Ingenieurfakultät Bau Geo Umwelt
Lehrstuhl für Computergestützte Modellierung und Simulation
Prof. Dr.-Ing. André Borrmann

Entwicklung einer Methode zum Festhalten des Standes von geprüften Bauwerksmodellen.

Kilian Siebenhütter

Masterthesis
für den Master of Science Studiengang Bauingenieurwesen

Autor:	Kilian Siebenhütter
Matrikelnummer:	
Betreuer:	Prof. Dr.-Ing. André Borrmann
	Sebastian Esser, M.Sc.
Ausgabedatum:	15. Dezember 2019
Abgabedatum:	15. Juni 2020

Abstract

The use of Building Information Modeling (BIM) in the construction industry has brought about many changes for companies and authorities. In addition to new concepts and working methods for handling models, new applications and systems are also needed to meet the requirements and make working methods more effective. The new applications and concepts serve, among other things, to replace analog and manual working methods with digital variants.

A similar approach is implemented in this thesis. The aim is to develop a new concept of how the state of a model can be recorded, similar to a building plan with an approval stamp. The functions of the traditional version must be retained. It should be possible to record the state of an inspected structure and confirm this with a signature. The „digital plan stamp“ is not applied to a simple plan, but should record the most important information of a building model at the time of the inspection. In order to be able to retrieve this state afterwards, a kind of „template“ of the model will be created. With the help of this template it is possible to „lay“ the state of the check over a modified variant of the model, in order to note thus recognizable deviations, e.g. component changes, between the two versions. Furthermore, it is to be investigated how the template can be used for a partial check of models or a check of a special subject.

An implementation of the method of the digital plan stamp shall explain this concept in more detail. The template is generated within the created Windows application, using the building model in Industry Foundation Classes (IFC) format and the graph database *Neo4j*. The data integrity of the template and the relevant model data is guaranteed by a digital signature. With the implementation, different application scenarios from the building construction sector can be demonstrated. The main field of application envisaged is the inspection of structures, either by the inspection structural engineer or for approval by the building authorities.

Zusammenfassung

Der Einsatz von Building Information Modeling (BIM) in der Baubranche brachte viele Veränderungen für Unternehmen und Behörden mit sich. Neben neuen Konzepten und Arbeitsmethoden für den Umgang mit Modellen werden auch neue Anwendungen und Systeme benötigt um den Anforderungen gerecht zu werden bzw. die Arbeitsweise effektiver zu gestalten. Die neuen Anwendungen und Konzepte dienen unter anderem dazu, die analogen und manuellen Arbeitsmethoden durch digitale Varianten abzulösen.
Ein ähnliches Vorgehen wird in dieser Arbeit umgesetzt. Es soll ein neues Konzept entwickelt werden, wie die digitale Version des traditionellen Planstempel aussehen könnte. Dabei müssen die Funktionen der traditionellen Version beibehalten werden. Wie bei einem Planstempel soll es also möglich sein den Stand eines geprüften Bauwerks festzuhalten und dies durch eine Signatur zu bestätigen. Der „digitale Planstempel“ wird dabei nicht auf einen einfachen Plan angewendet, sondern soll die wichtigsten Informationen eines Bauwerkmodells zum Zeitpunkt der Prüfung festhalten. Damit dieser Stand auch im Nachhinein abgerufen werden kann wird eine Art „Schablone“ des Modells angefertigt. Mithilfe dieser Schablone ist es möglich den Stand der Prüfung über eine veränderte Variante des Modells zu „legen“, um somit Abweichungen, wie z.B. Bauteiländerungen, zwischen den beiden Versionen erkenntlich zu machen. Außerdem soll untersucht werden, wie die Schablone bei einer Teilprüfung von Modellen oder einer Fachprüfung eingesetzt werden kann.
Eine beispielhafte Implementierung der Methode des digitalen Planstempels soll dieses Konzept genauer erläutern. Die Schablone wird dabei innerhalb der erstellten Windows-Anwendung, mithilfe des Gebäudemodells in Industry Foundation Classes (IFC)-Format und der Graphdatenbank *Neo4j*, erzeugt. Die Datenintegrität der Schablone und der relevanten Modelldaten wird durch eine digitale Signatur gewährleistet. Mit der Implementierung lassen sich verschiedene Einsatzszenarien aus dem Hochbau aufzeigen. Das angedachte Haupteinsatzgebiet ist dabei die Prüfung von Bauwerken, sei es durch den Prüfstatiker oder bei der Genehmigung durch die Baubehörde.

Inhaltsverzeichnis

Abkürzungsverzeichnis

2D Zweidimensional
3D Dreidimensional
AES Advanced Encryption Standard
BauVorlV Bauvorlagenverordnung
BayBO Bayerische Bauordnung
BIM Building Information Modeling
CAD Computer Aided Design
DES Data Encryption Standard
DSA Digital Signature Algorithm
EQL Express Query Language
FEM Finite-Elemente-Methode
GUID Globally Unique Identifier
HOAI Honorarordnung für Architekten und Ingenieure
IDE Integrated Development Environment
IFC Industry Foundation Classes
JSON JavaScript Object Notation
LOD Level of Development
LOG Level of Geometry
LOI Level of Information
LPH Leistungsphase
MD5 Message-Digest Algorithm 5
MVD Model View Definition
OWL Web Ontology Language
POW Proof of Work
QL4BIM Query Language for Building Information Models
RDF Resource Description Framework
SDAI Standard Data Access Interface
SPF STEP Physical File
SQL Structured Query Language
STEP Standard for the Exchange of Product model data
WPF Windows Presentation Foundation
XML Extensible Markup Language

Kapitel 1

Einleitung

1.1 Einführung in das Thema und Motivation

Das Bauwerksmodell findet immer mehr Einzug in die deutschen bzw. internationalen Bauunternehmen. Grund hierfür ist die Building Information Modeling (BIM)-Methode. BIM ist aktuell eines der, wenn nicht sogar das größte Thema in der Baubranche. Von Jahr zu Jahr steigt die Anzahl der Firmen, welche BIM als Arbeitsmethode verwenden (NBS, 2019). Hierbei wird die Methode, je nach Unternehmen, unterschiedlich eingesetzt. Die wenigsten Firmen verwenden BIM für jedes Projekt. Laut NBS (2019) haben nur 15% der befragten Personen die BIM-Methode in jedem Projekt eingesetzt. Betrachtet man die Abbildung 1.1, erkennt man jedoch, dass ca. 60% der Befragten die Methode bei jedem zweiten Projekt verwenden. Diese Information und die weitere Zunahme der Einführungen von BIM lässt vermuten, dass diese Arbeitsweise in Zukunft eine noch größere Rolle im Baubereich spielen wird.

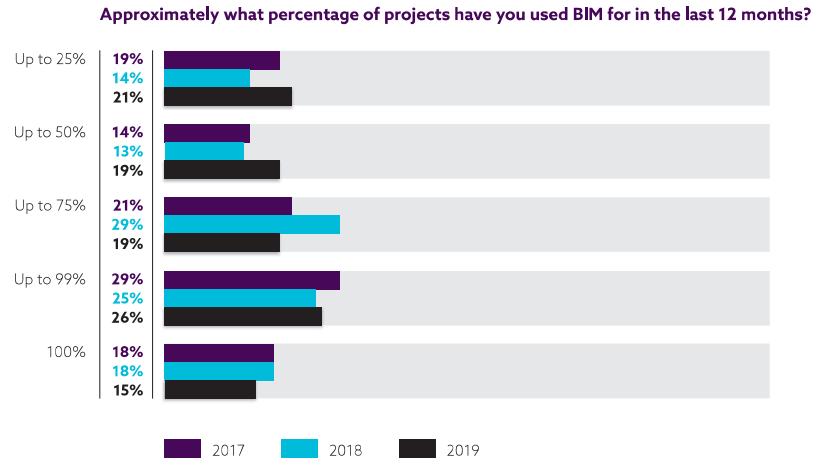

Abbildung 1.1: Umfrage im Vereinigten Königreich: Die Befragten haben die Verwendung von BIM für Projekte in Prozent angegeben. (NBS, 2019, S.18)

Durch die Einführung dieser Methode werden viele Arbeitsschritte vereinfacht oder fallen komplett weg. Somit können mithilfe des erstellten Bauwerkmodells Information ausgelesen oder bestimmte Abläufe automatisiert werden. Die Arbeit der Beteiligten kann durch so ein Modell erleichtert werden, die Erstellung und Pflege des Modells schafft jedoch neue Aufgaben und Probleme. Allein das Anfertigen des Modells stellt einen großen Arbeitsaufwand dar. Das Modell wird über die verschiedenen Planungsphasen aufgebaut und wird somit, je weiter das Projekt fortschreitet, komplexer. Zum Start wird das Gebäude als Volumenmodell erstellt. Diesem werden Grobelemente, wie Wände und Decken, zugeordnet, welche im nächsten Schritt durch Fenster oder Trennwände erweitert werden. Die Bauteile werden mit den übergeordneten Elementen verknüpft und im Laufe der Planung von vielen verschiedenen Fachplanern mit Informationen gefüllt (Sommer, 2016). Dabei nimmt der Detaillierungs- und Informationsgrad von Bauteilen über den Verlauf der Planung immer weiter zu (Baunetz_Wissen, 2020b). Damit die Menge an Informationen auch wirklich verwendet werden kann und die Übersichtlichkeit durch diese nicht verloren geht, sind neue Arbeitsweisen und Programme besonders wichtig.

Viele Funktionen können bereits durch die unterschiedlichsten Programme durchgeführt werden. So sind Kosten- und Mengenermittlungen möglich, sowie Termin- und Tragwerksplanungen. Zusätzlich lässt sich mit bestimmten Programmen ein „Modell-Check“ durchführen. Dabei wird das Modell unter anderem auf Doppelungen und Kollisionen überprüft (Albrecht, 2015). Die Arbeitsweise bei der Erstellung von Modellen verändert sich durch den Einsatz dieser Programme nur gering. Jedoch können mit diesen Anwendungen Fehler zu einem frühen Zeitpunkt entdeckt und ausgemerzt werden, bevor weitreichende Probleme entstehen. Durch den Einsatz dieser Anwendungen kann das Leben der Planer teilweise vereinfacht werden und die Chance auf größere Probleme, wie z.B. eine Kostenüberschreitungen, lässt sich somit reduzieren.

Ähnlich motiviert ist auch die Methode, die in dieser Arbeit erläutert wird. Den Planern und Prüfern soll ein Werkzeug an die Hand gegeben werden, mit dem der Planungs- und Prüfprozess teilweise vereinfacht werden kann. Zusätzlich soll damit eine traditionelle Arbeitsweise an die heutigen Möglichkeiten der Technik angepasst werden. Im weiteren Verlauf wird diese Methode öfters auch als „digitaler Planstempel“ bezeichnet.

1.2 Ziel der Arbeit

Vorliegende Arbeit dient zur Erforschung der weiteren Digitalisierung des Planungs- und Prüfprozesses von Bauwerken und dadurch die mögliche Vereinfachung dieses Ablaufs. Dabei wird die Lebenszyklusphase „Planung“ genau unter die Lupe genommen und die Veränderungen, welche bei der Einführung von BIM entstehen, betrachtet. Aufgrund des Wandels von Plänen zu Modellen werden neuartige Methoden für bereits vorhandene Ar-

beitsmethoden benötigt. In dieser Arbeit soll eine solche Methode entwickelt werden. Diese Methode soll ein Schritt in die Richtung der vollkommenen Digitalen Prüfung und Genehmigung von Bauwerken sein und die damit verbundene Verringerung der Wege zwischen Planern, Prüfern, der Gemeinde und der Baubehörde.

Um diese Methode umzusetzen wird ein Programm entwickelt, welches Beteiligten der Planung und Prüfingenieuren bei der Arbeit helfen soll. Genauer gesagt soll diese Anwendung ähnlich wie ein Planstempel funktionieren. Dabei soll bei einer Prüfung der aktuelle Stand eines Modells signiert und abgespeichert werden. Es wird somit eine Art „Schablone“ erstellt. Zusätzlich soll es möglich sein diesen Stand nachträglich abzurufen und mit der aktuellen Version des Modells zu vergleichen um etwaige Unterschiede zu ermitteln. Die Methode könnte an mehreren wichtigen Stellen des Planungsprozesses zum Einsatz kommen. Beispielsweise bei der Prüfung des Modells durch einen Experten. Dabei sollte eine Unterscheidung der Fachrichtung des Prüfers möglich sein, damit z.B. der Prüfstatiker das Modell nach tragenden Bauteilen filtern kann. Zusätzlich kann die Methode bei der Erteilung einer Baugenehmigung durch die Baubehörde eingesetzt werden.

1.3 Struktur der Arbeit

Die Arbeit ist in acht Kapitel unterteilt. Nach der Einleitung, in welcher Thema und Motivation dargestellt werden, findet in den darauffolgenden Kapiteln eine Vermittlung der Grundlagen statt. In Kapitel 2 wird auf den Planungsvorgang eines Gebäudes eingegangen. Gestartet wird dieses Kapitel mit einer kurzen Beschreibung des Lebenszyklus eines Bauwerks und einer genaueren Unterteilung der Planungsphase. Fortgesetzt wird dies dann mit einer ausführlicheren Betrachtung der Planung. Dabei werden die einzelnen Leistungsphasen beschrieben, welche laut Honorarordnung für Architekten und Ingenieure (HOAI) der Planung zugeordnet werden können. Das darauffolgende Unterkapitel widmet sich der Arbeitsweise mit BIM. Hierbei wird ein allgemeiner Überblick über die Funktionsweise, die verschiedenen Varianten und die Probleme dieser Methode geliefert. Zum Abschluss des Kapitels 2 wird darauf eingegangen, wie der „Digitale Planstempel“ eingesetzt werden kann und welche Vorteile dadurch entstehen.

Kapitel 3 betrachtet die notwendigen technischen Eckpfeiler zur Erstellung und zur Filterung der Informationen eines Modells. Unter anderem wird die verwendete Software zur Gebäudemodellierung beschrieben. Damit die Daten aus diesen Programmen weiterverwendet werden können, erfolgt eine Betrachtung der verschiedenen Austauschformate. Zum Abschluss des Kapitels werden unterschiedliche Möglichkeiten untersucht, um die Arbeit mit dem Austauschformat Industry Foundation Classes (IFC) zu vereinfachen.

Der nächste Abschnitt, Kapitel 4, befasst sich allgemein mit den Herzstücken der Methode. Nämlich mit der Validierung von kryptografischen Verfahren zum Erstellen der Schablone und dem gewährleisten der Datenintegrität. Der Stand des Modells soll mithilfe von Hash-Werten in der Schablone vermerkt werden. Für die Gewährleistung der Datenintegrität werden zwei verschiedene Varianten betrachtet. Eine Speicherung der Schablone in einer Blockchain, also einer dezentralen Datenbank, oder die Verwendung einer digitalen Signatur. Im Kapitel der beiden Optionen wird zuerst ein grober Überblick über die Funktionsweise dieser Techniken geliefert und im Anschluss werden die verschiedenen Arten davon dargestellt.

Die Erläuterung zur Umsetzung der Methode befindet sich im Kapitel 5. Genauer gesagt werden alle notwendigen Vorgänge der Methode aufgezeigt und es wird ein tiefer Einblick in die technische Umsetzung des gesamten Ablaufs des „Digitalen Planstempels“ geliefert. Dabei wird der Umgang mit den Techniken, welche bereits im Kapitel 4 erwähnt wurden, genau beschrieben.

