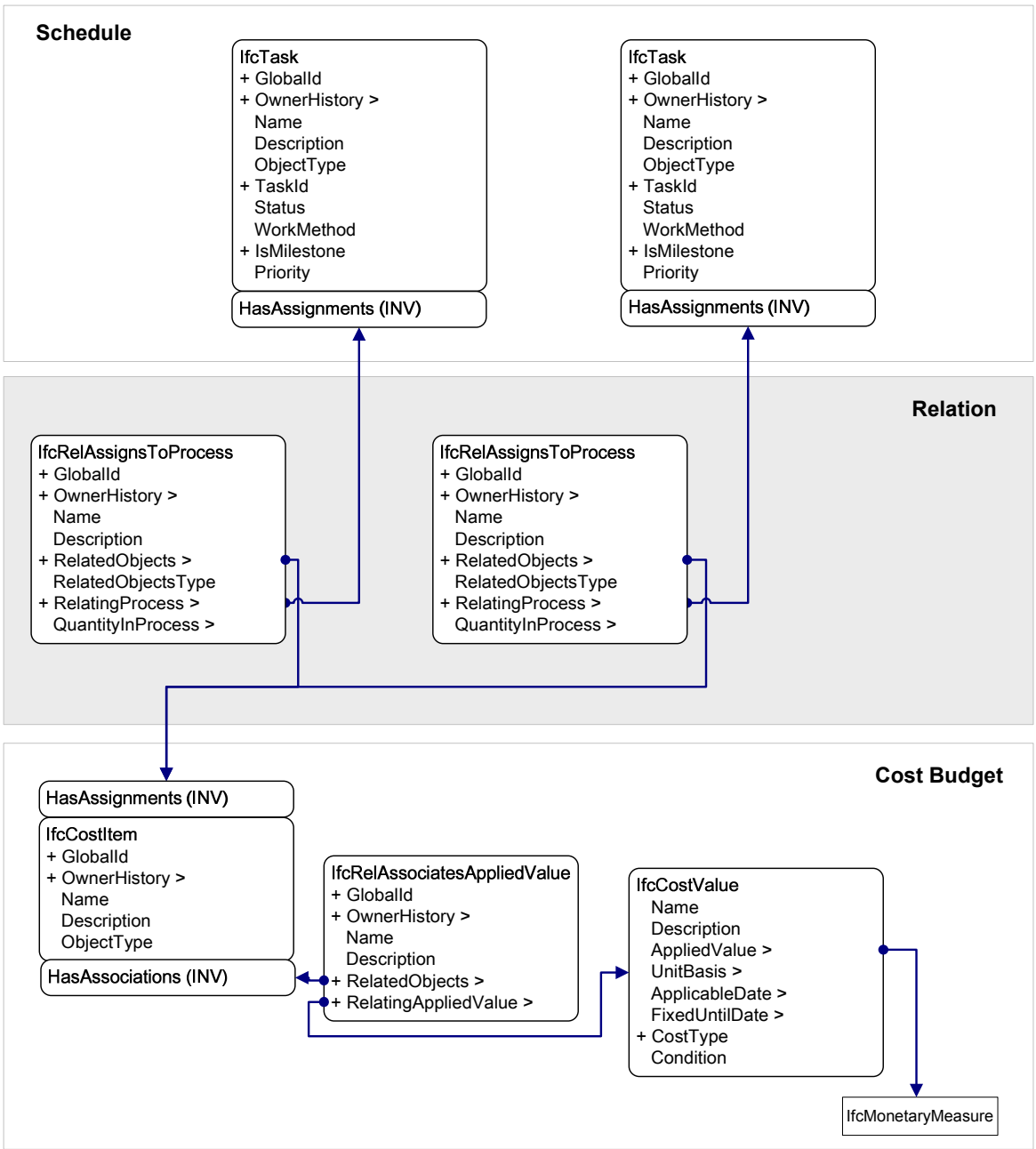


Abbildung C.12: Indirekte Zuordnung eines Kostenwerts zu ein oder mehreren Terminplanvorgängen in IFC

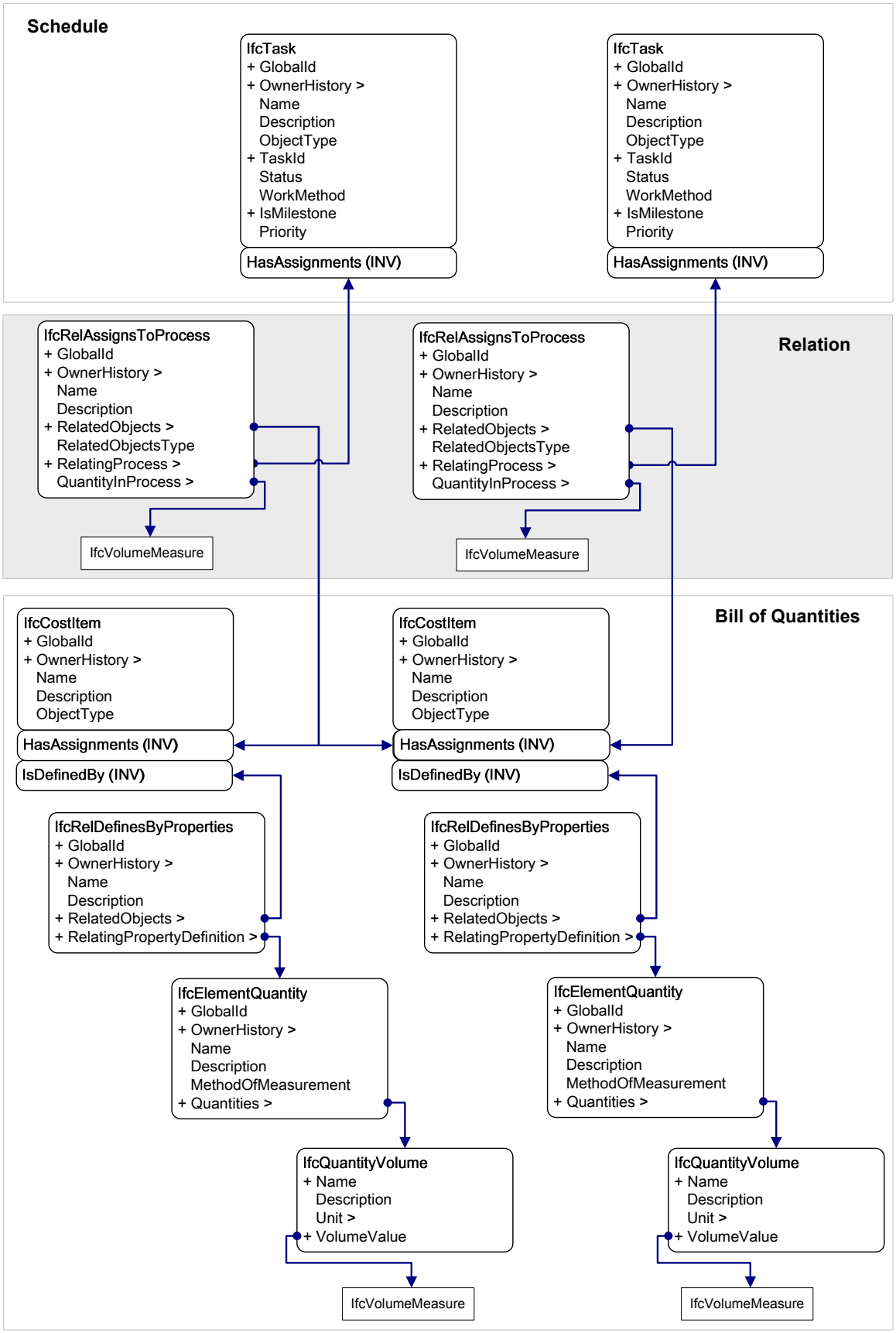


Abbildung C.13: Zuordnung von LV-Mengen zu Terminplanvorgängen in IFC

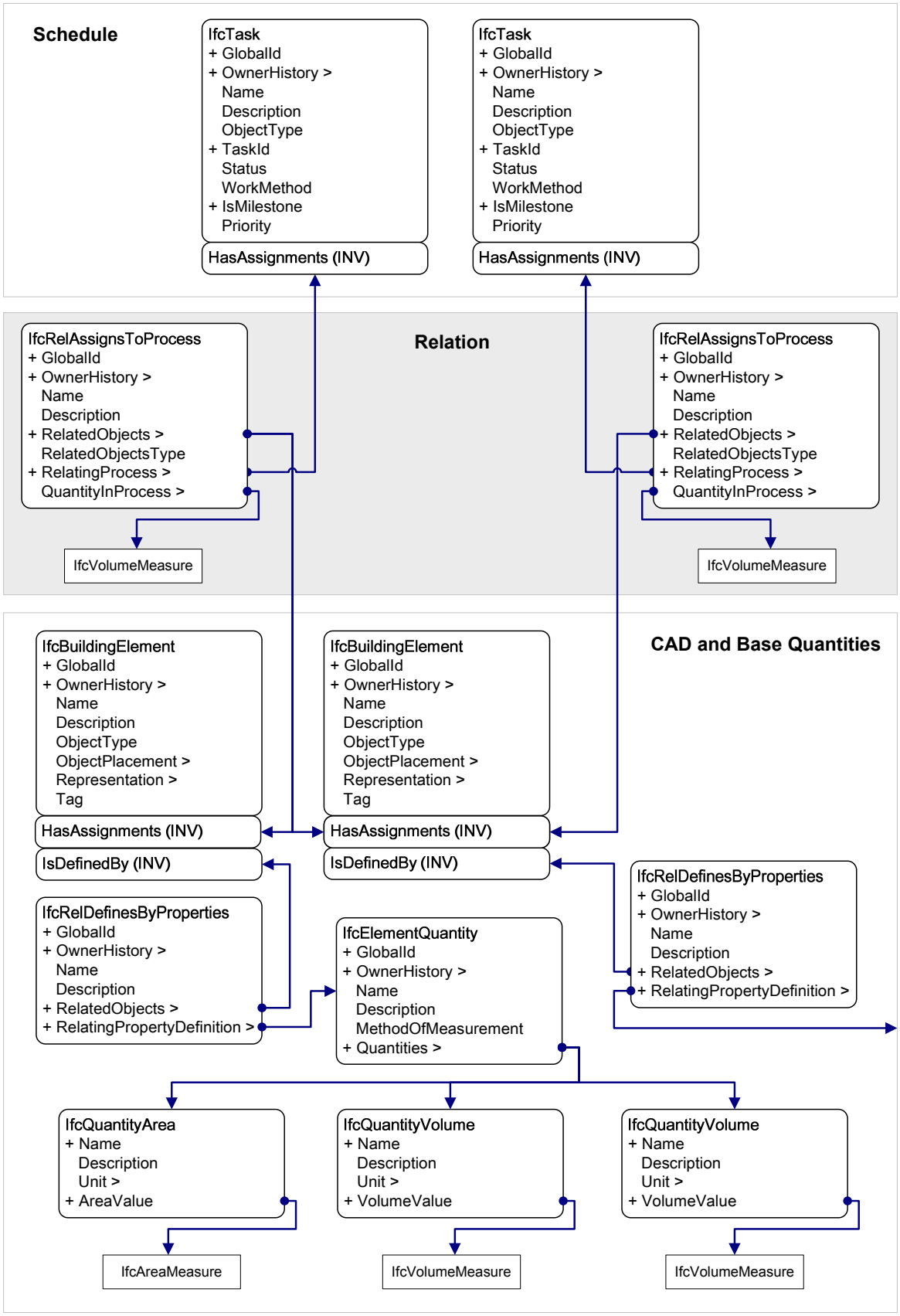


Abbildung C.14: Zuordnung von Basismengen und Terminplanvorgängen in IFC

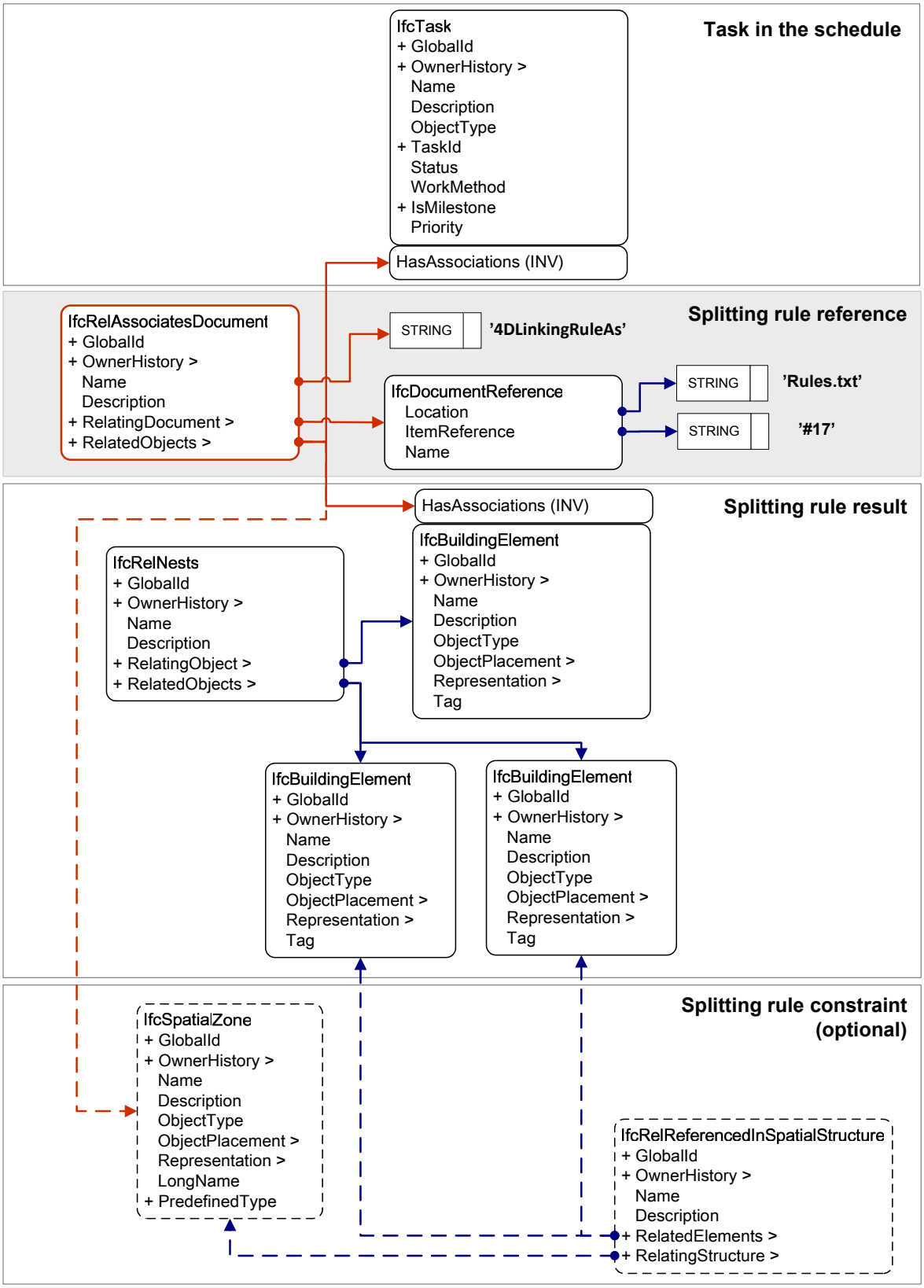


Abbildung C.15: Regelbasierte Verknüpfung in IFC

D Datenbankschema

Aktuell geladene Modellversionen

LM

Model	ID
CAD	1
LV	4
TP	7
4D	1
...	

Die gezeigten Daten dienen lediglich der Verdeutlichung des inhaltlichen Zusammenhangs. In der Umsetzung werden GUIDs in komprimierter Textform gemäß der IFC verwendet.

Daten und Historie des Teilmodells CAD (für andere Teilmodelle analog)

Referenzdatenbestand

CAD_OP

Objekt_ID	Attributname	Wert
ALR10348	Material	Beton
ALR10348	Dicke	0,30
ALR10400	...	

CAD_OS

Objekt_ID	Version	Status
ALR10348	1	0
ALR10370	4	1
ALR10400	...	

Metadaten der Modellversionen

CAD_MV

Modellversion	Kommentar	Pfad	Datum
1	Erstversion	C:\test1.ifc	01.06.09 12:13
2	Alternative 2	C:\test2.ifc	03.06.09 09:18
3	...		

Modellversionsgraph

CAD_DMV

src_MV	dest_MV	DMV_ID
1	2	AX32B7
2	3	FYT3LF
2	4	...

CAD_DMO

DMV_ID	Objekt_ID	del_OV	gen_OV
AX32B7	ALR10348	1	2
AX32B7	ALR10400	3	0
FYT3LF	...		

Objektversionsgraph

CAD_DOV

Objekt_ID	src_OV	dest_OV	DOV_ID
ALR10348	2	3	HI327B
ALR10400	1	4	73B3X
ALR10401	...		

CAD_DOO

DOV_ID	Attributname	del	gen
HI3274	Material	Naturstein	Beton
HI3274	Länge	2,01	-
73B3X	...		

Abkürzung	Bedeutung
LM	Loaded Models
CAD	CAD-Daten
LV	Leistungsverzeichnis
TP	Terminplan
4D	4D-Simulation
_MV	Modellversionen
_OP	Objektpool
_OS	Objektstatus
_DMV	Delta-Modellversion
_DMO	Delta-Modelloperation
_DOV	Delta-Objektversion
_DOO	Delta-Objektoperation

Spalte	**Datentyp**
Modellversion	Integer
Vorgänger	Integer
Kommentar	String
Datum	Datum mit Uhrzeit
Objekt_ID	String
Version	Integer
Status	Boolean
Attributname	String
Wert	Variant
src_MV	Integer
dest_MV	Integer
DVM_ID	String
del_OV	Integer
gen_OV	Integer
src_OV	Integer
dest_OV	Integer
DOV_ID	String
del	String
gen	String

Abbildung D.1: Datenbankschema für den Workspace

E Ehrenwörtliche Erklärung

Ich erkläre hiermit ehrenwörtlich, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Die aus anderen Quellen direkt oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet.

Bei der Auswahl und Auswertung von Material haben mir andere Personen weder entgeltlich noch unentgeltlich geholfen.