Im Kapitel 6 werden mithilfe des erstellten Programms zwei verschiedene Einsatzszenarien durchgespielt. Dabei werden die erhaltenen Ergebnisse aus der Anwendung dargestellt. Anhand der Szenarien, die bereits im Abschnitt 2.4.1 erläutert wurden, wird schrittweise der Ablauf und die Funktionen der Methode aufgezeigt.

Im vorletzten Kapitel 7 soll ein grober Überblick über den Forschungsstand und mögliche alternativen zur Umsetzung der Methode des digitalen Planstempels geliefert werden. Dabei wird unter anderem die Thematik eines „BIM-basierten Bauantrags“ betrachtet, sowie alternative Formate für Gebäudemodelle. Den Abschluss des Kapitels bildet die Betrachtung von Möglichkeiten um Veränderungen an Modellen bzw. Graphen zu erkennen.

Zusätzlich soll zum Abschluss dieser Arbeit, auf Grundlage der geschaffenen Erkenntnisse, ein Fazit gezogen und einen Ausblick auf die Zukunft gewehrt werden. Dabei sollen Möglichkeiten zur Verbesserung, sowie eine Einschätzung der weiteren Verwendbarkeit dieser Methode erläutert werden.

Kapitel 2

Planungsphase

2.1 Allgemein

Betrachtet man den kompletten Lebenszyklus eines Gebäudes wird klar, dass die Planung nur eine Phase von vielen Lebenszyklusphasen ist. Um genauer zu sein gibt es fünf verschiedene Phasen, diese werden wiederum in einzelne Hauptprozesse unterteilt (Helmus *et al.*, 2017). Abbildung 2.1 stellt diese Gliederung der Prozesse und Phasen dar. Für diese Arbeit ist jedoch nur der Abschnitt „Planung“ wichtig.

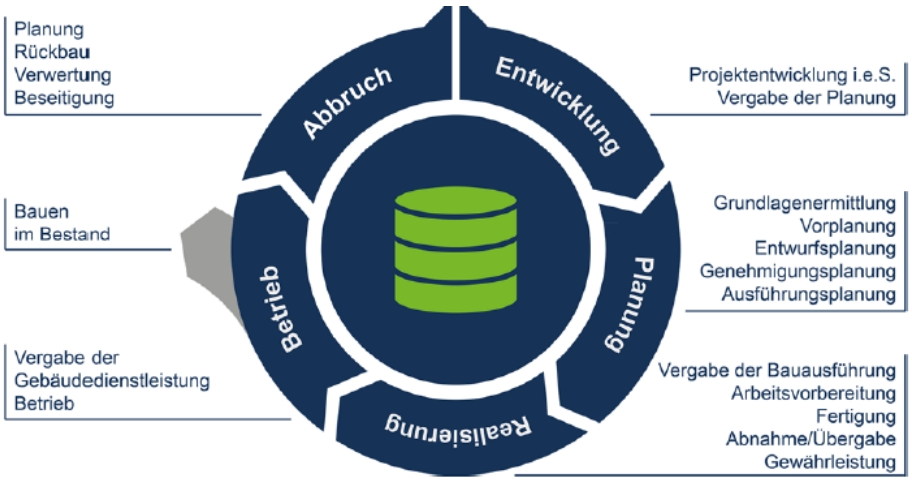

Abbildung 2.1: Lebenszyklusphasen eines Gebäudes. (Helmus *et al.*, 2017, S.6)

Möchte man nun genauer auf die Hauptprozesse eingehen, sollte man sich zuerst klar werden was ein Prozess eigentlich ist. Bergwanger *et al.* (2019) definiert diesen so:

> *„Unter Prozess versteht man die Gesamtheit aufeinander einwirkender Vorgänge innerhalb eines Systems. So werden mittels Prozessen Materialien, Energien oder*

auch Informationen zu neuen Formen transformiert, gespeichert oder aber allererst transportiert.“ (Bergwanger *et al.*, 2019)

Diese Definition beschreibt die Planungsphase ganz gut. Auch hier werden in den einzelnen Hauptprozessen neue Informationen geschaffen und bereits vorhandene Information weitergegeben und ergänzt. Wie genau die Hauptprozesse der Planung definiert sind und wie dort die Transformation der Informationen aussieht, wird im nächsten Kapitel beschrieben.

2.2 Ablauf der Planungsphase

Wie bereits im vorhergegangenen Absatz erwähnt und durch die Abbildung 2.1 grafisch dargestellt, wird die Planungsphase in mehrere Hauptprozesse, auch Leistungsphasen genannt, unterteilt. Laut der HOAI gibt es neun Leistungsphasen (§34 Abs. 3 HOAI). Wobei nur Leistungsphase 1 bis 5 der Planung zugeordnet werden kann (Helmus *et al.*, 2017).

- Grundlagenermittlung
- Vorplanung
- Entwurfsplanung
- Genehmigungsplanung
- Ausführungsplanung

In den folgenden Unterabschnitten werden die einzelnen Leistungsphasen genauer beschrieben. Die Aufgaben, die bei den Leistungsphasen der Planung anfallen, können je nach Art der Bebauung abweichen. So wird zwischen den Leistungsbildern „Gebäude und Innenräume, „Freianlagen, „Ingenieurbauwerke“ usw. unterschieden (Anlage 10 bis 15 HOAI). In diesem Fall wird nur auf das Leistungsbild „Gebäude und Innenräume“ eingegangen.

2.2.1 Grundlagenermittlung

Die Grundlagenermittlung ist die erste Leistungsphase (Hauptprozess) der Planung und somit ein wichtiger Bestandteil des gesamten Lebenszyklus. Die Tätigkeiten, die in dieser Phase zu leisten sind, lassen sich in Grundleistungen und besondere Leistungen aufteilen. Bei den Grundleistungen müssen Architekten und/oder Ingenieure den Auftraggeber bei der Erstellung der Aufgabenstellung oder der Bedarfsplanung unterstützen. Außerdem sollten sie bei der Entscheidung, über die möglichen Fachplaner, Hilfe leisten. Damit ein Übergang in die nächste Leistungsphase leichter fällt, ist eine gewissenhafte Zusammenfassung und Dokumentation der gewonnen Erkenntnisse und Ergebnisse besonders wichtig. Unter besondere

Leistungen fallen viele verschiedene Aufgaben. Dazu zählen unter anderem die Leistungen, „Bedarfsplanung“, „Standortanalyse“, „Machbarkeitsstudie“ usw. (Anlage 10 HOAI).

Zusammenfassend lässt sich sagen, dass diese Phase zur allgemeinen Betrachtung der Bebauung vorgesehen ist. Dabei wird der grundsätzliche Ablauf und Umfang des Projektes bestimmt und eine mögliche Machbarkeit getestet. Um diese Projekt dann auf den richtigen Weg zu leiten werden weitere fachliche Beteiligte engagiert.

2.2.2 Vorplanung

Die Leistungsphase (LPH) 2 beinhaltet die Vorplanung des Gebäudes. Zu den Leistungen dieses Abschnittes zählt unter anderem die Grundlagenanalyse, aus welcher dann die nötigen Aufgaben ermittelt werden. Zusätzlich wird ein erster Entwurf der Bebauung für das Planungskonzept angefertigt. Eine volle Ausarbeitung findet hierbei nicht statt. Zur Übersicht werden mehrere grobe Skizzen erstellt. Mithilfe dieser werden die verschiedenen Entwurfsansätze bewertet und ein Konzept ausgewählt. Besonders wichtig für diese Phase ist die Überprüfung der grundsätzlichen Genehmigungsfähigkeit. Somit können mögliche Probleme früh erkannt werden. Zum Abschluss sollte der Entwerfende eine erste Kostenschätzung, auf Grundlage der im Planungskonzept definierten Gebäudekennwerte, erstellen (Franke *et al.*, 2002).

2.2.3 Entwurfsplanung

Im Anschluss zur Vorplanung kommt die LPH 3, auch Entwurfsplanung genannt. Hierbei werden die Überlegungen zu dem grundsätzlichen Planungskonzept aus LPH 2 weitergeführt. Laut Franke *et al.* (2002) wird diese Phase wie folgt beschrieben:

> *„In der Entwurfsplanung (LP 3) wird die grundsätzliche Lösung der Bauaufgabe erarbeitet. Dabei müssen zahlreiche rechtliche, räumliche oder auch finanzielle Randbedingungen berücksichtigt werden.“* (Franke *et al.*, 2002, S.50)

Je nachdem um welche Art von Bebauung es sich handelt, verschiebt sich die Gewichtung der einzelnen Randbedingungen. So wird bei einem Bürogebäude besonders viel Wert auf die Optimierung der vermietbaren Fläche geachtet (Franke *et al.*, 2002). Eine ausgeglichene Betrachtung der verschiedenen Bedingungen gibt es meistens nicht. Zusätzlich werden Aufgaben, die in der Phase zuvor schon begonnen wurden, weiter geführt oder genauer ausgearbeitet. Gemeint sind z.B. die Zeichnungen des Gebäudes, der Terminplan, die Kostenberechnung und die Genehmigungsverhandlungen (Anlage 10 HOAI).

2.2.4 Genehmigungsplanung

Der folgende Abschnitt beschäftigt sich mit der Genehmigungsplanung. Hierbei werden die notwendigen Planungsunterlagen sowie die Anträge für das Baugenehmigungsverfahren erstellt. Um einen kompletten Überblick über die Bebauung zu geben, fließen in den Bauantrag zusätzlich die Unterlagen der Fachplaner (z.B. Wärmeschutznachweis) mit ein. Der entstandene Bauantrag muss für die Genehmigung bei der zuständigen Bauaufsichtsbehörde eingereicht werden (Franke *et al.*, 2002).

Besonders in dieser Leistungsphase könnte der digitale Planstempel die Arbeit der Planer und Prüfer vereinfachen. Wie genau diese Methode helfen und eingesetzt werden kann, wird im Kapitel 2.4 erläutert.

2.2.5 Ausführungsplanung

Sobald die Genehmigung der Behörde erteilt wurde bzw. ein genehmigungsfähiger Entwurf vorhanden ist, startet die letzte Leistungsphase der Planung. Dabei handelt es sich um die Ausführungsplanung. In dieser Phase werden die bereits angefangenen Aufgaben erweitert und verbessert. Somit werden für die spätere Ausführung Konstruktions- und Detailpläne im Maßstab 1:50 bis 1:1 erstellt. Zusätzlich müssen Terminplan und Ausführungsplanung fortgeschrieben werden (Anlage 10 HOAI).

Viele Aufgaben die in dieser Leistungsphase zu erledigen sind, werden für die nächste Phase „Vorbereitung der Vergabe“ benötigt. Beispielsweise können die genau angefertigten Zeichnungen bei der Erstellung einer Mengenermittlung verwendet werden. Die Vergabe (LPH 6) gehört jedoch nicht mehr zur Lebenszyklusphase „Planung“ sonder schon zur „Realisierung“.

2.3 Planung mithilfe von Building Information Modeling (BIM)

Im Gegensatz zu anderen Industriebereichen konnte in der Baubranche, trotz fortschreitender Digitalisierung, keine große Steigerung der Arbeitsqualität gemessen werden. Laut Albrecht (2015) entstehen trotz effektiverer Arbeitsweise, durch schnellere Kommunikationsmittel und Computer Aided Design (CAD)-Programme, weiterhin die gleichen Probleme bei der Abwicklung eines Bauprojekts.

„

- *Kostenüberschreitungen*
- *Terminverzug*

- *Ungenügende Kommunikation zwischen den Projektbeteiligten*
- *Qualitätsprobleme*

" (Albrecht, 2015, S.13)

Auch eine ständige Verbesserung der Baustoffe und Maschinen bringt nicht unbedingt die Lösung zu diesen Problemen. Um diesen Komplikationen Herr zu werden, muss man vor allem den Bauprozess genauer betrachten und mögliche Problemstellen in dieser Arbeitskette ausmerzen. Eine Prozessoptimierung kann z.B. durch den Einsatz von BIM erreicht werden (Albrecht, 2015). Wie genau dies möglich ist und um was es sich eigentlich bei BIM handelt, wird in den folgenden Abschnitten erläutert.

2.3.1 Einführung in BIM

Für BIM gibt es viele verschiedene Definitionen. Das Bundesministerium für Verkehr und digitale Infrastruktur (2015) definiert es wie folgt.

> *„Building Information Modeling bezeichnet eine kooperative Arbeitsmethodik, mit der auf der Grundlage digitaler Modelle eines Bauwerks die für seinen Lebenszyklus relevanten Informationen und Daten konsistent erfasst, verwaltet und in einer transparenten Kommunikation zwischen den Beteiligten ausgetauscht oder für die weitere Bearbeitung übergeben werden."* (Bundesministerium für Verkehr und digitale Infrastruktur, 2015, S.4)

BIM ist somit keine Software, wie oft gedacht wird. Es handelt sich um eine Methode, welche mithilfe von Computerprogrammen, bei der Planung, Ausführung und Betreibung von Gebäuden verwendet werden kann (Softtech, 2020). Um genauer zu sein, wird ein dreidimensionales Gebäudemodell mithilfe spezieller Software erstellt. Je nach Planungsphase und Anwendungsfall sollte das Modell mit wichtigen Informationen ergänzt werden. Zusätzlich müssen die bereits eingebrachten Daten überprüft werden, ob diese noch relevant sind. Somit sollte das Modell am Ende alle wesentlichen Informationen des Gebäudes beinhalten, damit die Projektbeteiligten jederzeit auf diese zugreifen können (Bundesministerium für Verkehr und digitale Infrastruktur, 2015). Damit verkürzen sich die Entscheidungszeiträume bei der Planung deutlich. Durch BIM lassen sich viele der in Kapitel 2.3 genannten Probleme, bei der Abwicklung eines Bauprojekts, verbessern. Durch das genaue Erfassen der Bauteile lassen sich die Baukosten und Mengenermittlungen exakt aus dem Modell entnehmen. Somit hat der Bauherr eine hohe Kostentransparenz, wodurch die Chance auf versteckte Kosten und Nachträge stark sinkt. Damit auch das Problem „Terminverzug" angegangen werden kann, ist es möglich eine Terminplanung in das Modell zu integrieren und die Baustelle, mithilfe der gesamten anderen Daten, zu simulieren (Albrecht, 2015).

Zusätzlich lässt sich, durch den Einsatz von BIM, die Arbeit der Fachplaner vereinfachen. Tragwerksplaner können, mithilfe der richtigen Software und den Informationen aus dem zentralen Datenmodell, Berechnungen durchführen und automatisierte Schal- und Bewehrungspläne erstellen. So kann z.B. das komplette Dreidimensional (3D)-Modell exportiert und damit eine Finite-Elemente-Methode (FEM)-Berechnung erstellt werden. Außerdem ist eine Bemessung von Einzelbauteilen, durch das Exportieren von Zweidimensional (2D)-Teilmodellen, möglich (Mensch und Maschine Deutschland GmbH, 2019). Ein weiterer Fachbereich, bei dem BIM Vorteile bringen könnte, ist die Bauphysik. Durch die vielen Informationen, die das Modell und somit die einzelnen Bauteile beinhalten, ist es unter anderem möglich den Wärmebedarf automatisch zu ermitteln (Albrecht, 2015).