Weitere Personen waren an der inhaltlich-materiellen Erstellung der vorliegenden Arbeit nicht beteiligt. Insbesondere habe ich hierfür nicht die entgeltliche Hilfe von Vermittlungs- bzw. Beratungsdiensten (Promotionsberater oder anderer Personen) in Anspruch genommen. Niemand hat von mir unmittelbar oder mittelbar geldwerte Leistungen für Arbeiten erhalten, die im Zusammenhang mit dem Inhalt der vorgelegten Dissertation stehen.

Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt.

Ich versichere ehrenwörtlich, dass ich nach bestem Wissen die reine Wahrheit gesagt und nichts verschwiegen habe.

Weimar, den 25. Oktober 2009 Jan Tulke

F Über den Autor

F.1 Lebenslauf

Persönliche Daten

Name:	Jan Tulke
Geburtstag und -ort:	23. April 1978 in Berlin
Familienstand:	ledig

Schulbildung

09/1990 – 06/1997:	Willi-Graf-Oberschule Berlin-Steglitz, Abschluss: Abitur
09/1984 – 08/1990:	Clemens-Brentano-Grundschule Berlin-Steglitz

Studium

10/1997 – 03/2002:	Bauingenieurwesen an der Technischen Universität Berlin
	Abschluss: Diplom-Ingenieur (Dipl.-Ing.)
	Vertiefung: Bauinformatik, Statik, Grundbau und Bodenmechanik
	Diplomarbeit: Die Berechnung monolithischer Decken-Balken-Systeme; Umsetzung in einer webbasierten Umgebung

Auszeichnung

07/2003:	Erwin-Stephan-Preis, vergeben für überdurchschnittliche Leistungen und kurze Studiendauer

Studienbegleitende Tätigkeit

07/2001 – 08/2002:	Studentischer Mitarbeiter WTM ENGINEERS Berlin (Ingenieurbüro & Prüfstatik)
09/2000 – 07/2002	Promotor der NEMETSCHEK AG Betreuung, Support und Vertrieb von Softwarelizenzen

Beruflicher Werdegang

seit 06/2009:	HOCHTIEF ViCon GmbH Leiter Forschung und Produktentwicklung
06/2006 – 06/2009	HOCHTIEF AG, Unternehmensentwicklung Projektleiter Forschung
11/2002 – 05/2009:	HOCHTIEF Construction AG, Consult Infrastructure Tragwerksplaner im Tief-, Tunnel-, Wasserbau Forschung und Entwicklung

Berufsbegleitende Tätigkeit

seit 06/2006	externe Promotion Bauhaus-Universität Weimar Professur Informatik im Bauwesen
02/2007 – 04/2008:	Dozent an der HOCHTIEF Akademie Fach Bauinformatik
08/2003 – 12/2003:	Forschungsaufenthalt University of California San Diego, USA Fachbereich Mechanical Engineering

F.2 Publikationen

[Benning u. a. 2008] Benning, Pierre ; Dehlin, Stefan ; Tulke, Jan ; Aberg, Peter ; Ryd, Nina ; Laine, Tuomas ; Jaeger, Johann ; Brandt, Tim: Collaboration Processes - A State of the Art / InPro-Project. Version: 2008. www.inpro-project.eu. 2008. – Forschungsbericht

[Benning u. a. 2009] Benning, Pierre ; Dumoulin, Claude ; Dehlin, Stefan ; Tulke, Jan ; Aberg, Peter ; Fristedt, Sven ; Holopainen, Riikka ; Schumacher, Sebastian ; Brandt, Tim ; Rommen, Lisanne: Collaboration Processes - Framework for Collaboration / InPro-Project. Version: 2009. www.inpro-project.eu. 2009. – Forschungsbericht

[Fijneman u. a. 2008] Fijneman, Martin ; Matthyssen, Arne ; Verhofstad, Frank ; Tulke, Jan: Process Mapping for Key Process 4 - Construction Scheduling / InPro-Project. 2008. – Forschungsbericht

[Houttu u. a. 2009] Houttu, Juha-Matti ; Dehlin, Stefan ; Ferino, Martial ; Fies, Bruno ; Nummelin, Olli ; Tulke, Jan ; Pfitzner, Markus ; Laine, Tuomas: Overview of Early Design Applications / InPro Project. 2009. – Forschungsbericht

[Koschorrek u. a. 2008] Koschorrek, Sabine ; Ester, Alexandra ; Tauscher, Eike ; Tulke, Jan: IFC-basiertes Modell für die 4D-Bauablaufanimation. In: *20. Forum Bauinformatik, Dresden*, 2008

[Langetepe u. a. 2009] Langetepe, Elmar ; Penninger, Rainer ; Tulke, Jan: Computing the visibility area between two simple polygons in linear time. In: *26th European Workshop on Computational Geometry, Dortmund*, 2009

[Liebich u. a. 2009] Liebich, Thomas ; Weise, Matthias ; Laine, Tuomas ; Jokela, Markku ; Tulke, Jan ; Böhms, Michel ; Bonsma, Peter ; Nummelin, Olli ; Houttu, Juha-Matti: InPro Building Information Model / InPro-Project. Version: 2009. www.inpro-project.eu. 2009. – Forschungsbericht

[Nour u. a. 2009] Nour, Mohamed ; Ferino, Martial ; Dehlin, Stefan ; Tulke, Jan ; Pfitzner, Markus ; Houttu, Juha-Matti ; Nummelin, Olli ; Laine, Tuomas: Overview of Information Management Applications, Including Object-Based Version Management / InPro-Project. Version: 2009. www.inpro-project.eu. 2009. – Forschungsbericht

[Outters u. a. 2007] Outters, Nils ; Woksepp, Stefan ; Liljeström, Kimmo ; Benning, Pierre ; Dumoulin, Claude ; Ferino, Martial ; Tulke, Jan ; Nummelin, Olli ; Ryd, Nina ; Laine, Tuomas ; Soumunen, Piia ; Jaeger, Johan ; Sulzer, Christian ; Pfitzner, Markus ; Neuberg, Frank ; Verhofstad, Frank: Use Cases / InPro-Project. Version: 2007. www.inpro-project.eu. 2007. – Forschungsbericht

[Pfitzner u. a. 2007] Pfitzner, Markus ; Neuberg, Frank ; Tulke, Jan ; Nummelin, Olli ; Benning, Pierre: Analysis of Existing Software Tools for Early Design / InPro-Project. Version: 2007. www.inpro-project.eu. 2007. – Forschungsbericht

[Schade u. a. 2009] Schade, Jutta ; Heikkilä, Katarina ; Benning, Pierre ; Schunke, Marcus ; Schreyer, Marcus ; Dehlin, Stefan ; Sormunen, Piia ; Hirvonen, Tapio ; Meiling, John ; Tulke, Jan ; Holopainen, Riikka: The InPro Lifecycle Design Framework for Buildings / InPro-Project. Version: 2009. `www.inpro-project.eu`. 2009. – Forschungsbericht

[Schreyer u. a. 2009] Schreyer, Marcus ; Benning, Pierre ; Ryd, Nina ; Tulke, Jan ; Jaeger, Johann ; Brandt, Tim: A Smart Decision Making Framework for Building Information Models / InPro-Project. Version: 2009. `www.inpro-project.eu`. 2009. – Forschungsbericht

[Schumacher u. a. 2009] Schumacher, Sebastian ; Couto, Ana ; Flaspöhler, Jan ; Jaeger, Johan ; Benning, Pierre ; Ryd, Nina ; Sandesten, Stefan ; Fristedt, Sven ; Tulke, Jan ; Chahrour, Racha ; Grys, Robert ; Fijneman, Martin ; Schade, Jutta ; Olofsson, Thomas ; Pfitzner, Markus ; Outters, Nils ; Houttu, Juha-Matti ; Nummelin, Olli: Communication Strategies / InPro-Project. 2009. – Forschungsbericht

[Theiler u. a. 2009] Theiler, Michael ; Tauscher, Eike ; Tulke, Jan ; Riedel, Thomas: Visualisierung von IFC-Objekten mittels Java3D. In: *21. Forum Bauinformatik, Karlsruhe*, 2009

[Tulke 2001] Tulke, Jan: *Berechnung monolithischer Decken-Balken-Systeme - Umsetzung in einer webbasierten Umgebung*, Technische Universität Berlin, Diplomarbeit, 2001. `http://www.jan-tulke.de/Diplomarbeit/homepage/Diplomarbeit.html`

[Tulke u. Hanff 2007] Tulke, Jan ; Hanff, Jochen: 4D Construction Sequence Planning - New Process and Data Model. In: *24th cib W78 Conference "Bringing ITC knowledge to work"*, 2007, 79-84

[Tulke u. a. 2008a] Tulke, Jan ; Nour, Mohamed ; Beucke, Karl: Decomposition of BIM objects for scheduling and 4D simulation. In: *7th European Conference on Product and Process Modelling (ECPPM)*, 2008, S. 653–660

[Tulke u. a. 2008b] Tulke, Jan ; Nour, Mohamed ; Beucke, Karl: A Dynamic Framework for Construction Scheduling based on BIM using IFC. In: *17th IABSE Congress "Creating and Renewing Urban Structures - Tall Buildings, Bridges and Infrastructure*, 2008 (978-3-85748-118-5)

[Weise u. a. 2009a] Weise, M. ; Liebich, T. ; Tulke, J. ; Bonsma, P.: Discussion paper: IFC support for model-based scheduling / Public specification of the NMP EU project InPro (IP 026716-2). 2009. – Forschungsbericht

[Weise u. a. 2009b] Weise, M. ; Liebich, T. ; Tulke, J. ; Bonsma, P.: IFC Support for Model-based Scheduling. In: *26th cib W78 Conference "Managing IT in Construction"*, 2009

Verzeichnisse

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Algorithmen

Verzeichnis der Auflistungen

Literaturverzeichnis

[Aalami u. a. 1998] Aalami, Florian B. ; Fischer, Martin A. ; Kunz, John C.: AEC 4D Production Model: Definition and Automated Generation / CIFE, Stanford University. 1998. – Forschungsbericht

[Abdalla u. Powell 1991] Abdalla, Jamal A. ; Powell, Graham H.: Versions and Alternatives of Structural Engineering design Objects / CIFE, Stanford University. 1991. – Forschungsbericht

[Abdul-Rahman u. Pilouk 2008] Abdul-Rahman, A. ; Pilouk, M.: *Spatial Data Modelling for 3D GIS*. Springer, 2008. – ISBN 9783540741664

[Adachi 2002] Adachi, Yoshinobu: Overview of partial model query language / VTT Technical Research Center of Finland Building and Transport SECOM CO., Ltd. 2002 (VTT-TEC-ADA-12). – Forschungsbericht

[Adeli u. Karim 2001] Adeli, Hojjat ; Karim, Asim: *Construction Scheduling, Cost Optimization, and Management - A new model based on neurocomputing and object technologies*. Spon Press, 2001. – ISBN 041524417X

[Akbas 2003] Akbas: *Geometry-based modeling and simulation of construction processes*, Stanford University, Diss., 2003. `http://www.stanford.edu/group/CIFE/online.publications/TR151.pdf`

[Akbas u. Fischer 2002] Akbas, Ragip ; Fischer, Martin: Construction Zone Generation Mechanisms and Applications. In: *19th International Symposium on Automation and Robotics in Construction (ISARC)*, 2002

[Akinci u. a. 2000] Akinci, Burcu ; Fischer, Martin ; Levitt, Raymond ; Carlson, Robert: Formalization and Automation of Time-Space Conflict Analysis, Working Paper Nr.59 / CIFE Standford University. 2000. – Forschungsbericht

[Allen u. a. 2005] Allen, Richard K. ; Becerik, Burcin ; Pollalis, Spiro N. ; Schwegler, Benedict R.: Promise and Barriers to Technology Enabled and Open Project Team Collaboration. In: *Journal of Professional Issues in Engineering Education and Practice* 131-4 (2005), S. 301–311

[Andrew 1979] Andrew, Alex: Another efficient algorithm for convex hulls in two dimensions. In: *Information Processing Letters* 9 (1979), S. 216–219

[Bauwirtschaft 1982] Bauwirtschaft, Tarifvertragsparteien der deutschen Bauwirtschaft: *Arbeitszeit-Richtwerte Hochbau*. Zeittechnik, 1982-2009

[Beer 2005] Beer, Daniel G.: *Systementwurf für verteilte Applikationen und Modelle im Bauplanungsprozess.* Weimar, Bauhaus-Universität Weimar, Diss., 2005

[Benning u. a. 2009] Benning, Pierre ; Dumoulin, Claude ; Dehlin, Stefan ; Tulke, Jan ; Aberg, Peter ; Fristedt, Sven ; Holopainen, Riikka ; Schumacher, Sebastian ; Brandt, Tim ; Rommen, Lisanne: Collaboration Processes - Framework for Collaboration / InPro-Project. Version: 2009. `www.inpro-project.eu`. 2009. – Forschungsbericht

[de Berg u. a. 2008] Berg, Mark de ; Kreveld, Marc van ; Cheong, Otfried ; Overmars, Mark: *Computational Geometry Algorithms and Applications.* Springer Verlag, 2008. – ISBN 978354077973

[Bernsdorff-Diehl-Klein 1998] Bernsdorff-Diehl-Klein, Ingenieurgesellschaft: *Terminplanung: Zeitbedarfswerte für Bauleistungen im Hochbau.* Ministerium Bauen und Wohnen des Landes Nordrhein-Westfalen, 1998. – ISBN 3930860031

[Bielfeld 2009] Bielfeld, Bert: *Basics Terminplanung.* Birkhäuser, 2009. – ISBN 9783764388720

[Borrmann 2008] Borrmann, André: *Computerunterstützung verteilt kooperativer Bauplanung durch Integration interaktiver Simulationen und räumlicher Datenbanken.* 2008

[Brüderlin u. Meier 2001] Brüderlin, Beat ; Meier, Andreas: *Computergrafik und Geometrisches Modellieren.* Teubner, 2001. – ISBN 3519029480

[Brinkhoff 2005] Brinkhoff, Thomas: *Geodatenbanksysteme in Theorie und Praxis.* Wichmann, 2005. – ISBN 387907433X

[buildingSMART 2003] buildingSMART: Case Studies - The CORENET project in Singapore. Version: 2003. `http://coreweb.nhosp.no/buildingsmart.no/html/files/Case_studie_Singapore.pdf`. 2003. – Forschungsbericht

[Bungartz u. a. 2002] Bungartz, Hans-Joachim ; Griebel, Miachael ; Zenger, Christoph: *Einführung in die Computergraphik.* Vieweg&Teubner, 2002. – ISBN 3528167696

[Castanheira 2003] Castanheira, Danilo Balby S.: *Geometria construtiva de sólidos combinada à representação b-rep*, Universidade de Brasilia, Diplomarbeit, 2003

[Chang 2009] Chang, Han-Shuo: *Color Schemes on 4D Models in 4D Construction Management Tools*, National Taiwan University, Diplomarbeit, 2009

[Chen 1996] Chen, Jianer: Computational Geometry: Methods and Applications / Computer Science Department Texas A&M University. 1996. – Forschungsbericht

[Chen 2008] Chen, Po-Han: Integration of cost and schedule using extensive matrix method and spreadsheets. In: *Automation in Construction* 18 (2008), S. 32–41

[Christiansson u. a. 2002] Christiansson, Per ; Dawood, Nashwan ; Svidt, Kjeld: Virtual Buildings and Tools to Manage Construction Process Operations. In: *CIB w78 Konferenz*, 2002

[Cooper 1998] Cooper, James W.: *The Design Patterns - Java Companion*. Addison-Wesley, 1998 `http://www.patterndepot.com/put/8/DesignJava.PDF`

[Copeland 2007] Copeland, Tom: *Generating Parsers with JavaCC*. Centennial Books, 2007. – ISBN 0976221438

[Dawood u. a. 2005] Dawood ; Scott ; Sriprasert ; Mallasi: The virtual construction site (VIRCON) tools: An industrial evaluation. In: *ITCON* 10 (2005), S. 43–54

[Dawood u. a. 2003] Dawood, Nashwan ; Sriprasert, Eknarin ; Mallasi, Zaki ; Hobbs, Brian: Development of an integrated information resource base for 4D/VR construction processes simulation. In: *Automation in Construction* 12 (2003), Nr. 12, S. 123–131

[Eastman 1999] Eastman, C. M.: *Building Product Models: Computer Environments Supporting Design and Construction*. Boca Raton, Florida USA : CRC Press, 1999. – ISBN 0849302595

[Eastman u. a. 2008] Eastman, Chuck ; Teicholz, Paul ; Sacks, Rafael ; Liston, Kathleen: *BIM Handbook - A Guide to Building Infromation Modeling for Owners, Managers, Designers, Engineers, and Contractors*. John Wiley & Sons, Inc., 2008. – ISBN 9780470185285

[Egenhofer u. Franzosa 1991] Egenhofer, M. ; Franzosa, R.: Point-Set Topological Spatial Relations. In: *International Journal of Geographical Information Systems* 5 (1991), S. 161–174

[Ericson 2005] Ericson, Christer ; Eberly, David H. (Hrsg.): *Real-Time Collision Detection*. Morgan Kaufmann (Elsevier), 2005

[Ester u. a. 2008] Ester, Alexandra ; Fitzner, Josephine ; Holzheu, Anica ; Koschorrek, Sabine: Modellbasierte Terminplanung / Bauhaus-Universität Weimar, Professur Betriebswirtschafslehre im Bauwesen. 2008. – Forschungsbericht

[Fijneman u. a. 2008] Fijneman, Martin ; Matthyssen, Arne ; Verhofstad, Frank ; Tulke, Jan: Process Mapping for Key Process 4 - Construction Scheduling / InPro-Project. 2008. – Forschungsbericht

[Firmenich 2002] Firmenich, Berthold: *CAD im Bauplanungsprozess: Verteilte Bearbeitung einer strukturierten Menge von Objektversionen*, Bauhaus-Universität Weimar, Diss., 2002

[Foley u. a. 1995] Foley ; Dam van ; Feiner ; Hughes: *Computer Graphics*. Addison-Wesley, 1995. – ISBN 0201848406

[Froese 2003] Froese, Thomas: Future Directions for IFC-based Interoperability. In: *ITcon* 8 (2003), S. 231–246

[Froese u. a. 1999] Froese, Thomas ; Fischer, Martin ; Grobler, Francois ; Ritzenthaler, John ; Yu, Kevin: Industriy Foundation Classes for Project Management - A Trial Implementation. In: *ITcon* 4 (1999), S. 17–36

[Ghosh 2007] Ghosh, Subir K.: *Visibility Algorithms in the Plane*. Cambridge University Press, 2007. – ISBN 9780521875745