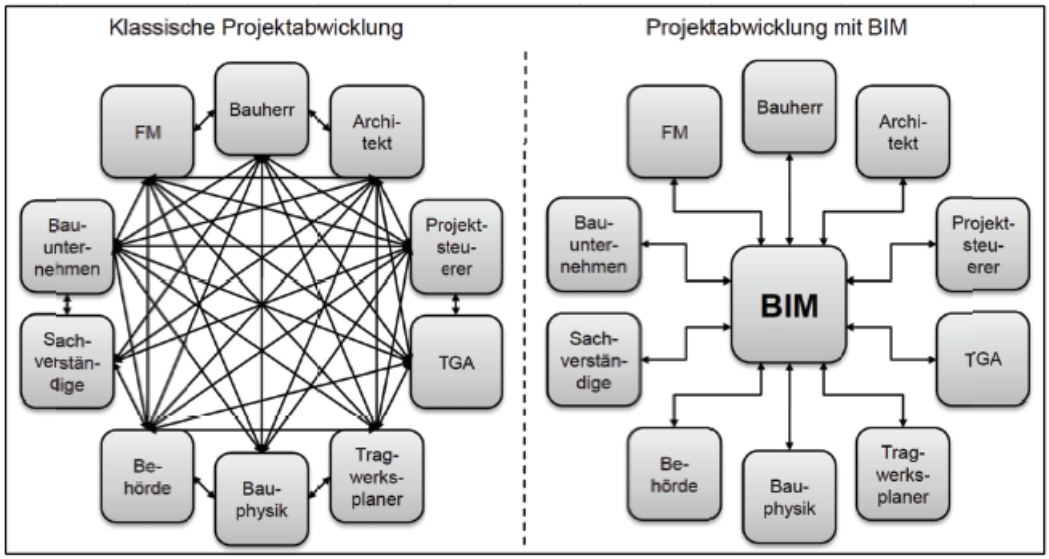

Abbildung 2.2: Projektabwicklungsarten. (Albrecht, 2015, S.21)

Abbildung 2.2 zeigt grafisch die Struktur der Projektabwicklung im klassischen Sinne und der Projektabwicklung mithilfe von BIM. Die Variante mit BIM ist wie folgt strukturiert. Das Datenmodell ist der zentrale Knoten und alle anderen Beteiligten sind mit diesem vernetzt. Somit können sich alle Personen an diesem Datenmodell beteiligen und daraus für sich wichtige Informationen herausfinden oder ihre Erkenntnisse einbringen. Bei Zusammenarbeit zwischen verschiedenen Beteiligten des Projekts läuft die Verständigung über dieses zentrale Modell ab (Albrecht, 2015). Sollten hieraus Änderungen entstehen, können diese sofort in das Datenmodell übernommen werden und stehen somit jeder anderen Person zur Verfügung. Betrachtet man hingegen die klassische Variante der Projektabwicklung, ist so ein Vorgehen nicht möglich. Änderungen die zwischen Architekten und Tragwerksplanern getroffen werden, müssen an jeden anderen Beteiligten einzeln weitergegeben werden. Wodurch Daten verloren gehen könnten und Personen möglicherweise mit veralteten Versionen arbeiten.

Der oben beschriebene Arbeitsvorgang, mit einem zentralen Datenmodell, zeigt die optimale Nutzung von BIM. Viele der positiven Eigenschaften sind aber auch bei der Verwendung auf

kleinerer Ebene, also nur innerhalb der eigenen Firma, zu beobachten. Was es für Abstufungen bei der Verwendung von BIM gibt und wie die unterschiedlichen Varianten eingesetzt werden können, wird im nächsten Kapitel genauer beschrieben.

2.3.2 BIM-Varianten

Es gibt verschiedene Varianten wie BIM bei einem Bauprojekt verwendet werden kann. Dabei wird zwischen den Methoden „little BIM“ und „big BIM“ unterschieden. Bei der ersten Variante „little BIM“ wird der Prozess nur innerhalb der eigenen Firma verwendet. Somit werden nur die Informationen in das erstellte Modell eingetragen, die für das jeweilige Büro von Nutzen sind. Albrecht (2015) liefert folgendes Beispiel:

> *„Ein Ingenieurbüro benötigt beispielsweise zur Berechnung der Statik eines Gebäudes nicht die Oberflächenbeschaffenheit des Fußbodens, die jedoch für den Facility Manager wesentlich sind.“* (Albrecht, 2015, S.22)

Man spricht auch von einer Insellösung (Albrecht, 2015). Für die zwei oben genannten Kategorien gibt es noch jeweils zwei Unterscheidungen. So ist von einer „little closed BIM“-Lösung die Rede, wenn eine einheitliche Software verwendet wird und kein Austausch der Daten mit anderen beteiligten Firmen stattfindet. Im Gegensatz dazu bedeutet „little open BIM“, dass Projektdaten mit anderen Beteiligten, die nicht die gleiche Software verwenden, ausgetauscht werden. Dies erfolgt z.B. über das Austauschformat Industry Foundation Classes (IFC) (N+P Informationssysteme GmbH, 2018). Auf dieses Format wird im Kapitel 3.2.1 noch genauer eingegangen.

Im Vergleich zur Insellösung, wird bei der „big BIM“-Lösung die Methode von mehreren, am besten allen, Beteiligten der Planung eingesetzt. Erfolgt die Bearbeitung mit einem einheitlichen Softwareumfeld, ist von „big closed BIM“ die Rede. Sollten die Softwarepakete unterschiedlich sein und ein Datenaustausch mithilfe von Austauschformaten wie IFC erfolgen, spricht man von „big open BIM“ (N+P Informationssysteme GmbH, 2018).

2.3.3 Probleme mit BIM

So gut die Arbeitsweise mit BIM klingt, gibt es immer noch Risiken und Probleme, weswegen viele Unternehmen Projekte weiterhin ohne diese Methode bearbeiten. Ein großes Risiko, welches bei der Verwendung von BIM entsteht, ist die Bestimmung der Eigentumsrechte an dem Datenmodell. Der Bauherr, der für das Projekt bezahlt, könnte sich als Eigentümer dieser Daten sehen. Die Projektbeteiligten, dessen Ideen und Arbeit in das Projekt fließen, müssen jedoch auch eigentumsrechtlich geschützt werden. Dadurch könnte es zu vielen Konflikten kommen, weswegen der Besitzanspruch der Daten vertraglich geklärt werden muss. Eine

einheitliche Festlegung des Eigentums an den Daten ist dabei selten möglich und sollte für jedes Projekt speziell beantwortet werden (Azhar, 2011).

Das womöglich größte Problem beim Arbeiten mit BIM ist, die Klärung der Verantwortlichkeit. Wer kontrolliert die Eingabe von Daten und ist der Schuldige bei möglichen Ungenauigkeiten. Außerdem ist die Frage, wer am Ende die Verantwortung trägt, falls durch Fehler Geldschäden entstehen oder Personen verletzt werden (Azhar, 2011).

Neben den oben genannten rechtlichen bzw. vertraglichen Problemen gibt es noch weitere Hürden, aufgrund dessen manche Unternehmen die BIM-Arbeitsweise noch nicht bei ihren Projekten verwendet. Dabei wird unter anderem zwischen technischen Hürden, preislichen Hürden und Nutzerproblemen unterschieden. Unter den technischen Hürden fallen folgenden Probleme. Damit die Methode optimal genutzt werden kann, sollte die Software auf den aktuellsten Stand upgedatet werden und eine leistungsfähige Hardware vorhanden sein. Sofern die Variante „open BIM“ verwendet wird, also verschiedene Softwareprogramme im Einsatz sind, muss der verlustfreie Datenaustausch zwischen den Parteien funktionieren (Albrecht, 2015). Hierzu kann das Austauschformat IFC verwendet werden. Der Austausch kann bei diesem Standard jedoch immer wieder mal problematisch sein, aufgrund der unterschiedlichen Unterstützung von Gewerken und Versionen durch die Softwarehersteller (Behaneck, 2015). Auch der Kostenaspekt veranlasst viele Unternehmen dazu vorerst die Projekte nicht mit der BIM-Methode zu bearbeiten. Nicht nur die Beschaffung von neuer Software und die Aktualisierung der Hardware, sondern auch die Einarbeitung bzw. Weiterbildung von Mitarbeitern ist kostenintensiv (Albrecht, 2015). Natürlich kann mit dieser Methode auch Geld gespart werden, durch reduzierte Arbeitsstunden und Personalkosten (Berger, 2017). Viele Firmen werden jedoch durch die hohen Initialkosten abgeschreckt. Eine geringere Verbreitung von BIM lässt sich somit vielleicht auch auf die Nutzer- und Akzeptanzprobleme zurückführen. Durch eine mögliche Einführung dieser neuen Arbeitsweise müssten alte, meist lang verwendete Vorgänge und Arbeitsweisen abgelegt und die meist neue Software gemeistert werden. Aufgrund dessen kann es zu langen Lernphasen für alle Beteiligten kommen (Albrecht, 2015).

2.4 Einsatz des digitalen Planstempels bei der Planung

In der Baubranche schreitet die Digitalisierung, im Vergleich zu anderen Branchen, nur langsam voran(Deutsche Telekom AG, 2018). Deswegen müssen auch im Baubereich neue Konzepte und Arbeitswege entwickelt werden. Oft kommt es vor, dass die Funktionsweise der alten Konzepte in die Neuen übernommen werden können. So unterscheidet sich die Arbeitsweise des digitalen Planstempels nur gering von der des traditionellen Stempels. Bei der digitalen, sowie der analogen Version des Planstempels überprüfen Mitarbeiter einer Baubehörde, ob die eingereichten Unterlagen zum Bauvorhaben zulässig sind und ob das Bauwerk genauso an diesem Ort gebaut werden kann. Falls alle Unterlagen geprüft wurden und richtig vorliegen,

kann der Prüfer das Bauvorhaben genehmigen (Art. 68 Abs. 1 BayBO). Hierbei wird unter anderem der Bauplan mit einem Genehmigungsvermerk versehen. Dieser „Stempel“ bezeugt, dass das Bauvorhaben, genau in diesem Zustand, zur Fertigung freigegeben ist (Art. 68 Abs. 2 BayBO). Es wird somit eine Art Schablone des geprüften Baustandes erstellt. Ähnlich läuft es bei der Prüfung durch einen Fachprüfer ab. Hierbei ist der Prüfumfang jedoch auf die fachspezifischen Bauteile reduziert.

Ein wichtiger Teil eines Planstempels ist die Signatur. Falls es zu Schäden oder Problemen an diesem Bauwerk kommen sollte, können die beteiligten Prüfer ermittelt und die Identitäten mithilfe der Signaturen bestätigt werden. Damit können Nachforschungen nach möglichen Schuldigen begonnen werden. Im Falle des analogen Stempels, muss der Prüfer die Unterlagen mit einer Unterschrift signieren (Technische Bauaufsicht der Stadt Landsberg am Lech, 2020). Ähnlich läuft es bei dem digitalen Planstempel ab. Hierbei wird durch bestätigen der Richtigkeit der Unterlagen im Programm, eine digitale Signatur speziell für diesen Prüfer erstellt. Obwohl auch diese Variante nicht zu 100 Prozent sicher ist, bringt die digitale Version der Unterschrift ein erhöhtes Maß an Sicherheit.

Ein interessanter Vergleich, zwischen den beiden Varianten, ist der Umgang mit den Plänen bzw. Modellen nach der Prüfung. Beim Genehmigungverfahren einer Baubehörde müssen i.d.R. die Unterlagen in dreifacher Ausführung eingereicht werden, damit bei einer erfolgreichen Genehmigung die Baubehörde, die Gemeinde und der Antragssteller eine Version bekommen (§2 BauVorlV). In der Gemeinde und der Baubehörde werden somit die Originalunterlagen aufbewahrt (Technische Bauaufsicht der Stadt Landsberg am Lech, 2020). Dies könnte vereinfacht werden, wenn mit einer speziellen Version des digitalen Planstempels gearbeitet wird. Die Speicherung erfolgt hierbei in einer Blockchain. Dadurch erhalten Gemeinden, Baubehörden und der Antragsteller Zugang zu diesem „Block“, in welchem die Antragsdaten und das Modell gespeichert wurden. Durch die Speicherung der Daten in der Blockchain, kann davon ausgegangen werden, dass das Modell und die anderen Angaben nicht verändert werden können. Eine getrennte Lagerung ist somit hinfällig.

Eine genauere Beschreibung der Funktionsweise von Blockchain und digitaler Signatur wird in Kapitel 4 geliefert.

2.4.1 Einsatzszenarien

Um einen ausführlicheren Überblick zu liefern, wo die Methode des digitalen Planstempels eingesetzt werden kann und welche Vorteile es in diesen Fällen bringt, sollen im folgenden mehrere Einsatzszenarien dargestellt werden. Außerdem kann Anhand dieser Beispiele die allgemeine Funktionsweise genauer erläutert werden. Die vier verschiedenen Einsatzszenarien lassen sich in zwei Übergruppen einteilen. Man unterscheidet zwischen **Gesamtprüfung** und **Teilprüfung**.

In Kapitel 6 werden zwei der im nächsten Abschnitt beschriebenen Szenarien schrittweise durchgespielt, um einen genauen Überblick zur Methode des digitalen Planstempels zu liefern.

Gesamtprüfung

Hiervon ist die Rede, wenn ein Bauvorhaben eingereicht wird, welches in sich vollständig ist. Dabei kann es sich um ein Einfamilienhaus, eine Halle usw. handeln. Findet eine Prüfung und Genehmigung statt, kann dieses Bauwerk komplett gebaut werden.

1. Szenario: Ein Gebäude wird zur Prüfung bei einem Fachprüfer oder bei der Behörde eingereicht. Der verantwortliche Prüfer ist zufrieden mit den beigelegten Unterlagen und erstellt den digitalen Planstempel. Dadurch wird der Stand des Modells festgehalten. Falls nun Schäden oder Probleme beim Bau oder danach auftreten, ist es möglich nachzuvollziehen, welche Personen für die Prüfung verantwortlich waren und inwiefern der gebaute Zustand von der genehmigten Version abweicht.

2. Szenario: Ähnlich wie im ersten Szenario wird ein Gebäude genehmigt und eine Schablone für das Modell erstellt. Nach mehreren Jahren möchten die Besitzer eine Nutzungsänderung der Garage beantragen. Sie stellen einen neuen Antrag auf Umnutzung und reichen das Modell des Hauses mit der umgebauten Garage ein. Der Prüfer in der Baubehörde kann nun die Schablone des Hauses verwenden, um zu ermitteln ob eine Änderung im Bestandsgebäude durchgeführt wurde oder ob die Änderungen wirklich nur die Garage betreffen.

Teilprüfung

Dabei wird das Bauvorhaben Abschnittsweise geprüft um eine Teilgenehmigung zu erhalten. Am Ende steht ein komplett genehmigtes Bauwerk.

3. Szenario: Bei großen Bauvorhaben kommt es teilweise vor, dass Bauabschnitte einzeln genehmigt werden. So wird z.B. zu Beginn nur das Erdgeschoss des Gebäudes geprüft und bewilligt, damit die Arbeiten am Gebäude starten können. Im weiteren Verlauf werden dann die nächsten Bauabschnitte eingereicht. Am Ende ist das Bauwerk komplett genehmigt und kann somit fertig gebaut werden.

Dies stellt eine besondere Herausforderung für die Methode des digitalen Planstempels dar, weil nur genau der genehmigte Bauabschnitt festgehalten werden soll und die anderen Abschnitte nicht. Wenn nun der nächste Bauabschnitt geprüft wird, muss ermittelt werden, ob sich an dem Modell in dem bereits genehmigten Abschnitt etwas verändert hat. Wurde z.B. eine neue Wand im schon geprüften Erdgeschoss geplant, muss dieser Unterschied auffallen und das Erdgeschoss neu geprüft werden.