[Goldstein 2001] Goldstein, Harry: 4D: Science Fiction or Virtual Reality? / McGraw Hill Construciton. 2001. – Forschungsbericht

[GPM 2003] GPM, Gesellschaft für Projektmanagement: *Projektmanagement Fachmann, Band 2*. RKW-Verlag, 2003. – ISBN 3926984570

[GSA 2008] GSA, U.S. General Services Administration: *Statement of intention to support Building Information Models with open standards*. `http://www.gsa.gov/Portal/gsa/ep/programView.do?pageTypeId=17109&ooid=20917&programPage=%2Fep%2Fprogram%2FgsaDocument.jsp&programId=14501&channelId=-24291`. Version: 2008

[Hall u. a. 1991] Hall, Keith ; Wiederhold, Gio ; Law, Kincho: A Framework form Change Management in a Design Database / CIFE, Stanford University. 1991. – Forschungsbericht

[Hanff 2003] Hanff, Jochen: *Abhängigkeiten zwischen Objekten in ingenieurwissenschaftlichen Anwendungen*. Shaker Verlag, 2003. – ISBN 383222280

[Heesom u. Mahdjoubi 2004] Heesom, D. ; Mahdjoubi, L.: Trends of 4D CAD applications for construction planning. In: *Construction Management and Economics* 22 (2004), Nr. February, 22, S. 171–182

[Heesom 2006] Heesom, David: Interactive Generation of "Multi-Level of Detail"4D CAD Simulations. In: *6th International Conference on Construction Applications of Virtual Reality*, 2006

[Hinze 2004] Hinze, Jimmie W.: *Construction Planning and Scheduling*. Pearson Education, 2004. – ISBN 0130928615

[Hofstadler 2007] Hofstadler, Christian: *Bauablaufplanung und Logistik im Baubetrieb*. Springer Verlag, 2007. – ISBN 3540343202

[Hubbard 1990] Hubbard, Philip M.: Constructive Solid Geometry for Triangulated Polyhedra / Brown University, Department of Computer Science. 1990 (CS-90-07). – Forschungsbericht

[Issa u. a. 2003] Issa, Raja R. A. ; Flood, Ian ; O'Brien, William J.: *4D CAD and Visualization in Construction: Developments and Applications*. Routledge, 2003. – ISBN 9789058093547

[Jaeger u.a. 2007] Jaeger, J. ; Leuthner, B. ; Markwardt, A. ; Siegler, S. ; Sulzer, C.: Comparative Analysis of Best Practice / InPro project. Version: 2007. http://www.inpro-project.eu/publications.asp. 2007. – Forschungsbericht

[Jan Reinhardt 2004] Jan Reinhardt, James Garrett Burcu A.: SiDaCoS: Product and Process Models on Construction Sites. 2004. – Forschungsbericht

[Jongeling 2006] Jongeling, Rogier: *A Process Model for Work-Flow Management in Construction*, Luleå University of Technology, Diss., 2006

[Jongeling u.a. 2005] Jongeling, Rogier ; Kim, Jonghoon ; Mourgues, Claudio ; Fischer, Martin ; Olofsson, Thomas: Quantitative analyses using 4D models - an explorative study. In: *International Confereence on Construction Engineering and Management (ICCEM)*, 2005, S. 830–835

[Jongeling u. Olofsson 2007] Jongeling, Rogier ; Olofsson, Thomas: A method for planning of work-flow by combined use of location-based scheduling and 4D CAD. In: *Automation in Construction* 16-2 (2007), S. 189–198

[Kang u.a. 2005] Kang, Leen-Seok ; Paulson, Boyd C. ; Hak Kim, Joong-Min K.: Business Breakdown Structure for Construction Management and Web-based Application System. In: *ITCON* 10 (2005), S. 169–191

[Kankainen u. Seppänen 2003] Kankainen, Jouko ; Seppänen, Olli: A Line-of-Balance based scheduling planning and control system. In: *11. Annual conference on Lean Construction*. Virginia, USA, 2003

[Karstila u. Serén 2005] Karstila, K. ; Serén, K.: IFC Aspect Card Library / Ergebnisbericht des Pro-IT Projektes. Version: 2005. http://virtual.vtt.fi/virtual/proj6/proit/julkiset_tulokset/proit_aspectcardlibrary_v1_1_2005_01_27.pdf. 2005. – Forschungsbericht

[Karstila u. Serén 2004] Karstila, Kari ; Serén, Kalle: Implementation Guidelines for IFC 2x2 Concrete Domain / Confederation of Finnish Construction Industries and Betoni. Version: 2004. http://www.betoni.com/files/files/IFC_Concrete_ImplementationGuideLine_V1.pdf. 2004. – Forschungsbericht

[Kerzner 2009] Kerzner, Harold: *Project Mangement*. Wiley, 2009. – ISBN 9780470278703

[Kähkönen u. Leinonen 2003] Kähkönen, Kalle ; Leinonen, Jarkko: Visual product chronology as a solution for accessing building product model data. In: *CIB w78 Conference*, 2003

[Kline u.a. 2004] Kline, Kevin E. ; Kline, Daniel ; Hunt, Brand: *SQL in a nutshell*. O'Reilly, 2004. – ISBN 1565927443

[König 2006] König, Markus: Workflow-Management in der Baupraxis. In: *Tag des Baubetriebs Bauhaus-Universität Weimar*, 2006

[Koch 2009] Koch, Christian: *Bauwerksmodellierung im kooperativen Planungsprozess: Mit der Objektorientierung zur Verarbeitungsorientierung*, Bauhaus-Universität Weimar, Diss., 2009

[Koche 2006] Koche, Brian: Revolutionary Scheduling: The Practical Application of 4D Technology form a CM Perspective. In: *CMAA Spring Leadership Forum*, 2006

[Kochendörfer u. a. 2007] Kochendörfer, Bernd ; Liebchen, Jens H. ; Viering, Markus G.: *Bau-Projekt-Management: Grundlagen und Vorgehensweise*. Teubner Verlag, 2007. – ISBN 3835100114

[Koo u. Fischer 2003] Koo, Bonsang ; Fischer, Martin: Formalizing Construction Sequencing Constraints for Rapid Generation of Schedule Alternatives / CIFE, Stanford University. Version: 2003. `http://cife.stanford.edu/online.publications/WP075.pdf`. CIFE, 2003. – Forschungsbericht

[Koschorrek u. a. 2008] Koschorrek, Sabine ; Ester, Alexandra ; Tauscher, Eike ; Tulke, Jan: IFC-basiertes Modell für die 4D-Bauablaufanimation. In: *20. Forum Bauinformatik, Dresden*, 2008

[Lai u. Kang 2009] Lai, Kuan-Chen ; Kang, Shih-Chung: Collision detection strategies for virtual construction simulation. In: *Automation in Construction* 18 (2009), S. 724–736

[Laidlaw u. a. 1986] Laidlaw, David H. ; Trumbore, W. B. ; Hughes, John F.: Constructive Solid Geometry for Polyhedral Objects. In: *ACM SIGGRAPH Computer Graphics* 20(4) (1986), S. 161–170

[Langetepe u. a. 2009] Langetepe, Elmar ; Penninger, Rainer ; Tulke, Jan: Computing the visibility area between two polygons in linear time / Rheinische Friedrich-Wilhelms-Universität Bonn, Institut für Informatik I. 2009. – Forschungsbericht

[Lee u. Zlatanova 2009] Lee, J. ; Zlatanova, S.: *3D Geo-Information Science*. Springer, 2009. – ISBN 9783540873945

[Leen-Seok u. a. 2004] Leen-Seok, K. ; Yong-Su, L. ; Jung-Min, K.: 4D-Modellsystem; Funktionsanalyse für die Bauplanung. In: *Computer Spezial -Software für Architekten Ingenieure Bauunternehmen* Feb. (2004), S. 13–15

[Liebich u. a. 2004] Liebich, Thomas ; Forester, James ; Karstila, Kari ; Wix, Jeff: IFC 2x Edition 2 Model Implementation Guide / International Alliance for Interoperability Modeling Support Group. Version: 2004. `http://www.buildingsmart.de/4/4_01.htm`. 2004. – Forschungsbericht

[Liebich u. a. 2008] Liebich, Thomas ; Weise, Matthias ; Böhms, Michel ; Bonsma, Peter ; Fies, Bruno ; Bourdeau, Marc ; Dunwell, Steve ; Pratt, Jeff ; Babic, Nenad C. ; Tibaut, Andrej ; Nour, Mohamed: Open Standards for Interoperability between Applications in Early Design / InPro project. Version: 2008. `http://www.inpro-project.eu/publications.asp`. 2008. – Forschungsbericht

[Liebich u. a. 2009] Liebich, Thomas ; Weise, Matthias ; Laine, Tuomas ; Jokela, Markku ; Tulke, Jan ; Böhms, Michel ; Bonsma, Peter ; Nummelin, Olli ; Houttu, Juha-Matti: InPro Building Information Model / InPro-Project. Version: 2009. www.inpro-project.eu. 2009. – Forschungsbericht

[Marsch u. Fritze 1995] Marsch, Jürgen ; Fritze, Jörg: *SQL: Eine praxisorientierte Einführung*. 3. Vieweg, 1995. – ISBN 3528252103

[Mechnig 1997a] Mechnig, Michael: Grundzüge der Terminplanung und die Bedeutung des Austauschs von Terminplanungsdaten, Teil 1. In: *Baumarkt* 11 (1997), November, S. 43–46

[Mechnig 1997b] Mechnig, Michael: Grundzüge der Terminplanung und die Bedeutung des Austauschs von Terminplanungsdaten, Teil 2. In: *Baumarkt* 12 (1997), Dezember, S. 49–51

[Mengenermittlung 2007a] Mengenermittlung, IAI Arbeitskreis M.: Definition der Basismengen / buildingSMART, IAI - Industrieallianz für Interoperabilität e.V. 2007. – Forschungsbericht

[Mengenermittlung 2007b] Mengenermittlung, IAI Arbeitskreis M.: Mengenwerte in IFC Austauschdatei / buildingSMART, IAI - Industrieallianz für Interoperabilität e.V. 2007. – Forschungsbericht

[Metsker u. Wake 2006] Metsker, Steven J. ; Wake, William C.: *Design Patterns in Java*. 2006. – ISBN 0321333020

[Mikulakova u. a. 2008] Mikulakova, E. ; König, M. ; Tauscher, E. ; Beucke, K.:: Case-Based Reasoning for Construction Tasks. In: *XIIth International Conference on Computing in Civil and Building Engineering (ICCCBE), Beijing, China*, 2008

[Möller 1997] Möller, Tomas: A Fast Triangle-Triangle Intersection Test. In: *Journal of Graphics Tools* 2 (1997), S. 25–30

[Märki u. a. 2007] Märki, Fabian ; Fischer, Martin ; Kunz, John ; Haymaker, John: Decision Making for Scheduling Optimization / CIFE, Stanford University. 2007. – Forschungsbericht

[Nigudkar 2005] Nigudkar, Narendra S.: *Effectiveness of 4D construction modeling in detecting time-space conflicts on construction sites*, Texas A&M University, Diplomarbeit, 2005

[Nummelin u. a. 2007] Nummelin, O. ; Böhms, M. ; Verhofstad, F. ; Bourdeau, M. ; Pfitzner, M. ; Neuberg, F.: Requirements for an Open ICT Platform / InPro project. Version: 2007. http://www.inpro-project.eu/publications.asp. 2007. – Forschungsbericht

[Oracle 2007] Oracle: *Oracle Database 11g Workspace Manager Overview*. White Paper. 2007

[O'Rourke 1998] O'Rourke, Joseph.: *Computational Geometry in C.* Cambridge University Press, 1998

[Pahl u. Damrath 2000] Pahl, Peter J. ; Damrath, Rudolf: *Mathematische Grundlagen der Ingenieurinformatik.* Springer Verlag, 2000. – ISBN 3–540–60501–0

[Porkka u. Kähkönen 2007] Porkka, Janne ; Kähkönen, Kalle: Software development approaches and challenges of 4D product models. In: *24th CIB w78 Conference*, 2007 (CIB W78 Conference)

[Preparata u. Shamos 1985] Preparata, Franco P. ; Shamos, Michael I. ; Gries, David (Hrsg.): *Computational Geometry an Introduction.* Springer-Verlag, 1985

[Pries 2007] Pries, August: IFC als Standardformat des Building Information Modells (BIM) - Planungs- und baubegleitende Datenerfassung / CAD-Stelle Bayern. Version: 2007. `http://www.buildingsmart.de/pdf/ps_2007-02.pdf`. 2007. – Forschungsbericht

[Qu u.a. 2005] Qu, Honggang ; Pan, Mao ; Wang, Bin ; Wang, Yong ; Wang, Zhangang: Boolean operations of triangulated solids and their applications in the 3D geological modelling. In: *International Symposium on Spatio-temporal Modeling, Spatial Reasoning, Analysis, Data Mining and Data Fusion*, 2005

[Rebolj u.a. 2008] Rebolj, Danijel ; Babic, Nenad us ; Magdic, Ales ; Podbreznik, Peter ; Psunder, Mirko: Automated construction activity monitoring system. In: *Advanced Engineering Informatics* 22-4 (2008), S. 493–503

[Riedel u. Wender 2007] Riedel, Thomas ; Wender, Katrin: Realisierung von Anfragefunktionalität für einen dynamischen Partialmodellverbund. In: *Forum Bauinformatik*, 2007

[Rönneblad u. Olofsson 2003] Rönneblad, Anders ; Olofsson, Thomas: Application of IFC in Design and Production of Precast Concrete Constructions. In: *ITcon* 8 (2003), S. 167–180

[Robert 2008] Robert, Christian R.: *Entwicklung eines Bauzeitennachtrages und Untersuchung der Möglichkeit der verbesserten Durchsetzbarkeit durch den Einsatz von 4D-Visualisierungswerkzeugen*, Fachhochschule Kaiserslautern, Diplomarbeit, 2008

[Roman 2002] Roman, Steven: *Access Database - Design & Programming.* O'Reilly, 2002. – ISBN 9780596002732

[Rösch u. Volkmann 1994] Rösch, W. ; Volkmann, W.: *Bauprojektmanagment - Terminplanung mit System für Architekten und Ingenieure.* Rudolf-Müller-Verlag, 1994. – ISBN 3481006926

[Rueppel u. Klauer 2004] Rueppel, Uwe ; Klauer, Thomas: Internet-based Workflow-Management for Civil engineering Projects. In: *International Conference on Computing in Civil and Building Engineering (ICCCBE)*, 2004

[Russell u. a. 2009] Russell, Alan ; Staub-French, Sheryl ; Tran, Ngoc ; Wong, William: Visualizing high-rise building construction strategies using linear scheduling and 4D CAD. In: *Automation in Construction* 18 (2009), S. 219–236

[Sari 2006] Sari, Rina: Unteruchung der Anwendung ArchiCAD-TeamWork Funktion und Change Manager von Graphisoft in einem Bauplanungsprozess / Bauhaus-Universität Weimar. 2006. – Forschungsbericht

[Schelle u. a. 2006] Schelle, Heinz ; Ottmann, Roland ; Pfeiffer, Astrid: *Project Manager*. German Association for Project Management (GPM), 2006. – ISBN 3924841306

[Schexnayder u. Mayo 2004] Schexnayder, Clifford J. ; Mayo, Rechard E.: *Construction Management Fundamentals*. McGraw Hill Construction, 2004. – ISBN 0072818778

[Schraufstetter u. Borrmann 2007] Schraufstetter, S. ; Borrmann, A.: AABB-Bäume als Grundlage effizeinter Algorithmen einer räumlichen Anfragesprache. In: Merkel, A. P. (Hrsg.) ; Schütz, R. (Hrsg.) ; Wiessflecker (Hrsg.): *Forum Bauinformatik*. Graz : Verlag der Technischen Universität Graz, 2007, S. 145–159

[Seren u. Karstila 2001] Seren, Kalle ; Karstila, Kari: MS Project - IFC Mapping Specification / VTT & Eurostep. 2001. – Forschungsbericht

[Sheppard 2004] Sheppard, Laurel M.: Virtual Building for Construction Projects. In: *IEEE Computer Graphics and Applications* (2004), Nr. January/February, S. 6–12

[Soini u. a. 2004] Soini, Mika ; Leskelä, Ilkka ; Seppänen, Olli: Implementation of Line-of-Balance based scheduling and project contol system in a large construction company. In: *12. Annual conference on Lean Construction*. Elsinore, Danemark, 2004

[Staub-French 2004] Staub-French: Feature-based Product Modeling for Building Construction. In: *10. International Conference on Computing in Civil and Building Engineering (ICCBE)*. Weimar, 2004

[Staub-French 2000] Staub-French, Sheryl: Practical and Research Issues in using Industry Foundation Classes for Construction Cost Estimating / CIFE, Stanford University. 2000. – Forschungsbericht

[Staub-French u. Fischer 2001] Staub-French, Sheryl ; Fischer, Martin: Industrial Case Study of Electronic Design, Cost & Schedule Integration / CIFE, Stanford University. 2001. – Forschungsbericht

[Staub-French u. a. 2002a] Staub-French, Sheryl ; Fischer, Martin ; Kunz, John ; Paulson, Boyd ; Ishii, Kos: A Formal Process to Create Resource-Loaded & Cost-Loaded Activities Related to Feature-Based Product Models / CIFE, Stanford University. 2002. – Forschungsbericht

[Staub-French u. a. 2002b] Staub-French, Sheryl ; Fischer, Martin ; Kunz, John ; Paulson, Boyd ; Ishii, Kos: An Ontology for Relating features of Building Product Models with Construction Activities to Support Cost Estimating / CIFE, Stanford University. 2002. – Forschungsbericht

[Staub-French u. a. 2008] Staub-French, Sheryl ; Russell, Alan ; Tran, Ngoc: Linear Scheduling and 4D Visualisation. In: *Journal of Computing in Civil Engineering* 22 (2008), S. 192–205

[Stoiber 2007] Stoiber, Franz: *Analyse der Verwendbarkeit von IFC 2x3 als produktneutrales Austauschformat der österreichischen Hochbaurichtlinien*, Technische Universität Wien, Diplomarbeit, 2007

[Succar 2009] Succar, Bilal: Building information modelling framework: A research and delivery foundation for industry stakeholders. In: *Automation in Construction* 18 (2009), S. 357–375

[Tantisevi u. Akinci 2009] Tantisevi, Kevin ; Akinci, Burcu: Transformation of a 4D product and process model to generate motion of mobile cranes. In: *Automation in Construction* 18 (2009), S. 458–468