4. Szenario: Der digitale Planstempel kann auch auf kleinerer Ebene eingesetzt werden. Beispielsweise ist der Einsatz innerhalb eines Planungsbüros möglich. Möchte der Chef die Planung für ein bestimmtes Geschoss abschließen, kann er diesen Teil des Modells absegnen und eine Schablone dafür erstellen. Seine Angestellten werden jedoch weiter an diesem Gebäude planen und somit auch Veränderungen im Modell vornehmen. Deswegen muss der Chef am Ende der Planungsphase überprüfen, ob in dem bereits von ihm genehmigten Abschnitt Änderungen vorgenommen wurden. Die Signierung der abgespeicherten Abschnitte spielt in diesem Beispiel eine nicht so große Rolle wie z.B. bei der Genehmigung durch eine Baubehörde.

Abbildung 2.3 soll nochmal einen kurzen Überblick liefern, in welchen Bereichen der Einsatz des digitalen Planstempels möglich wäre und wie unterschiedlich die Verwendung je nach Szenario ist.

Einsatzmöglichkeiten des digitalen Planstempels		Gesamtprüfung		Teilprüfung	
		1. Szenario	2. Szenario	3. Szenario	4. Szenario
Planung	Grundlagenermittlung				
	Vorplanung				
	Entwurfsplanung				x
	Genehmigungsplanung	x	x	x	x
	Ausführungsplanung			x	
Realisierung		x		x	
Betrieb		x	x		

Abbildung 2.3: Darstellung der Einsatzmöglichkeiten innerhalb der verschiedenen Lebenszyklusphasen.

Kapitel 3

Erläuterungen zur Erstellung und Verwendung von Gebäudemodellen

In diesem Kapitel wird der Umgang mit Bauwerksmodellen erläutert. Dabei wird zuerst ein Building Information Modeling (BIM)-Programm, zur Erstellung eines Gebäudemodells, dargestellt. Im Anschluss werden Möglichkeiten betrachtet, wie das Modell exportiert werden kann und wie dies speziell für das dargestellte Austauschformat aussehen würde. Mit den angefertigten Austauschdateien kann das virtuelle Bauwerk in anderen Applikationen verwendet werden. Dadurch ist der Einsatz von vielen neuen Funktionen möglich. Zur Veranschaulichung werden die Methoden, der einzelnen Abschnitte, an einem Beispielprojekt aufgezeigt. Somit wird zu Beginn ein Gebäudemodell mit Revit erstellt, welches dann in eine IFC-Datei exportiert wird. Die Datei kann zum Abschluss in eine Graphdatenbank geladen oder mithilfe von anderen Anwendung weiterbearbeitet werden.

3.1 Gebäudemodellierung

Das Fundament der BIM-Methode ist das Gebäudemodell (Bundesministerium für Verkehr und digitale Infrastruktur, 2015). Dabei wird das Bauwerk als virtuelle Repräsentation dargestellt. Die Abläufe zur Erstellung des Modells können teilweise von Unternehmen zu Unternehmen unterschiedlich sein. Grundsätzlich läuft die Entwicklung jedoch so ab, dass zuerst die Grobelemente (Wände, Decken) des Gebäudes dargestellt und diese im Anschluss durch Detailelemente (leichte Trennwände, Fenster) ergänzt werden (Sommer, 2016). Um dies umzusetzen, wird eine leistungsstarke Software benötigt. Wie auch in anderen Bereichen gibt es eine Vielzahl an verschiedener Anbieter dieser Programme. Grundsätzlich sind die Funktionen der verschiedenen Applikationen gleich, jedoch gibt es bestimmte Unterschiede, welche die Anwendungen besser bzw. schlechter geeignet für bestimmte Modellierungsaufgaben macht.

Dies hängt jedoch stark von der persönlichen Präferenz des Nutzers ab. Auch in Deutschland ist die Softwarelandschaft stark geteilt, wobei drei Softwarelösungen herausstechen. Zu den am häufigsten verwendeten Programme in Deutschland zählen: Revit (Autodesk), Allplan (Nemetschek) und ArchiCAD (Graphisoft) (BauInfoConsult GmbH, 2019). Auch in anderen Ländern werden diese Anwendungen am häufigsten eingesetzt. Abbildung 3.1 zeigt das Ergebnis einer Umfrage im Vereinigten Königreich. Die Befragten sollten hierbei angeben, welche Programme sie für das Erstellen von Zeichnungen und Modellen verwenden. Die Antworten wurden nach Softwareanbietern gegliedert. Weiter aufgespalten erreicht Revit (Autodesk) einen Nutzungswert von 46% und ist somit die meistgenutzte BIM-Anwendung unter den befragten Personen. Auf dem zweiten Platz liegt ArchiCAD (Graphisoft) mit 15% (NBS, 2019). In den folgenden Unterabschnitten soll nun genauer auf eine der oben genannten Anwendungen eingegangen werden.

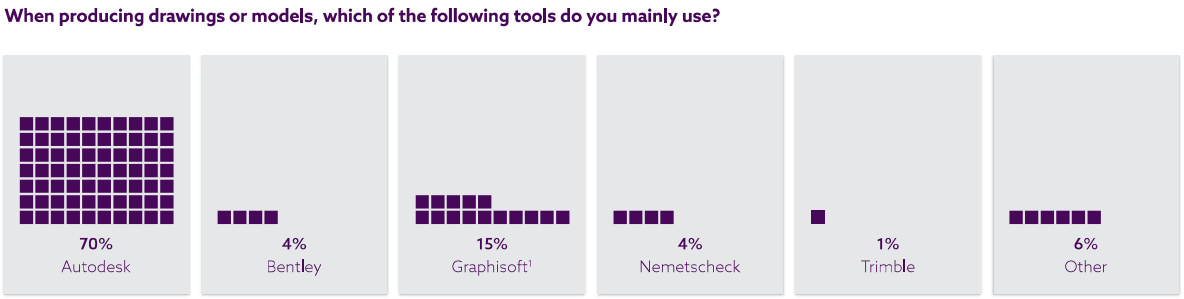

Abbildung 3.1: Umfrage im Vereinigten Königreich: Welches Programm wird für die Erstellung von Zeichnungen und Modellen verwendet? (NBS, 2019, S.32)

3.1.1 Revit

Wie oben bereits erwähnt wurde, zählt Revit zu den meistgenutzten Anwendungen zur Erstellung von Gebäudemodellen (NBS, 2019). Genau wie AutoCAD, ein sehr erfolgreiches Zeichenprogramm, wurde Revit von der US-amerikanischen Firma Autodesk entwickelt. John Walker gründete die Firma 1982 in Kalifornien und brachte bald darauf AutoCAD auf den Markt (FundingUniverse, 2006). Das Unternehmen hat ein weites Spektrum an Programmen für die unterschiedlichsten Aufgaben. Beispielsweise kann die Anwendung *Maya LT* zur 3D-Spieleentwicklung verwendet werden oder es lassen sich mithilfe von *Eagle* Leiterplatten und Stromlaufpläne entwerfen(Autodesk, 2020). Im folgenden wird jedoch nur auf eines der genannten Programme eingegangen, nämlich Revit.

Die Funktionsweise von Revit soll anhand der Erstellung eines einfachen Hauses gezeigt werden. Der erste Schritt der Modellierung ist die Erstellung der Struktur. Somit werden Bodenplatte, Wände, Decken und das Dach erstellt. Jedes dieser Elemente wird als Volumenelement erzeugt. Damit lässt sich sehr schnell das Grundgerüst eines, vom Design her, einfachen Gebäudes herstellen. Natürlich ist das Haus so noch nicht fertig. Es fehlen noch die Detailele-

mente wie z.B. Fenster, Türen und leichte Trennwände (Sommer, 2016). Damit der Sinn von BIM gewahrt wird, müssen die Bauteile mit Informationen gefüllt werden. Dabei erhöht sich der Fertigstellungsgrad, auch Level of Development (LOD) genannt, des virtuellen Bauwerks über die gesamte Planungszeit. Das LOD setzt sich aus dem Level of Geometry (LOG), teilweise auch als „Level of Detail“ definiert, und dem Level of Information (LOI) zusammen (Trimble MEP, 2020). Eine einheitliche Definition wie der Fertigstellungsgrad in den verschiedenen Phasen der Planung aussehen sollte, gibt es noch nicht. Dies muss je nach Projekt speziell definiert werden. Jedoch kann mithilfe einer Skala der steigenden Informationsgehalt, für die einzelnen Phasen, eingeordnet werden. Die Einteilung steigt dabei in 100er Schritten an. Laut Baunetz_Wissen (2020b) sehen die verschiedenen Stufen der Skala wie folgt aus, wobei in diesem Fall die Abkürzung „LOD“ für „Level of Detail“ steht und somit die Geometrie gemeint ist.

> ”
>
> - *Vorentwurfsmodell – LOD/LOI 100*
> - *Entwurfsmodell – LOD/LOI 200*
> - *Genehmigungsmodell – LOD/LOI 300*
> - *Modell zur Angebotskalkulation – LOD/LOI 350 (optional)*
> - *Ausführungsmodell – LOD/LOI 400*
> - *As-built Modell – LOD/LOI 500*
>
> “ (Baunetz_Wissen, 2020b)

Das Kategorisieren der eingebrachten Informationen und den Darstellungen der Bauteile, unterstützt die Planer bei der Kommunikation mit anderen. Meilensteine, sowie Übergaben können hiermit gut definiert werden (Trimble MEP, 2020). Außerdem hilft es dabei festzustellen, welche Modellinformationen zu welcher Projektphase gehören und wie verlässlich diese zu einem bestimmten Zeitpunkt sind (Baunetz_Wissen, 2020b). Abbildung 3.2 soll die Einteilungen von LOI und LOG, in diesem Fall als „Level of Detail“ bezeichnet, beispielhaft anhand einer Tür darstellen.

Nicht nur die Detailelemente, wie Türen und Fenster, werden mit Informationen gefüllt. Auch Elemente wie Wände und Decken benötigen zugeordnete Attribute, wie z.B. das Material aus dem das Grobelement besteht. Das Material kann hierbei aus einer kleinen vorgefertigten Liste auswählt oder selber vom Modellierer definiert werden. Dadurch lässt sich eine Vielzahl an verschiedener Möglichkeiten erstellen. Den Materialien bzw. den erstellten Wandtypen können Preise zugeordnet werden, um eine automatisierte Kostenermittlung durchführen zu lassen.

Abbildung 3.2: Verlauf des Detaillierungsgrads (LOD) und Informationsgrads (LOI) am Beispiel einer Tür. (Schatz, 2020)

Weiterer solcher Auflistungen können mithilfe von Revit erstellt werden. Somit kann sich z.B. der Verantwortliche für den Einkauf, von Materialien und Bauteilen, eine Mengenermittlung anzeigen oder sich alle verbauten Türen auflisten lassen. Die Bauelemente können mit einer Vielzahl an verschiedenen Daten gelistet werden. Die Zusammenstellung dieser ist hierbei dem Ersteller überlassen. Der „Door Schedule“ aus Abbildung 3.3 führt alle Türen des Hauses auf und gibt, je nach Art der Tür, zusätzlich die Kosten, die Anzahl, die Abmessungen, sowie den Typ jeder Tür an. Falls man nun mit den Informationen weiterarbeiten möchte, ist es möglich die Liste zu exportieren. Diese Funktionen vereinfachen das Leben der Projektbeteiligten stark. Wie in Kapitel 2.3.1 bereits erläutert wurde, lässt sich dadurch eine höhere Kostentransparenz erreichen, wodurch die Chance auf versteckte Kosten sinkt. Eine bildliche Darstellung der Erstellungsschritte soll in Abbildung 3.3 gezeigt werden.

Im folgenden Abschnitt wird darauf eingegangen, wie es möglich ist, das Modell zwischen den Projektbeteiligten auszutauschen. Dabei geschieht dies nicht nur nachdem das virtuelle Gebäude fertig modelliert wurde, sondern ein Austausch der Daten kann auch schon während der Erstellungsphase notwendig sein.

3.2 Austauschformat

Austauschformate sind ein wichtiger Teil von Building Information Modeling. Ohne den Austausch unter den Projektbeteiligten könnten viele der Projekte gar nicht durchgeführt wer-

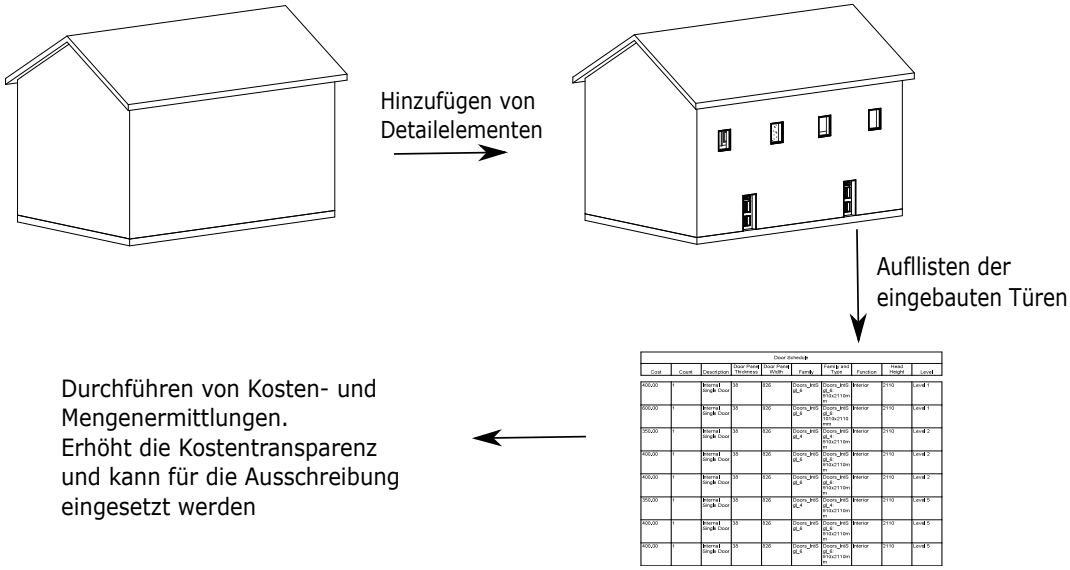

Abbildung 3.3: Grober Erstellungsablauf eines Modells.

den. Essenziell sind diese Austauschformate vor allem bei Projekten, die mit dem „open BIM“ Ansatz laufen, da hierbei mit unterschiedlicher Softwareumgebung gearbeitet wird. Der Austausch von Daten und Modellen kann deswegen nur über programmunabhängige Formate stattfinden (N+P Informationssysteme GmbH, 2018). Deswegen sind diese neutralen, anbieterunabhängigen Formate wichtig, um eine Schnittstelle zwischen der Software des einen Beteiligten und der Software des anderen herzustellen. Dabei ist es sehr wichtig, dass ein Austausch der Daten ohne größere Verluste stattfinden kann.

Im folgenden Unterabschnitt wird eines dieser Datenformate vorgestellt. Mithilfe dessen das Beispiel, welches in Kapitel 3.1.1 gestartet wurde, fortgesetzt wird. Damit kann die generelle Darstellung des Modells in diesem Format betrachtet werden.