[Tanyer u. Aouad 2005] Tanyer, Ali M. ; Aouad, Ghassan: Moving beyond the fourth dimension with an IFC-based single project database. In: *Automation in Construction* 14 (2005), S. 15–32

[Tauscher u. a. 2009] Tauscher, E. ; Mikulakova, E. ; Beucke, K. ; König, M.: Automated Generation of Construction Schedules based on the IFC Object Model. In: *ASCE International Workshop on Computing in Civil Engineering, Austin, Texas, USA*, 2009

[Theiler 2008] Theiler, Michael: *Interaktives Ermitteln von Bauteilinformationen aus STEP-Dateien*, Bauhaus-Universität Weimar, Professur Informatik im Bauwesen, Diplomarbeit, 2008

[Theiler 2009] Theiler, Michael: *IFC-Import-Funktion für Cademia*, Bauhaus-Universität Weimar, Professur Informatik im Bauwesen, Diplomarbeit, 2009

[Theiler u. a. 2009] Theiler, Michael ; Tauscher, Eike ; Tulke, Jan ; Riedel, Thomas: Visualisierung von IFC-Objekten mittels Java3D. In: *21. Forum Bauinformatik, Karlsruhe*, 2009

[Thulasiraman u. Swamy 1992] Thulasiraman, K. ; Swamy, M. N. S.: *Graphs: Theory and algorthms.* Jozhn Wiley & Sons, 1992. – ISBN 9780471513568

[Tsai u. a. 2008a] Tsai, M. H. ; Kang, S. C. ; Hsieh3, S. H.: Developing the Workflow for Implementing a 4D Construction Management Tool in a Construction Firm. In: *Eleventh East Asia-Pacific Conference on Structural Engineering & Construction (EASEC-11) "Building a Sustainable Environment"*, 2008

[Tsai u. a. 2008b] Tsai, Menghan ; Kang, Shihchung ; Hsieh, Shanghsien ; Kuo, Chenghan ; Pei, Tengjau ; Yeh, Kaichen: Experiences on Study of Setting-up a4D Construction Management Tool in High-rise Construction Project. In: *Proceedings of 12th International Conference on Computing in Civil and Building Engineering (ICCCBE08)*, 2008

[Tulke u. Hanff 2007] Tulke, Jan ; Hanff, Jochen: 4D Construction Sequence Planning - New Process and Data Model. In: *24th cib W78 Conference "Bringing ITC knowledge to work"*, 2007, 79-84

[Tulke u. a. 2008] Tulke, Jan ; Nour, Mohamed ; Beucke, Karl: Decomposition of BIM objects for scheduling and 4D simulation. In: *ECPPM - eWork and eBusiness in Architecture, Engineering and Construction* (2008)

[Uher 2003] Uher, Ghomas E.: *Programming and Scheduling Techniques*. UNSW Press, 2003. – ISBN 0868407259

[Vad 2007] Vad, Janos S.: *Textbasierte Objektversionierung von Planungsdaten für die Weiterverarbeitung in Ingenieruanwendungen*, Bauhaus-Universität Weimar, Diplomarbeit, 2007

[Vanecek 1990] Vanecek, George: Brep-Index: A Multi-Dimensional Space Partitioning Tree / Computer Sciences Department, Purdue University. 1990 (CSD-TR-1051). – Forschungsbericht

[Vanecek u. a. 1992] Vanecek, George ; Nau, Dana S. ; Karinthi, Raghu R.: Performance Analysis of a Polyhedral Boolean Set Operations Algorithm / Computer Science Department Purdue University. 1992. – Forschungsbericht

[Viescas 1989] Viescas, John: *SQL: Die relationale Datenbanksprache*. Vieweg, 1989. – ISBN 3528047356

[de Vries u. Broekmaat 2007] Vries, B. de ; Broekmaat, M.: Generation of a construction planning from a 3D CAD model. In: *Automation in Construction* 16 (2007), S. 13–18

[Vygen u. a. 2002] Vygen, Klaus ; Schubert, Eberhard ; Lang, Andreas: *Bauverzögerung und Leistungsänderung*. Werner Verlag, 2002. – ISBN 3804138632

[Waly u. Thabet 2002] Waly, Amed F. ; Thabet, Walid Y.: A Virtual Construction Environment for preconstruction planning. In: *Automation in Construction* 12 (2002), S. 139–154

[Wang u. a. 2007] Wang, H. ; Akinci, B. ; J., Garett: A Formalism for Detecting Version Differences in Data Models. In: *Journal of Computing in Civil Engineering* 21 No.5 (2007), S. 321–330

[Weise u. a. 2009] Weise, M. ; Liebich, T. ; Tulke, J. ; Bonsma, P.: Discussion paper: IFC support for model-based scheduling / Public specification of the NMP EU project InPro (IP 026716-2). 2009. – Forschungsbericht

[Weise 2006] Weise, Matthias: *Ein Ansatz zur Abbildung von Änderungen in der modell-basierten Objektplanung.* Technische Universität Dresden, 2006. – ISBN 978–3–86005–557–1

[Weiss 2005] Weiss, Timo: Theoretische und praktische Untersuchung 3 dimensionaler Analysefunktionalität im Kontext dynamischer Modellverwaltungssysteme: Diplomarbeit / Bauhaus-Universität Weimar, Fakultät Bauingenieurwesen, Professur Informations- und Wissensverarbeitung. 2005. – Forschungsbericht

[Wender 2009] Wender, Katrin: *Das virtuelle Bauwerk als Informationsumgebung für die Planung im Bestand: Zur Organisation und Strukturierung einer digitalen Bauwerksakte*, Bauhaus-Universität Weimar, Diss., 2009

[Winch u. Kelsey 2005] Winch, Graham M. ; Kelsey, John: What do construction project planners do? In: *International Journal of Project Management* 23 (2005), S. 141–149

[Winstanley u. Hoshi 1992] Winstanley, Graham ; Hoshi, Kunito: Model-based Planning Utilizing Activity Aggregation Based on Zones / CIFE, Stanford University. 1992. – Forschungsbericht

[Young u. a. 2007] Young, Norbert W. ; Jones, Stephen A. ; Bernstein, Harvey M.: Interoperability in the Construction Industry. In: *SmartMarket Report.* McGraw Hill Construction, 2007

[Young u. a. 2008] Young, Norbert W. ; Jones, Stephen A. ; Bernstein, Harvey M.: Building Information Modeling (BIM). In: *SmartMarket Report.* McGraw Hill Construction, 2008

[Zamzow 2008] Zamzow, Rouven: *Verwendung der Baufortschrittssimulation für die Bewertung beanspruchter Raumstrukturen im Bauprozess*, Universität Duisburg-Essen, Diss., 2008

[Zhou u. a. 2009] Zhou, Wei ; Heesom, David ; Georgakis, Panagiotis ; Nwagboso, Cristopher ; Feng, Adam: An interactive approach to collaborative 4D construction planning. In: *ITCON* 14 (2009), S. 30–47

Glossar

4D-Simulation

Der Begriff 4D-*Simulation* hat sich in der Literatur für die Animation des geometrischen Bauwerksmodells zum Zweck der Visualisierung des geplanten Bauablaufs etabliert. Objekte des CAD-Modells werden zunächst mit Vorgängen im Terminplan verknüpft. Während des Ablaufs der 4D-Simulation über den betrachteten Projektzeitraum werden aktive Vorgänge durch eine Änderung der Darstellung der verknüpften CAD-Objekte visualisiert. Dieses erfolgt heutzutage üblicherweise durch das Ein- bzw. Ausblenden sowie durch Ändern der Farbe der CAD-Objekte am Start- bzw. Endzeitpunkt der aktiven Vorgänge (Änderung zu diskreten Zeitpunkten). Erweiterte Darstellungsmöglichkeiten wie die Bewegung oder das Schneiden von Objekten sowie eine kontinuierliche Änderung der Darstellung wären denkbar.

Basismengen

Um nachfolgenden Planungsprozessen einen leichteren Zugriff auf Mengeninformationen auch ohne Berechnung zu ermöglichen, werden von Entwurfs- bzw. CAD-Programmen in der Regel quantifizierende Werte in Form von Objekteigenschaften (Attributen) zur Verfügung gestellt. Diese werden als Basismengen bezeichnet. In IFC werden für Modellobjekte die Anzahl sowie Längen-, Flächen-, Volumen-, Gewichts- und Zeitwerte als Eigenschaften unterstützt. Üblicherweise werden von CAD-Programmen die Hauptgeometrieabmessungen sowie daraus unmittelbar abgeleitete Flächen und Volumendaten übermittelt. Nationale Berechnungsvorschriften finden dabei keine Berücksichtigung. Naturgemäß stehen Basismengen nur für explizit modellierte Objekte zur Verfügung.

BIM

Diese Abkürzung wird in der Literatur in zwei verschiedenen Kontexten verwendet. Ursprünglich als Abkürzung für „Building Information Model“ bzw. im Deutschen für „Bauwerksinformationsmodell“ eingeführt, bezeichnet sie die Gesamtheit der strukturierten Daten, die zur virtuellen Abbildung eines Bauprojektes im Computer verwendet werden. Neuerdings wird die Abkürzung jedoch auch für „Building Information Modeling“ verwendet und beschreibt damit den Prozess der gemeinsamen Erzeugung und Nutzung eines digitalen Gebäudemodells als Grundlage für die Planung.
Im Rahmen dieser Arbeit wird versucht, den Kontext der Verwendung stets klarzustellen.

BPMN

Die „Business Process Modeling Notation“[72,73] ist eine graphische Spezifikationssprache zum Zweck der allgemein verständlichen Beschreibung von Geschäftsprozessen. Sie erlaubt neben der Darstellung verbundener Einzelvorgänge auch deren Zuordnung zu Akteuren sowie die Erfassung des Informationsaustauschs.
Die Firmen BIZAGI[74] und TIBCO[75] bieten frei verfügbare Softwareprodukte zur Modellierung von Geschäftsprozessen auf Basis von BPMN.

CAD

„Computer Aided Design“ bezeichnet die elektronische Erstellung von Konstruktionsplänen. Hiermit ist bis heute in der Regel ein zeichnungsorientiertes Vorgehen gemeint, bei dem zweidimensionale Linien und nicht dreidimensionale Objekte gezeichnet werden. Die Unterscheidung zwischen 3D-CAD und BIM ist unscharf, jedoch fokussiert der Begriff CAD auf die Erfassung der Geometrie. Im Unterschied zu BIM können weitere Bauteileigenschaften und Zusatzinformationen lediglich in informeller Textform erfasst werden. Die Semantik dieser Informationen geht dabei verloren, was eine Weiterverarbeitung erschwert bzw. verhindert.

Clash Detection

Hierbei handelt es sich um eine geometrische Konfliktortung auf der Basis eines dreidimensionalen Computermodells. Die Geometrien der geprüften Bauteile werden automatisch auf gegenseitige Durchdringung geprüft. Wird eine Durchdringung gefunden, so besteht ein Konflikt. Der Konflikt kann entweder in einem Planungs- oder einem Modellierfehler begründet sein. Die so gefundenen Konflikte werden nach weiterer Verarbeitung (Filtern und Sortieren) in einem Report zusammengestellt und als Grundlage für Koordinierungsbesprechungen verwendet. Die Treffsicherheit und Effizienz der automatischen Konfliktortung hängt maßgeblich von der Qualität und Struktur der verwendeten Modelldaten ab (siehe hierzu auch [Benning u. a. 2009]). Durch die automatisierte Ortung von Fehlern werden jedoch eine Arbeitserleichterung und höhere Zuverlässgkeit bei der Fehlerkontrolle erreicht.

CPM

Die „Critical Path Method“ ist eine Methode der Netzplantechnik (siehe auch DIN69900-1), bei der der Terminplan als mathematischer Graph, bestehend aus Knoten (Start- und Endereignisse) und Kanten (Vorgänge), abgebildet wird. Auf Basis der logischen Verknüpfung der Vorgänge und ihrer Dauer werden für jeden Vorgang die frühesten und spätesten Ausführungszeitpunkte sowie die daraus resultierenden Pufferzeiten berechnet. Eine Kette von Vorgängen ohne Pufferzeiten, die Projektstart und -ende verknüpft, wird als kritischer Weg bezeichnet, da Verzögerungen in diesen Vorgängen unmittelbar zur Verlängerung der Gesamtprojektdauer führen.

[72] www.bpmn.org
[73] http://de.wikipedia.org/wiki/Business_Process_Modeling_Notation
[74] www.bizagi.com
[75] www.tibco.com

FEM

Die „Finite Element Method“ ist ein numerisches Näherungsverfahren zur Lösung von partiellen Differentialgleichungen. Sie wird im Bauwesen zur Simulation des physikalischen Verhaltens von Bauteilen und des Baugrunds verwendet. Neben dem Haupteinsatzbereich in der Statik zur Berechnung von Schnittkräften und Verformungen wird sie auch in anderen Bereichen, wie z.B. der Berechnung des Wärmetransports, genutzt.

Gewerk

Mit Gewerk wird eine Fachdisziplin (Handwerksberuf) innerhalb der Bauausführung bezeichnet.

IFC

Die „Industry Foundation Classes“ sind ein objektorientiertes Datenmodell zum Datenaustausch im Bauwesen (siehe auch Kapitel 3.2).

Leistungsverzeichnis (LV)

Die im Rahmen eines Bauprojektes zu erbringenden Leistungen werden bzgl. Menge und Qualität im Leistungsverzeichnis beschrieben. Dieses ist Grundlage für die Kostenermittlung und den Preisvergleich zwischen mehreren Leistungsanbietern. Eine Gliederung der einzelnen Teilleistungen erfolgt hierarchisch gemäß der Projektstruktur (z.B. Los, Abschnitt) und darunter thematisch nach Leistungsarten. Eine Addition der Teilleistungen erfolgt über die Kosten, denen verschiedenartige Mengendimensionen zugrunde liegen können. Zum Austausch von LV-Daten wird in Deutschland derzeit vorwiegend das GAEB[76]-Datenformat verwendet.

LoB

„Line of Balance“ ist eine ursprünglich für die Planung von Linienbauwerken (Straßen, Tunneln, usw.) entwickelte, ortsbezogene Terminplanungsmethode, bei der Vorgänge in einem Weg-Zeit-Diagramm dargestellt werden. Heutzutage findet diese Methode jedoch auch im Hochbau Anwendung, wobei die Wegachse die Mengen der in den einzelnen Geschossen, Mietbereichen oder Wohnungen auszuführenden Bauarbeiten repräsentiert. Grundlage dieser Methode ist die Annahme einer gleichbleibenden Produktivität bei der Ausführung eines Terminplanprozesses. Um Störungen im Bauablauf zu vermeiden, ist der Einsatz von Ressourcen und damit die Produktivität so zu optimieren, dass sich im Weg-Zeit-Diagramm parallel verlaufende Geraden ergeben. Weitere Informationen findet man in [Kankainen u. Seppänen 2003], [Soini u. a. 2004] und über LoB in Verbindung mit 4D in [Jongeling 2006], [Staub-French u. a. 2008] und [Russell u. a. 2009].

LV-Mengen

Im Leistungsverzeichnis werden alle für die Berechnung der Kosten einer Bauleistung erforderlichen Mengen aufgeführt. Diese Mengen berücksichtigen nationale Berechnungsvorschriften und können neben den direkt aus dem Modell berechneten auch zusätzliche Mengen für nicht modellierte Objekte enthalten. Die Qualität

[76] www.gaeb.de

und Vollständigkeit ist von LV-Mengen gegenüber Basismengen aufgrund ihrer Bedeutung für die Kostenermittlung höher.

OQL

Die „Object Query Language“ ist eine Anfragesprache für Objektdatenbanken. Sie ist stark an SQL angelehnt.

PSP-Code

Der „Projektstrukturplan-Code“ ist eine kapitelähnliche Nummerierung, die zur eindeutigen Bezeichnung jedes Elements innerhalb eines Terminplans verwendet wird. Er erlaubt es, jeden Vorgang eindeutig in die Struktur des Terminplans einzuordnen.

ROI

Der Begriff „Return On Investment“ beschreibt das Verhältnis aus Gewinn zu eingesetztem Kapital und ist somit eine Messgröße für den finanziellen Erfolg eines Unternehmens oder einer Einzelinvestition.

Semantik

Unter Semantik versteht man die Bedeutung, die verwendete Zeichen (Begriffe, Symbole, Gesten) für Teilnehmer (Personen oder Softwareprogramme) einer Kommunikation haben.

SQL

Die „Structured Query Language“ ist eine Sprache zur Eingabe, Abfrage und Änderung von Daten in relationalen Datenbanken. SQL ist von ANSI und ISO standardisiert und wird von allen gängigen relationalen Datenbanksystemen unterstützt.

Topologie

Die Topologie ist ein Teilgebiet der Mathematik, die sich mit Eigenschaften von Objekten beschäftigt, die bei umkehrbar eindeutigen, stetigen Abbildungen unverändert bleiben. Hiervon sind im Rahmen der Bauwerksmodellierung insbesondere die Beziehungen und räumlichen Anordnungen von Objekten untereinander (ohne Betrachtung metrischer Aspekte) von Interesse. Damit bildet die Topologie neben der Geometrie und Semantik eine der drei Säulen der Datenmodellierung.

Workflow

Der Begriff Workflow bezeichnet die Steuerung der Reihenfolge von Teilarbeitsprozessen. In der Planung von Bauwerken wird hierdurch sichergestellt, dass die Planungsbeteiligten in einer definierten Reihenfolge auf Daten zugreifen bzw. diese erzeugen oder validieren. Die Steuerung erfolgt durch ein Benachrichtigungssystem sowie die Verwaltung von Statusinformationen für die im Prozess befindlichen Daten.

Workspace

Mit Workspace wird im Rahmen dieser Arbeit eine lokale Datenhaltung bezeichnet, die alle für die Aufgaben eines Fachplaners relevanten Daten enthält. Der Fachplaner hat uneingeschränkten Zugriff auf die im Workspace enthaltenen Daten. Die Weitergabe von Daten an andere Projektpartner erfolgt durch eine explizite Veröffentlichung auf einem zentralen Projektdatenserver oder alternativ per Datei (siehe auch Kapitel 4.1).

XML

Die „Extensible Markup Language“ ist eine allgemeine Beschreibungssprache zum Austausch hierarchisch strukturierter Daten in Textdateien. Durch die Einschränkung der abbildbaren Daten mittels eines Schemas können daraus anwendungsbezogene Datenbeschreibungssprachen erstellt werden.