3.2.1 IFC

Ein weitverbreitetes und häufig verwendetes Austauschformat ist IFC oder auch „Industry Foundation Classes“ genannt. Laut einer Umfrage verwenden 77% der Befragten dieses Format, wenn sie an einem Projekt beteiligt sind (NBS, 2019). Das Format wird von der internationalen Non-Profit-Organisation *buildingSMART International* entwickelt und gepflegt (buildingSMART, 2020a). Laut buildingSMART (2019d) kann IFC wie folgt beschrieben werden.

> *„In general, IFC, or "Industry Foundation Classes", is a standardized, digital description of the built environment, including buildings and civil infrastructure. It is an open, international standard (ISO 16739-1:2018), meant to be vendor-neutral,*

> *or agnostic, and usable across a wide range of hardware devices, software platforms, and interfaces for many different use cases.“* (buildingSMART, 2019d)

IFC wird hierbei als eine digitale Beschreibung eines Bauwerks bezeichnet. Besonders wird dabei hervorgehoben, dass dieses Format bei vielen verschiedenen Szenarien und anbieterneutral eingesetzt werden kann (buildingSMART, 2019d). Dabei vereint dieses Datenformat die geometrische Darstellung eines Bauvorhabens sowie die bauwerksrelevanten Informationen, welche von den Planern in das Modell eingebracht wurden. Die Einsatzszenarien, wie bereits weiter oben erwähnt wurde, sind sehr breit gefächert. Der Datenaustausch ist somit zwischen vielen verschiedenen Parteien möglich. Es können damit z.B. Architekten, Tragwerksplaner, technische Gebäudeausrüster und Brandschutzplaner untereinander Daten und Modelle austauschen, ohne mit gleichen Softwarelösungen zu arbeiten (Tekla, 2019).
Mithilfe von IFC kann das komplette Modell mit ihren Geometrien und Informationen dargestellt werden. Der Austausch dieser Modelle und Informationen muss dabei jedoch nicht immer dem vollen Umfang der selbst erstellten Dateien entsprechen. Zum Beispiel wäre es nicht sinnvoll dem Hersteller der Fenster das komplette Modell des Gebäudes zu schicken, wenn dieser nur an der Anzahl und Abmessungen der Fenster interessiert ist (buildingSMART, 2020c). Damit bestimmte Austauschszenarien durchgeführt werden können, wurde von buildingSMART das Konzept von Model View Definition (MVD) eingeführt (Borrmann *et al.*, 2018).

Genauer gesagt handelt es sich hierbei um eine Teilmenge des gesamten IFC-Schemas, um den Datenaustausch für einen Arbeitsfall zu beschreiben. Dadurch lässt sich genau bestimmen, wem welche Informationen geliefert werden. So kann bei einem Austausch zwischen Architekt und Tragwerksplaner, das Modell auf den für den Tragwerksplaner relevanten Umfang reduziert werden. Eine Teilmenge des Modells kann mithilfe der meisten Programme, die einen Export von BIM-Daten unterstützen, erstellt werden (buildingSMART, 2020c).

In den folgenden Unterabschnitten wird zuerst auf die technische Struktur des Austauschformats eingegangen und danach eine Datei für das Beispielgebäude erzeugt.

Technische Struktur

Das IFC-Datenmodell kann mithilfe der Datenmodellierungssprache EXPRESS beschrieben werden, welche in der ISO 10303-11 definiert ist. Dieses Daten-Schema kann textlich und grafisch dargestellt werden. Die grafische Darstellung ist auch unter dem Namen EXPRESS-G bekannt (Nahar, 2017). EXPRESS ist ein objektorientiertes Datenmodell. Wie bei der objektorientierten Programmierung werden dabei Objekte als Klassen dargestellt und mit Attributen und Beziehungen zu anderen Objekten gefüllt (Borrmann *et al.*, 2018). Abbildung 3.4 zeigt den Aufbau der Entity-Klasse „IfcProduct“. Die Attribute und Referenzen zu anderen Klassen (Entities) sind dabei gut zu erkennen.

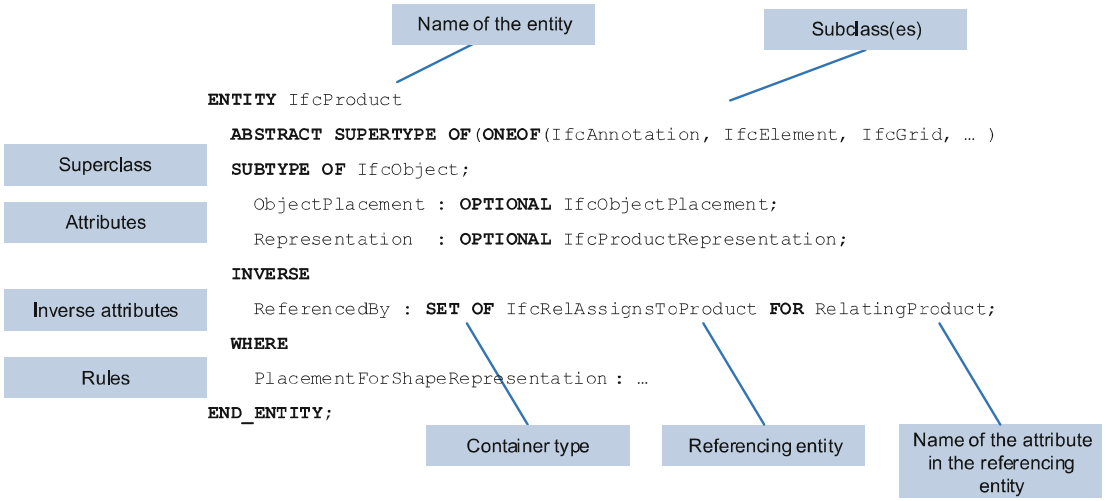

Abbildung 3.4: Definition einer Klasse (entity) mithilfe der Datenmodellierungssprache EXPRESS. (Borrmann *et al.*, 2018, S.87)

Die Modellierung mit EXPRESS kann dabei jedoch keine echten Beispiele aus der Praxis darstellen, es ist nur möglich ein Datenmodell zu definieren (Borrmann *et al.*, 2018). Um ein konkretes Beispiel der EXPRESS-Definition aufzuzeigen werden andere Methoden benötigt. Möglich ist dies unter anderem mit einer Extensible Markup Language (XML) Datei oder einem STEP Physical File (SPF) Dokument mit der Dateiendung „.ifc“ (Nahar, 2017). Dabei wird die Standard for the Exchange of Product model data (STEP)-Variante am häufigsten verwendet (buildingSMART, 2019a). Die STEP-Datei ist vom Aufbau her zweigeteilt. Es ist dabei einmal in den *Header* und in den *Data* Abschnitt unterteilt. Der *Header* Teil beinhaltet hierbei die allgemeinen Informationen zu dieser IFC-Datei (Nahar, 2017). Unter anderem wird die EXPRESS Version sowie die verwendete IFC Version angegeben. Außerdem liefert der *Header* Informationen zum Erstellungszeitpunkt, Namen der Datei und Angaben zum Ersteller. Das Codebeispiel 3.1 zeigt den *Header* des Beispielgebäudes.

Der Abschnitt *Data* einer IFC-Datei soll die konkreten, in EXPRESS definierten Klassen (Entities) eines Bauwerks darstellen. Dabei können für ein einfaches Einfamilienhaus schon mal mehrere zehntausend Einträge zusammenkommen. Damit die einzelnen Einträge nach bestimmten Kriterien gefiltert werden können, lassen sich diese einem bestimmten Abschnitt zuordnen. Diese Abschnitte können wiederum anderen Überkategorien zugeschrieben werden. Daraus ergibt sich eine Baumstruktur, die in Abbildung 3.5 dargestellt wird. Durch die Zuordnung zu einem bestimmten Gebäude oder Geschoss können sich die Elemente nach diesem Kriterien auflisten lassen (Autodesk, 2018).

Wie bereits weiter oben beschrieben wurde, werden den Klassen Attribute und Referenzen zu anderen Entitäten zugeordnet. Durch diese Referenzierungen ist es möglich Bauteile miteinander zu verknüpfen. Abbildung 3.6 zeigt die Beziehung zwischen einer Wand im Gebäude und dem, in dieser Wand, verbauten Fenster. Über die Klassen *IfcRelVoidsElement*, *IfcO-*

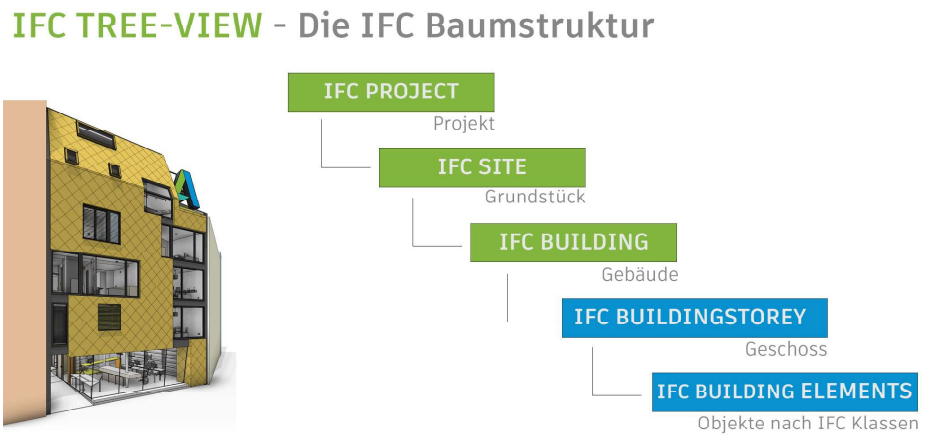

Abbildung 3.5: Darstellung der IFC Baumstruktur. (Autodesk, 2018, S.9)

peningElement und *IfcRelFillsElement* wird eine Beziehung zwischen der Wand und dem Fenster hergestellt. Durch zahlreiche solcher Verknüpfungen wird das Gebäude in der IFC-Datei dargestellt. Somit lassen sich der Wand noch andere geometrische und alphanumerische Informationen zuordnen. Beispielsweise wird der Wand mithilfe einer weiteren „Rel"-Klasse das Material zugeordnet. Dabei handelt es sich bei all diesen Entities um Unterklassen von *IfcRelationship* (Borrmann *et al.*, 2018).

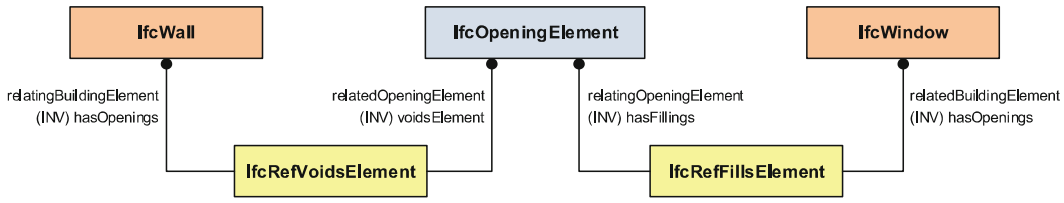

Abbildung 3.6: Darstellung der Beziehung zwischen einer Wand und dem verbauten Fenster. (Borrmann *et al.*, 2018, S.94)

Beispiel

In diesem Abschnitt sollen die Konzepte aus dem vorangegangen Absatz anhand eines Beispiels veranschaulicht werden. Sobald ein Austausch zwischen Projektbeteiligten, mit unterschiedlicher Softwareumgebungen, notwendig ist, kommt das Format IFC zum Einsatz. Eine IFC-Datei lässt sich dabei sehr leicht erstellen. Bei Revit muss dazu nur das Modell, als IFC-Format, exportiert werden. Hierbei kann noch genauer definiert werden, welche Informationen und wie detailliert das Modell übernommen werden soll und mit welchen Daten der Header gefüllt wird. In folgender Auflistung wird die Kopfzeile für das Beispielprojekt dargestellt. Diese liefert Informationen zur Versionsnummer von EXPRESS und gibt die Version des IFC-Schemas an. Außerdem wird in diesem Fall der Name und die E-Mail des Erstellers aufgeführt.

Listing 3.1: Header einer IFC-Datei.

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('ViewDefinition [CoordinationView_V2.0]'),'2;1');
FILE_NAME('Project Number','2020-05-06T09:27:33',('Max Mustermann','max.
    mustermann@email.com'),(''),'The EXPRESS Data Manager Version
    5.02.0100.07 : 28 Aug 2013','20180328_1600(x64) - Exporter 19.0.1.1 -
    Alternate UI 19.0.1.1','');
FILE_SCHEMA(('IFC2X3'));
ENDSEC;
```

Die Kopfzeile alleine bringt jedoch noch nicht viel. Der wichtige Teil, der *Data*-Abschnitt, liefert die geometrischen und alphanumerischen Informationen des Gebäudemodells (Baunetz_Wissen, 2020a). Die folgenden Codebeispiele werden nur einen sehr kleinen Teil der gesamten Datei darstellen, da für so ein Gebäude mehrere zehntausend Einträge zusammenkommen. Am Interessantesten, an diesem Beispiel, ist wahrscheinlich die Definition der einzelnen Bauelemente. Das folgende Textbeispiel zeigt deswegen die Definition einer Wand. Das Listing 3.3 stellt passend zur Instanziierung der Wand, einen Teil des Vererbungsgraphen der Entity-Klasse „IfcWallStandardCase“ dar.

Listing 3.2: Definieren einer Wand in einer IFC-Datei.

```
#269= IFCWALLSTANDARDCASE('0grKb8u8L7$AHILoHRbfKk', #47, 'Basic Wall:Generic -
    375mm:204901', $, 'Basic Wall:Generic - 375mm:252', #217, #267, '204901',$);
```

Listing 3.3: Teile des Vererbungsgraphen der Entity-Klasse IfcWallStandardCase. (buildingSMART, 2019c)

```
Inheritance Graph:
ENTITY IfcWallStandardCase
 ENTITY IfcRoot
          1 GlobalId          :     IfcGloballyUniqueId;
          2 OwnerHistory      :     OPTIONAL IfcOwnerHistory;
          3 Name              :     OPTIONAL IfcLabel;
          4 Description       :     OPTIONAL IfcText;
 ENTITY IfcObject
          5 ObjectType        :     OPTIONAL IfcLabel;
 ENTITY IfcProduct
          6 ObjectPlacement   :     OPTIONAL IfcObjectPlacement;
          7 Representation    :     OPTIONAL IfcProductRepresentation;
 ENTITY IfcElement
          8 Tag               :     OPTIONAL IfcIdentifier;
 ENTITY IfcWall
          9 PredefinedType    :     OPTIONAL IfcWallTypeEnum;
END_ENTITY;
```

Das Listing 3.2 zeigt die Serialisierung des Datenmodells. Auf den ersten Blick sieht die Instanziierung der Wand sehr unübersichtlich aus. Dabei hat jedoch jedes Attribut innerhalb

der runden Klammer seine Berechtigung. Die Attribute werden untereinander mit Kommas abgegrenzt. Die Zahlen innerhalb des Vererbungsgraphen geben die Reihenfolge der Attribute an. An erster Stelle steht die *GlobalId* dieser Instanz. Dabei handelt es sich um eine einzigartige Kennung, auch Globally Unique Identifier (GUID) genannt (Tauscher *et al.*, 2016). Diese Kennung muss bei jeder IFC Objekt Instanz vorhanden sein (buildingSMART, 2020b). Es ist auch das einzige Attribut, bei dem im Listing 3.3 nicht der Zusatz „OPTIONAL“ vorhanden ist. Der nächste Teil der Klammer referenziert die Entity-Klasse *IfcOwnerHistory*. Diese soll die Identifikation und die Geschichte der Datei beinhalten. Unter anderem wird aufgezeichnet welcher Nutzer und welche Applikation zuletzt Änderungen vorgenommen hat (buildingSMART, 2020b). Die Referenzierung erfolgt hierbei mithilfe des Symbols „#“ und der Nummer der anderen Instanz. In diesem Beispiel würde man die IfcOwnerHistory-Instanz bei der Nummer 47 finden. Die Nummer drei stellt den Namen der Wand dar. Der Name wurde in Revit vergeben und somit übernommen. Der vierte Teil der Klammer sollte die *Description* der Instanz beinhalten. In diesem Fall ist die *Description* mit dem Wert *null* belegt. Dies ist möglich, da im Vererbungsgraphen der Zusatz *OPTIONAL* vorhanden ist. Wenn ein Attribute mit dem Wert *null* belegt werden soll, kann dies mit folgenden Symbol „$“ dargestellt werden (Koonce *et al.*, 1998). Der *ObjectType* (5) hat in diesem Beispiel die gleiche Bezeichnung wie der Name. Dies hat den einfachen Grund, dass der Name in Revit nicht neu definiert wurde. Die nächsten beiden Referenzen stellen das *ObjectPlacement* (6) und die *Representation* (7) der Wand dar. Die referenzierten IFC-Instanzen sind dabei *IfcLocalPlacement* und *IfcProductDefinitionShape*. Das *ObjectPlacement* soll hierbei das Element im Modell platzieren und die *Representation* gibt der Wand eine Form. Diese geometrischen Informationen werden dabei jedoch nicht bereits in den „abstract supertype“ Klassen *IfcObjectPlacement* und *IfcProductRepresentation* festgelegt, sondern werden in Klassen, die von den *supertype* Klassen erben, definiert (buildingSMART, 2020b).
Den Abschluss dieser Instanziierung bilden die Attribute *Tag* (8) und *PredefinedType*(9). *Tag* kann als eine Art zusätzliches Label dienen. Dort kann die Seriennummer des Produkts oder die Positionsnummer der Wand eingetragen werden (buildingSMART, 2020b). Der *PredefinedType* ist vom Typ *IfcWallTypeEnum* und soll einen generischen Wandtypen angeben. Es kann z.B. zwischen *STANDARD*, *SOLIDWALL*, *PARAPET* usw. unterschieden werden (buildingSMART, 2020b). Im falle des Beispiels ist das Attribut mit Wert *null* festgelegt worden.

Listing 3.4: Von IfcWallStandardCase referenzierte Klassen.

```
1 #47= IFCOWNERHISTORY(#44,#5,$,.NOCHANGE.,$,$,$,1580978636);
2 #217= IFCLOCALPLACEMENT(#126,#216);
3 #267= IFCPRODUCTDEFINITIONSHAPE($,$,(#222,#265));
```

Listing 3.4 zeigt die Instanziierungen der referenzierten Entities und dessen Verknüpfungen zu anderen Klassen. Auch diese Klassen referenzieren andere, welche mit wieder anderen verknüpft sind. Somit entsteht eine lange Kette an verschiedenen Entities. Die Wand, welche

in unserem Fall das Ausgangselement ist, erhält durch diese Verkettung viele verschiedene geometrische und alphanumerische Attribute. Somit lassen sich komplexe Gebäudestrukturen realisieren und mit baurelevanten Informationen verbinden.

Sind alle Verbindungen zur Darstellung des Gebäudes geknüpft, ist die Serialisierung des Modells abgeschlossen und man erhält als Ergebnis eine IFC-Datei. Wie diese Datei weiterverwendet werden kann wird im nächsten Abschnitt genauer erläutert.

3.3 Informationsbeschaffung

Nachdem die IFC-Datei erfolgreich erstellt wurde, stellt sich nun die Frage wie diese eingesetzt werden kann. Auf diese Frage gibt es eine Vielzahl von Antworten. Die Datei kann z.B. verwendet werden, um das Modell mithilfe von anderen Gebäudemodellierungsprogrammen zu ergänzen. Zusätzlich lässt sich das IFC-Dokument mit sogenannten *Viewern* grafisch darstellen, wodurch auch eine Betrachtung der IFC-Struktur möglich ist. *BIM-Toolkits* und *IFC-Queries* sind zwei weitere Konzepte, die eingesetzt werden können, um Informationen aus dem Modell zu ziehen oder teilweise auch das Bauwerk ergänzen können. Auf diese beiden Themen wird in den folgenden Abschnitten genauer eingegangen. Welche der genannten Konzepte bei der Umsetzung des digitalen Planstempels zum Einsatz gekommen sind und in welchem Umfang, wird im Kapitel 5 erläutert.

3.3.1 BIM-Toolkits

Es gibt eine große Anzahl an Toolkits (englisch für Werkzeugsatz), die unterstützend bei der Arbeit mit IFC eingesetzt werden können. Dabei sind diese Werkzeuge in den unterschiedlichsten Programmiersprachen erstellt worden. Damit diese Tools nun mit der IFC-Datei im STEP-Format, arbeiten können, werden bestimmte Bindungen (*Binding*) benötigt (Amann *et al.*, 2015). Die folgenden Unterabschnitte zeigen zwei verschiedene Arten, wie diese Bindung abläuft. Dazu passend wird jeweils ein Toolkit vorgestellt.

Early Binding

Amann *et al.* (2015) erklärt *Early Binding* wie folgt.

> *„Beim Early Binding werden durch ein geeignetes Mapping-Verfahren die Entitäten des zur STEP P21-Datei gehörenden EXPRESS-Schemas in der Zielprogrammiersprache (Hostsprache) abgebildet.“* (Amann *et al.*, 2015, S.194)

Somit werden aus dem EXPRESS-Schema, Klassen in der verwendeten Programmiersprache. Dabei muss das *Binding* für jedes Schema nur einmal durchgeführt werden. Falls nun

aber neue IFC-Entities hinzukommen oder bereits vorhandene verändert wurden, muss ein *Mapping* der Daten erneut erfolgen (Amann *et al.*, 2015).

Ein Toolkit, welches das *Early Binding*-Prinzip verwendet, ist das *xBIM toolkit* (Esser & Aicher, 2019). Damit können im IFC-Format gespeicherte Modelle grafisch dargestellt und bearbeitet werden. Das *open-source*-Toolkit ist in der objektorientierten Programmiersprache *C#* geschrieben. Die Verwendung dieses Hilfsmittel ist unter anderem durch das *NuGet*-System möglich. Mithilfe dessen ist ein Zugriff auf die abgebildeten EXPRESS-Klassen in *C#* möglich (xbim, 2020b).

Die folgenden Codebeispiele sollen einen kleinen Überblick über den Leistungsumfang von xbim liefern. Es kann z.B. das Modell nach bestimmten Elementtypen durchsucht werden (Listing 3.5). Außerdem ist es möglich Bauteile zu löschen (Listing 3.6). Wobei dies nur sehr vorsichtig betrieben werden sollte, da die verknüpften Objekte dieses Elements nicht gelöscht werden (xbim, 2020a).

Listing 3.5: Herausfiltern von Wänden aus der IFC-Datei. (xbim, 2020a)

```
1 var firstWall = mode.Instances.FirtsOrDefault<IfcWall>();
2 var allWalls = model.Instances.OfType<IfcWall>();
3 var specificWall = model.Instances.Where<IfcWall>(w => w.Name == "
    Brick-wall");
```

Listing 3.6: Löschen einer bestimmten Tür im Modell. (xbim, 2020a)

```
1  //Ermitteln einer vorhandenen Tür
2  var id = "3cUkl32yn9qRSPvBJVyWYp";
3  var theDoor = model.Instances.FirstOrDefault<IIfcDoor>(d => d.GlobalId ==
       id);
4
5  using (var txn = model.BeginTransaction("Delete the door"))
6  {
7      //Löschen der Tür mit der oben definierten id
8      model.Delete(theDoor);
9      //commit changes
10     txn.Commit();
11 }
```

Late Binding

Das *Late Binding* wird, wie der Name schon andeutet, erst später durchgeführt. Das bedeutet, dass im Vergleich zum *Early Binding* das „binding“ erst während der Laufzeit stattfindet. Die Verbindung wird hierbei über das Standard Data Access Interface (SDAI) hergestellt. Dieses Interface ist eine Programmierschnittstelle, mit dessen Hilfe STEP-Dateien verwendet

werden können (Amann *et al.*, 2015). Somit lässt sich Software entwickeln, die auf STEP-Dateien basiert (STEP Tools, Inc., 2019). Die Bindings sind für viele verschiedene Programmiersprachen definiert worden, jedoch sind nur die folgenden drei standardisiert C, C++, und Java. Der Vorteil dieser Art von Binding ist, dass nicht wie beim Early Binding zuerst die Entitäten des EXPRESS-Schemas in Entitäten der gewünschten Programmiersprache umgewandelt werden müssen. Ein Zugriff auf die EXPRESS-Klassen ist dadurch während der Laufzeit möglich (Amann *et al.*, 2015).

Ein Tool, dass mit der Late Binding Methode arbeitet ist die Programmbibliothek *IfcOpenShell* (IfcOpenShell, 2019). Ähnlich wie bei dem Early Binding Toolkit können Bauteile herausgefiltert, hinzugefügt und verändert werden. Die Codestruktur ist dabei jedoch etwas anders. Um die genannten Aktionen auszuführen, muss die Schnittstelle mit dem Namen der Entitäten angepeilt werden. Listing 3.7 zeigt dieses Vorgehen im IfcOpenshell Python Interface. Durch den String "IfcWall" werden alle Wände, die in der IFC-Datei *Einfaches_Haus.ifc* beinhaltet sind, herausgelesen. Den Wänden, die in der Liste *walls* gespeichert wurden, können weitere Informationen entnommen werden.

Listing 3.7: Laden der IFC-Datei und herausfiltern aller Wände in diesem Gebäude.

```
1 import ifcopenshell as ifc
2
3 file = ifc.open(r"C:\...\Einfaches_Haus.ifc")
4 walls = file.by_type("IfcWall")
5 for wall in walls:
6     print(wall)
```

Neben dem einfachen Herauslesen von Bauteilen lässt sich auch das IFC-Modell erweitern. So ist es möglich neue Wände oder andere Elemente hinzuzufügen oder die bereits enthaltenen Instanziierungen mit neuen Attributen zu ergänzen. Der Code, für eine Erzeugung neuer Bauteile, ist dafür um einiges komplexer und umfangreicher, als der bereits dargestellte. Auflistung 3.8 zeigt nur einen kleinen Ausschnitt des notwendigen Codes zum erstellen einer neuen Wand.

Listing 3.8: Erstellen einer neuen Wand mit IfcOpenShell. (Chen, 2015)

```
1 axis_representation = ifcfile.createIfcShapeRepresentation(context, "Axis
    ", "Curve2D", [polyline])
2
3 body_representation = ifcfile.createIfcShapeRepresentation(context, "Body
    ", "SweptSolid", [solid])
4
5 product_shape = ifcfile.createIfcProductDefinitionShape(None, None, [
    axis_representation, body_representation])
6
```

```
7 wall = ifcfile.createIfcWallStandardCase(create_guid(), owner_history, "
    Wall", "An_awesome_wall", None, wall_placement, product_shape, None)
```

3.3.2 IFC-Queries

Der komplexe Aufbau einer IFC-Datei macht es sehr schwierig Informationen aus dieser herauszuziehen. Es wird viel Know-how benötigt um sich in der Datei zurecht zu finden und Daten herauszulesen. Beispielsweise stellt es ein hohen Aufwand dar, alle Fenster einer Wand aus dem Dokument zu entnehmen (Tauscher *et al.*, 2016). Deswegen werden andere Methoden benötigt, die eine einfache Handhabung mit einer IFC-Datei ermöglichen. BIM-Queries (englisch für Abfragen) könnten hierbei eine wichtige Rolle spielen.

Ein Ansatz, der bereits vor mehr als 20 Jahren beleuchtet wurde, ist die Express Query Language (EQL) (Koonce *et al.*, 1998). Mithilfe dieser Sprache ist es möglich Daten im STEP-Format zu filtern (Tauscher *et al.*, 2016). Ähnlich wie bei der Structured Query Language (SQL) ist es laut Koonce *et al.* (1998) möglich, bei der EQL Instanziierungen und Attribute abzufragen, diese hinzuzufügen und falls notwendig zu löschen. Für Abfragen werden ähnliche Befehle wie bei SQL verwendet, z.B. *SELECT*,*FROM* und *WHERE* (Koonce *et al.*, 1998). Im Gegensatz zu EQL handelt es sich bei der nächsten Abfragesprache um eine, die speziell auf Gebäudemodelle im IFC-Format ausgelegt ist (Daum & Borrmann, 2014). Query Language for Building Information Models (QL4BIM) ist dabei nicht nur zum filtern von einfachen Attributen vorgesehen, sondern kann auch Abfragen zu räumlichen Informationen ausführen (Tauscher *et al.*, 2016). Unter anderem kann mit dem Befehl *Touches* getestet werden, ob sich die genannten Elemente berühren (Daum, 2017).

Eine weitere Möglichkeit, um ein IFC-Gebäudemodell nach Informationen zu filtern, ist der Einsatz von Graphendatenbanken. Der nächste Abschnitt liefert zuerst allgemeine Informationen zu Graphen und stellt dann die Abfragebefehle für eine bestimmte Datenbank dar.

Graphen

Strukturell werden Graphen mit Knoten und Kanten beschrieben. Dabei werden Informationen in den Knoten gespeichert und Beziehungen zwischen den Knoten, mithilfe von Kanten dargestellt (Luber & Litzel, 2019). Ähnlich wie bei einem Graphen, gibt es auch bei der IFC-Datenstruktur Attribute und Beziehungen zwischen den Klassen. Deswegen ist es möglich eine IFC-Datei als Graphen abzubilden (Xu, 2018). Dabei wird das IFC-Objektmodell in einen gerichteten Graphen umgewandelt, wobei die Klassen als Knoten und die Beziehungen zwischen diesen als Kanten dargestellt werden (Tauscher *et al.*, 2016). In Abbildung 3.7 wird beispielhaft der Knoten der Wand #269, mit den Beziehungen zu *IfcOwnerHistory*, *IfcLocalPlacement* und *IfcProductDefinitionShape*, dargestellt.

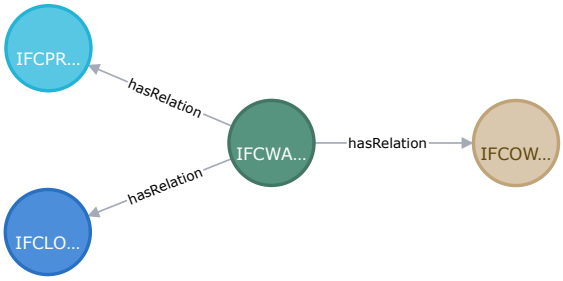

Abbildung 3.7: Darstellung einer Wand als Knoten in der Graphdatenbank Neo4j. Aufzeigen der Verbindungen zwischen IfcWallStandardCase (Grün), IfcOwnerHistory (Beige), IfcLocalPlacement (Blau) und IfcProductDefinitionShape (Türkis).

Am Graphen können nun mit bestimmten Algorithmen Abfragen durchgeführt werden. Um z.B. schnellstmöglich das Material einer Wand herauszufinden, würde es sich anbieten den *Kürzesten Pfad* oder auch *shortest path Algorithmus* anzuwenden (Tauscher *et al.*, 2016). Zur Umsetzung dieser theoretischen Ansätze bieten sich Graphdatenbanken an. Damit das IFC-Instanzmodell in einer Graphdatenbank verwendet werden kann, müssen die Klassen und Beziehungen mithilfe eines speziellen Programms eingelesen werden. Somit lassen sich unter anderem der *shortest Path Algorithmus* und einfache Datenbankenabfragen umsetzten. Zur Veranschaulichung der Möglichkeiten einer Graphdatenbank wird die Open-Source-Datenbank *Neo4j* verwendet. Alle folgenden Grafiken und Codebeispiele, sowie die Abbildung 3.7, wurden mithilfe dieser Datenbank erstellt. Beispielsweise lässt sich die einfache Verbindung zwischen einer Wand und dem integrierten Fenster mithilfe der folgenden Abfrage darstellen. Das erzeugte Ergebnis dieses Befehls wird grafisch in der Abbildung 3.8 aufgezeigt.

Listing 3.9: Abfrage um die Fenster innerhalb einer bestimmten Wand darzustellen.

```
1 MATCH(n:IFCWALLSTANDARDCASE)--(a:IFCRELVOIDSELEMENT)--(b:
      IFCOPENINGELEMENT)--(c:IFCRELFILLSELEMENT)--(d:IFCWINDOW) WHERE n.
      EntityId=33842 RETURN n,a,b,c,d
```

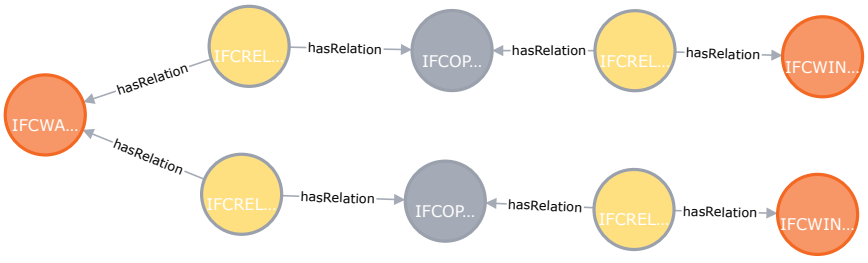

Abbildung 3.8: Darstellung der Beziehungen zwischen einer Wand und den verbauten Fenstern mit einer Graphdatenbank.

Wie bereits in der Veröffentlichung von Ismail *et al.* (2017) aufgezeigt wurde, sind die Einsatzmöglichkeiten von IFC, im Zusammenspiel mit Neo4j, sehr breit gefächert. Unter anderem wurden in diesem Text dargestellt, wie mithilfe von Abfragen ein Rettungsweg erstellt

werden kann (Ismail *et al.*, 2017). Somit lässt sich sagen, dass die Umwandlung einer IFC-Datei in einen Graphen, die Arbeit mit virtuellen Bauwerken vereinfachen kann. Unter anderem ist der Einsatz dieser Speichermethode und dessen Abfragen, bei der Visualisierung der IFC-Beziehungen und der Analyse des Gebäudes, sehr hilfreich (Ismail *et al.*, 2017). Da die Verbindungen der Knoten bei den meisten Bauwerken gleich sind, können somit standardisierte Abfragen erstellt werden. Es können z.B. alle Fenster einer Wand aufgelistet oder die zusätzlichen Attribute einer Tür, wie Kosten und Material, dargestellt werden. Im Vergleich zu anderen Methoden, die Queries für IFC-Modelle erzeugen, ist für diese Variante jedoch ein zusätzlicher Zwischenschritt notwendig. Die Abfragen können nicht direkt an der IFC-Datei durchgeführt werden, sondern es müssen zuerst die Entities der Datei in eine Graphdatenbank importiert werden. Zusätzlich ist ein erweitertes Wissen bezüglich der IFC-Beziehungen notwendig, um solche Abfragen wie im Beispiel 3.9 zu erstellen.

Kapitel 4

Allgemeine Grundlagen zur Informationssicherheit

Nachdem bereits in den ersten Kapiteln erläutert wurde, wann dieser digitale Planstempel zum Einsatz kommen kann und wie die Gebäudemodelle erstellt werden, soll nun in diesem Abschnitt darauf eingegangen werden, wie die Datenintegrität der geprüften Modelle gewährleistet werden kann. Außerdem findet eine Betrachtung von kryptografischen Werkzeugen statt, mit denen ein Vergleich von unterschiedlichen Modellen möglich ist. Inwiefern jedoch die genannten kryptografischen Verfahren bei der Umsetzung des digitalen Planstempels zum Einsatz kommen, wird im Kapitel 5 genauer erläutert.

Der erste Abschnitt (4.1) geht auf die Verschlüsselung der Modelle ein. Diese wäre notwendig, um die Betrachtung der gespeicherten Daten durch andere zu verhindern. Am Anfang dieses Abschnitts (4.1.1) wird ein kleiner historischer Überblick geliefert, im Anschluss werden verschiedene Verschlüsselungsarten erläutert. Im nächsten Abschnitt werden Hash-Funktionen behandelt (4.2). Diese sind ein sehr wichtiges Werkzeug der Kryptografie und werden in den unterschiedlichsten Verfahren (digitale Signatur und Blockchain) eingesetzt. Der nächste Teil (4.3) beschäftigt sich mit der Signatur des geprüften Bauwerks. Damit auch nach Jahren eine Nachprüfbarkeit besteht, muss das Modell mit einer Signatur versehen werden. Somit lässt sich dem Bauwerk genau zuordnen, welche Person für die Genehmigung bzw. Prüfung des Gebäudes zuständig war. Zuerst wird hierbei allgemein auf das Thema eingegangen und im Anschluss werden verschiedene Signaturarten aufgezeigt. Der letzte Teil (4.4) dieses Kapitels stellt eine Methode vor, wie die Integrität des Modells garantiert werden kann, nämlich die Blockchain. Mithilfe dieser Speicherungsart fällt sofort auf, wenn Änderungen an den gespeicherten Daten vorgenommen wurden. Eine Manipulation der Daten ist damit ausgeschlossen. Somit ist es möglich den original Stand des genehmigten oder geprüften Gebäudes abzurufen. Ähnlich wie im Unterkapitel „Digitale Signatur“ wird zuerst auf das allgemeine Konzept dieser Technik eingegangen und danach werden verschiedene Arten vorgestellt.

4.1 Verschlüsselung

Der Wunsch nach privaten und verschlüsselten Konversationen ist seit Anbeginn der Zeit ein wichtiges Element des menschlichen Lebens. Sei es im Krieg, um verschlüsselte Angriffspläne zu verschicken, oder in der heutigen Zeit, um private Nachrichten zwischen Freunden hin und her zu senden oder die persönlichen Daten beim Onlinebanking zu sichern. Das Grundkonzept einer Verschlüsselung ist dabei recht einfach. Der Ausgangspunkt sind Daten, die nur von bestimmten Beteiligten betrachtet werden sollten. Diese Daten, seien es Texte, Bilder oder andere Dateien, können mithilfe eines Schlüssels codiert werden. Damit die andere Partei auch Zugang zu dem Inhalt der Daten hat, muss dieser Partei der Schlüssel bekannt sein. Diese Methode ist jedoch nur solange sicher, wie der Chiffrierungsschlüssel für andere unbekannt bleibt. Diese Art der Chiffrierung wird auch **symmetrische Verschlüsselung** genannt. Symmetrisch, weil die Ver- und Entschlüsselung mit dem gleichen Schlüssel durchgeführt wird. Diese Variante wird bereits seit mehreren tausend Jahren, in den unterschiedlichsten Arten, verwendet (Kryptowissen.de, 2020b). Eine andere Variante der Codierung ist die **asymmetrische Verschlüsselung**. Der Unterschied zur symmetrischen Variante ist, dass zwei sich ergänzende Schlüssel benötigt werden. Einer davon wird für die Verschlüsselung und der andere für die Entschlüsselung verwendet (Kryptowissen.de, 2020a). Eine genauere Erläuterung der Funktionsweise und Einsatzmöglichkeiten, wird im Unterkapitel 4.1.2 geliefert. Doch zuerst sollen folgende Beispiele einen kleinen historischen Überblick, über die Entwicklung der symmetrischen Verschlüsselungsmethoden liefern.

4.1.1 Historischer Überblick

Bereits ca. 600 v. Chr. wurde eines der ersten bekannten Verschlüsselungstechniken der Welt verwendet. Die Spartaner haben mithilfe eines Stabes und einem Pergamentband Nachrichten verschlüsselt. Dabei wurde das Pergamentband um einen Stab mit bestimmten Durchmesser gelegt und beschrieben. Das Band wurde losgeschickt und konnte nur von den Personen gelesen werden, die einen Stab mit gleichem Durchmesser verwendet haben. Bei der Verwendung eines Stabes mit anderem Durchmesser würde die Nachricht keinen Sinn ergeben. Der Stock ist somit der Schlüssel, mit dem diese Nachricht verschlüsselt wurde (Stöltzel, 2010).
Mehrere hundert Jahre später findet eine der bekanntesten historischen Verschlüsselungsmethoden ihre Verwendung. Diese Methode, die heute unter dem Namen Caesar-Verschlüsselung bekannt ist, ist vom Konzept her sehr einfach. Jeder Buchstabe des Alphabets wird um eine bestimmte Anzahl an Stellen verschoben. Die Anzahl der Verschiebungen ist dabei der Schlüssel. Wie der Name schon sagt, wurde diese Technik von Caesar verwendet um seine Nachrichten zu verschlüsseln. Die damals von ihm verwendete Austauschzahl war die drei. Somit würde aus einem A ein D, aus einem K ein N, aus X ein A usw. werden. Der Empfänger der Nachricht müsste nur den Schlüssel anwenden und könnte den Text entziffern. Nach heu-

tigen Standards gilt dieses Verfahren als sehr unsicher (Serlo, 2020).
Zum Abschluss des historischen Überblicks, wird auf eine komplexere Verschlüsselungsmethode eingegangen, nämlich auf eine Verschlüsselungsmaschine. Diese Maschine wurde von Arthur Scherbius 1918 entwickelt und trägt den Namen *Enigma*. Die Enigma-Maschine wurde von der Wehrmacht im 2. Weltkrieg zur Verschlüsselung von geheimen Nachrichten verwendet. Die Funktionsweise des Gerätes, sieht dabei wie folgt aus. Die Nachricht kann über eine Tastatur eingegeben werden. Wird die Taste eines bestimmten Buchstabens gedrückt, wird ein elektrischer Impuls durch ein Steckbrett und mehrere Walzen gesendet. Das Steckbrett und die Walzen verändern durch verschiedene Verknüpfungen den eingegebenen Buchstaben, wodurch am Ende ein komplett anderer Buchstabe herauskommt. Die Codes dieser verschachtelten und komplexen Verschlüsselungsmaschine konnten nur unter großen Aufwand durch die britische Gruppe um Alan Turing geknackt werden (Manz, 2019).

Bei all diesen genannten Verfahren handelt es sich um symmetrische Verschlüsselungen. Erst sehr spät wurde die Methode der asymmetrischen Verschlüsselung entwickelt. Die heutige Verwendung dieser beiden Varianten werden im folgenden Abschnitt erläutert.

4.1.2 Verschlüsselungsarten

Symmetrische Verschlüsselung

Wie bereits erwähnt wurde, wird bei dieser Codierungsart der gleiche Schlüssel für die Ver- und Entschlüsselung verwendet. Deswegen hat die Geheimhaltung des Schlüssels höchste Priorität. Das Verfahren wird somit auch *Secret-Key-Verschlüsselung* genannt. Der Vorteil dieser Methode ist, dass die Codierung schnell durchgeführt wird. Aufgrund dessen ist diese Chiffrierung die bevorzugte Verschlüsselungsmethode für große Datenmengen. Um nach dem historischen Überblick auch noch eine moderne Variante darzustellen, wird im nächsten Absatz die heutzutage meistgenutzte symmetrische Verschlüsselungsmethode dargestellt (Paar & Pelzl, 2016).

Dabei handelt es sich um den Advanced Encryption Standard (AES). Dieser wurde bei einer öffentlichen Ausschreibung im Jahre 1997 von Joan Daemen und Vincent Rijmen unter dem Namen *Rijndael* eingereicht. Drei Jahre später wurde dieses Verfahren zum Sieger der Ausschreibung gekürt und als neuer *Encryption Standard* festgelegt. Der AES ist somit der Nachfolger des Data Encryption Standard (DES), welcher Aufgrund seiner geringen Schlüssellänge (56 Bit) nicht mehr gegen die *Brute-Force-Angriff* modernerer Computer gewappnet war (Manz, 2019). Als *Brute-Force-Angriff* werden Angriffe bezeichnet, bei denen die Hacker durch Ausprobieren von vielen verschiedenen Zeichenkombinationen durch Zufall den Schlüssel finden möchten (Kaspersky, 2020). Kurze Schlüssel sind dabei natürlich anfälliger. Der AES unterstützt drei Schlüssellängen 128, 192 und 256 Bit und eine Blocklänge von 128

Bit. Daten die länger als diese 128 Bit des Blocks sind, müssen in mehrere Blöcke unterteilt werden (Manz, 2019).

Jedoch gibt es ein sehr großes Problem bei der symmetrischen Verschlüsselung. Wie findet die Übergabe des Schlüssels zwischen Ersteller und Empfänger statt? Der Schlüssel kann nicht einfach mit der Nachricht mitgesendet werden, da die Nachricht abgefangen werden kann. Die Nachricht könnte somit, von der Partei in der Mitte, gelesen und auch verändert werden (Elektronik-Kompendium, 2020d). Der Empfänger hätte keine Ahnung das der Text verändert wurde und würde dem geschriebenen Glauben schenken. Diese Art von Angriff, auch *Man-in-the-Middle-Angriff* genannt, ist sehr gefährlich. Beide Parteien, Sender und Empfänger, können von der Partei in der Mitte gelenkt werden. Um wieder auf den historischen Kontext zurückzukommen. Wenn eine Nachricht von Caesar an seine Truppen abgefangen wird und die abfangende Person den Schlüssel kennt, könnt diese den Truppen falsche Angaben machen und dadurch eine Schlacht entscheiden.
Um solche Angriffe zu verhindern, darf der Schlüssel nur den berechtigten Beteiligten bekannt sein. Es gibt verschiedene Möglichkeiten, wie der Austausch des Schlüssels stattfinden kann. Beispielsweise über Seitenkanäle, dabei wird der Schlüssel über einen anderen Weg als die Nachricht überliefert, oder der Schlüssel wird bei einem persönlichen Treffen direkt übergeben (Elektronik-Kompendium, 2020d). Eine andere Variante ist die asymmetrische Verschlüsselung, oder eine Kombination der beiden Techniken.

Asymmetrische Verschlüsselung

Im Vergleich zur symmetrischen Verschlüsselung werden bei dieser Art, zwei sich ergänzende Schlüssel verwendet. Das Verfahren wird auch oft als *Public-Key-Verfahren* bezeichnet, da einer der Schlüssel öffentlich und der andere privat ist. Das Vorgehen bei dieser Methode sieht wie folgt aus. Damit eine Person z.B. eine verschlüsselte Nachricht an einen Freund schicken kann, muss der Sender den öffentlichen Schlüssel des Empfängers haben. Mit diesem *Public Key* wird der Text der Nachricht chiffriert und an den Empfänger gesendet. Dieser kann nun mit seinem *private Key* die Nachricht entschlüsseln. Die Verbindung zwischen Public und Private Key wird durch einen mathematischen Algorithmus vorgenommen (Elektronik-Kompendium, 2020a). Wichtig dabei ist, dass der öffentliche Key, wie der Name schon sagt, der Öffentlichkeit zugänglich gemacht wird. Problematisch wird es, wenn ein falscher public Key im Umlauf ist. Die Verschlüsselung einer Nachricht ist dabei möglich, die Entschlüsselung jedoch nicht. Das wichtigste ist jedoch, dass der private Schlüssel nicht veröffentlicht wird, da sonst alle verschlüsselten Daten entschlüsselt werden können. Eine Problem der asymmetrischen Methoden ist, dass dafür um einiges mehr an Rechenleistung gebraucht wird, als für die symmetrischen Varianten. Dadurch arbeiten diese Verschlüsselungen um einiges langsamer. Teilweise sogar um den Faktor 1000 langsamer (Elektronik-Kompendium, 2020a). Ein Einsatz bei sehr große Dateien ist deswegen problematisch.

Die bekannteste asymmetrische Methode, ist das RSA-Verfahren. Dieses Kryptoverfahren wurde von Ronald Rivest, Adi Shamir und Leonard Adleman 1977 entwickelt und wird hauptsächlich für die Verschlüsselung von kleinen Datenmengen und für die Erstellung von digitalen Signaturen verwendet (Paar & Pelzl, 2016). Das System hinter der Verschlüsselung ist dabei recht einfach. Die Sicherheit hinter RSA ist die, dass es extrem schwierig ist aus einem Produkt zweier großer Primzahlen, diese wieder zu rekonstruieren (Elektronik-Kompendium, 2020c). Diese Einwegfunktion, eine Funktion die sehr schwer umzukehren ist, wird auch Primfaktorzerlegung genannt (Havenstein, 2007). Deswegen werden für RSA zwei zufällige geheime Primzahlen ausgewählt. Mithilfe des Produkts der beiden Primzahlen und zweier Exponenten, einem privaten Exponenten und einem öffentlichen, werden die Ver- und Entschlüsselungsgleichungen durchgeführt.

RSA stellt eine sehr gute Lösung dar, um kleine Daten zu verschlüsseln. Sobald jedoch die Datenmenge steigt, ist eine reine asymmetrische Verschlüsselung eine schlechte Option. Jedoch kann das Verfahren in Zusammenarbeit mit der symmetrischen Methode diesem Problem entgegenwirken. Dadurch würden die Schwachstellen beider Varianten ausgeglichen werden. Da die Verschlüsselung der Daten mithilfe des schnelleren Verfahrens (symmetrische Verschlüsselung) durchgeführt wird und der Schlüssel mit einer asymmetrischen Methode sicher übergeben werden kann.

4.2 Hash-Funktionen

Hash-Funktionen sind ein sehr wichtiges Werkzeug in der Kryptografie, diese Funktionen werden in vielen Bereichen als ergänzendes Mittel eingesetzt. Der Sinn einer Hash-Funktion ist der, dass ein beliebig langer Text oder sogar eine gesamte Datei in eine Zeichenkette mit festgelegter Länge umgewandelt wird. Da die Zeichenkette, oder auch Hash genannt, für dieselbe Nachricht immer gleich ist, wird der Hash-Wert auch als digitaler Fingerabdruck bezeichnet (Paar & Pelzl, 2016). Damit eine Hash-Funktion für kryptografische Verfahren eingesetzt werden darf, gibt es strenge Regeln die erfüllt werden müssen. Elektronik-Kompendium (2020b) nennt folgende Anforderungen.

„

- *Eindeutigkeit: Eine identische Zeichenfolge muss zum selben Hash-Wert führen.*
- *Reversibilität: Der Hash-Wert darf nicht in die ursprüngliche Zeichenfolge zurückberechnet werden können.*
- *Kollisionsresistenz: Zwei unterschiedliche Zeichenfolgen dürfen nicht den gleichen Hash-Wert ergeben.*

“ (Elektronik-Kompendium, 2020b)

Es ist sehr wichtig, dass all diese Anforderungen von kryptografischen Hash-Funktionen erfüllt werden. Wenn z.B. ein Hash-Wert zur Authentisierung eingesetzt wird, darf es nicht möglich sein das eine andere Zeichenfolge den gleichen Hash-Wert hervorbringt. Das System wäre nicht mehr sicher. Das Beispiel der Authentisierung zeigt auch wieso die Eindeutigkeit einer Funktion gewährleistet sein muss, da eine Autorisierung nicht erteilt werden kann, wenn der Hash-Wert unterschiedlich ist. Damit der Hash-Wert einzigartig bleibt, müssen schon die kleinsten Veränderungen einen Unterschied im Ergebnis bewirken.
Listing 4.1 zeigt anhand eines Beispiels, wie eine kleine Änderung des ursprünglichen Textes den gesamten Hash-Wert verändert. Der Text ist hierbei mit der Hash-Funktion *SHA-224* codiert worden.

Listing 4.1: Darstellung der unterschiedlichen Hash-Werte beim Tausch eines Satzzeichens.

Diese Nachricht soll gehasht werden. `wird zu`
df2a6a19f76e5f74c826fb5bfaef30cb94d2fbbdb75e61ac04ac48fe

Diese Nachricht soll gehasht werden! `wird zu`
0460d7c243e5cb1ea58b34c30aae4bf10c5e141de452310e103e2ff6

Diese Eigenheiten macht die Hash-Methode so wertvoll für Techniken wie die digitale Signatur und die Blockchain. Da bereits die kleinsten Abweichungen Veränderungen hervorrufen und somit manipulierte bzw. veränderte Daten auffallen würden. Dadurch ist auch ein Einsatz von Hash-Werten bei der Umsetzung des digitalen Planstempel sinnvoll.

4.3 Digitale Signatur

4.3.1 Allgemein

Schon recht frühen haben die Menschen bestimmte Schriftstücke oder Bilder mit einer Art Signatur versehen (DocuSign, 2017). Dabei unterschieden sich die Methoden wie signiert wurde. Eine beliebte Methode im Mittelalter war das Siegel (Krauth, 2018). Eine weniger eindrucksvolle Variante, war die normale Unterschrift. Dabei setzte der Unterzeichnende seinen Namen auf ein Bild oder ein Schriftstück um die Richtigkeit und Zugehörigkeit dessen zu bezeugen (DocuSign, 2017). Die Unterschrift hat sich bis in die heutige Zeit gehalten und wird immer noch verwendet um Verträge zu bestätigen. Doch auch für die Unterschrift, welche sich so lange gehalten hat, werden alternativen benötigt. Da die Digitalisierung voranschreitet und viele Geschäftsprozesse am besten nur noch digital stattfinden sollten, werden neue Möglichkeiten benötigt wie eine Signatur durchgeführt werden kann. Möglich ist dies unter anderem durch eine digitale Version der Unterschrift oder durch eine digitale Signatur. Wobei beide Varianten ziemlich ähnlich sind (Paar & Pelzl, 2016).

> *„Digitale Signaturen sind eines der wichtigsten kryptografischen Werkzeuge, die heutzutage zum Einsatz kommen.“* (Paar & Pelzl, 2016, S.297)

Laut Paar & Pelzl (2016) haben diese wichtigen Werkzeuge ein breitgefächertes Einsatzgebiet. So können sie bei der Aktualisierung von Software oder bei der Signierung von bindenden Verträgen verwendet werden. In dieser Arbeit sollen Signaturen verwendet werden um zu gewährleisten, dass das Modell nur von der genannten Person geprüft wurde und diese somit die Verantwortung für die Genehmigung bzw. Prüfung trägt. Außerdem schützt eine digitale Signatur vor der Unehrlichkeit anderer. Bei Verhandlungen zwischen zwei Parteien, sind beide dadurch an ihr Wort gebunden.

Die allgemeine Funktionsweise einer digitalen Signatur ist dabei recht einfach. Wie bei der asymmetrischen Verschlüsselung gibt es einen öffentlichen und einen privaten Schlüssel. Im Vergleich zur Verschlüsselung wird bei einer Signatur das Verfahren umgekehrt angewendet. Das bedeutet, dass der Sender die Nachricht mit seinem privaten Schlüssel signiert und dem Empfänger die Nachricht und die Signatur zukommen lässt. Der Empfänger kann dann das erhaltene Paket, aus Nachricht und Signatur, mithilfe des öffentlichen Schlüssel kontrollieren (Paar & Pelzl, 2016). Jedoch besteht auch hier wieder das Problem der Leistungsfähigkeit der asymmetrischen Verschlüsselung. Das signieren großer Datenmengen würde viel Zeit in Anspruch nehmen und manche Signaturmethoden sind sogar auf die Textlänge einer kurzen E-Mail begrenzt. Um dieses Problem zu beheben gibt es verschiedene Ansätze. Die Verwendung von Hash-Werten ist dabei am vielversprechendsten (Paar & Pelzl, 2016). Wie im Abschnitt (4.2) zuvor bereits erläutert wurde, besitzt das Ergebnis einer Hash-Funktion immer die gleiche Länge, egal welche Größe die Daten haben. Somit kann die Signatur für den Hash-Wert der Datei erstellt werden. Um es noch etwas genauer zu beschreiben. Der Hash-Wert wird mit dem privaten Schlüssel des Senders codiert und zusammen mit den Daten an den Empfänger gesendet. Der Empfänger muss nun überprüfen, ob die Signatur wirklich von dieser bestimmten Person stammt. Dies ist möglich, in dem der Empfänger die gesendeten Daten mit dem gleichen Hash-Verfahren verschlüsselt und gleichzeitig die Signatur mit dem öffentlichen Key des Senders entschlüsselt. Sollte am Ende der gleiche Hash-Wert herauskommen, können die Daten nur von der Person stammen (DocuSign, 2020). Grafisch wird dieses Vorgehen in Abbildung 4.1 gezeigt.

4.3.2 Signatur Arten

Im folgenden Kapitel sollen mehrere der bekanntesten Verfahren zur Erstellung von digitalen Signaturen dargestellt werden.
Das meistgenutzte Signaturverfahren ist die RSA-Signatur. Auch im Bereich der Verschlüsselung zählt es zu den beliebtesten Methoden (Paar & Pelzl, 2016). Da die Methode im Kapitel 4.1.2 bereits erläutert wurde, wird in diesem Abschnitt darauf verzichtet. Zu beach-

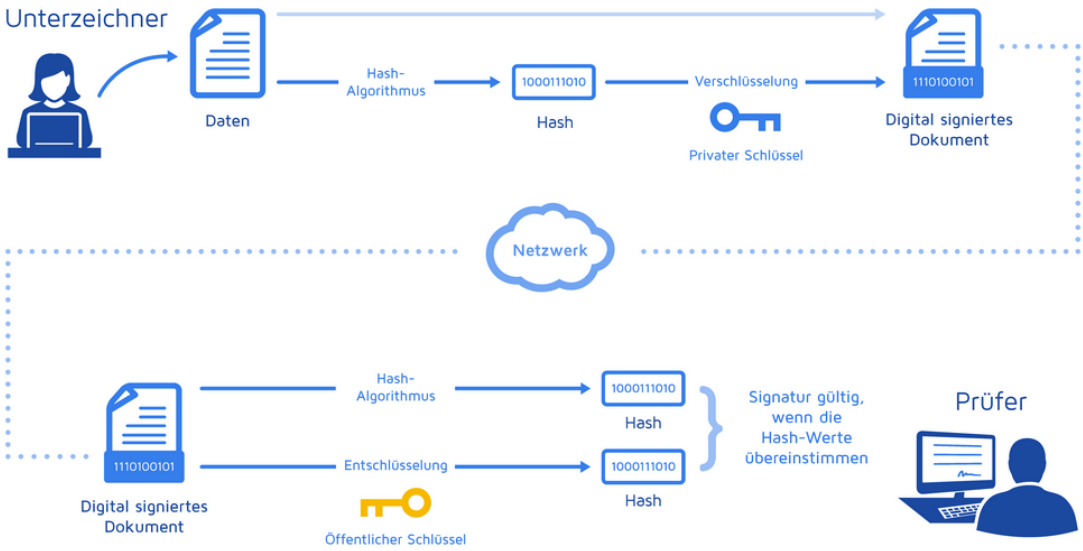

Abbildung 4.1: Erstellung einer digitalen Signatur. (DocuSign, 2020)

ten ist dabei nur, dass das RSA-Verfahren bei der Signatur umgekehrt angewendet wird. RSA ist eines der Signaturverfahren, das nur für begrenzte Datengröße benutzt werden kann. Die mögliche Datengröße liegt meist zwischen 128 und 384 Byte (Paar & Pelzl, 2016). Deswegen ist die Verwendung von Hash-Werten sinnvoll.

Bei den folgenden Verfahren handelt es sich um das Elgamal-Signaturschema und dem Digital Signature Algorithm (DSA). Wie das RSA-Verfahren, benötigen auch diese Methoden eine Einwegfunktion, um eine sichere Signatur zu erstellen. Die Methoden basieren deswegen auf dem diskreten Logarithmusproblem. Beutelspacher *et al.* (2010) erläutert das Problem wie folgt.

> *„Gegeben seien eine Primzahl p und zwei ganze Zahlen g, y. Gesucht ist eine ganze Zahl x mit der Eigenschaft* $g^x \mod p = y$. *Gesucht ist also der Logarithmus von y zur Basis g, allerdings nicht über den reellen Zahlen, sondern modulo einer Primzahl, daher auch der Name diskreter Logarithmus.“* (Beutelspacher *et al.*, 2010, S.132)

Dabei wird das Elgamal-Verfahren, das einfachere von beiden, nicht so häufig in der Praxis eingesetzt. Der DSA punktete dabei mit einer kürzeren Signaturlänge und dem Widerstand gegenüber bestimmten Angriffen. Das Elgamal-Verfahren ist unter anderem folgenden Angriffen ausgesetzt. Für Elgamal können gültige Signaturen für zufällige Nachrichten erzeugt werden. Das bedeutet, dass ein Angreifer eine Signatur erstellen kann, die vom Empfänger als gültig anerkannt wird, die erstellte Nachricht dabei jedoch keinen Sinn ergibt. Außerdem kann der private Schlüssel berechnet werden, falls der temporäre Schlüssel häufiger verwendet wird. Damit ist das Verfahren hinfällig. Um diesen Angriffen zu entgehen, wird häufiger das DSA-Verfahren eingesetzt (Paar & Pelzl, 2016).

4.4 Blockchain

4.4.1 Allgemein

Blockchain wird von vielen als die aufregendste technologische Entwicklung der letzten 20 Jahre bezeichnet. Wobei erste Grundlagen bereits in den 1990er Jahren erforscht wurden, erfolgte die erste echte Implementierung erst viele Jahre später (SAS, 2018). 2008 wurde das Whitepaper zu Bitcoin von Satoshi Nakamoto veröffentlicht, in dem die technischen Grundlagen dieser Kryptowährung erläutert wurden (bitcoin.de, 2020). In den folgenden Jahren wurden immer mehr Kryptowährungen entwickelt. Viele basieren dabei auf der Codebasis von Bitcoin oder benutzen die gleiche Blockchain (SAS, 2018). Durch die Zunahme an verschiedenen digitalen Währungen und der Popularität von Bitcoin, wurde auch die Technologie dahinter bekannter. Die Einsatzmöglichkeiten der Blockchain-Technologie sind dabei nicht nur auf Kryptowährungen beschränkt. Durch die Verbindung zu Bitcoin wurde jedoch für viele Menschen die Verknüpfung von Blockchain und finanziellen Transaktionen geschlossen. Dabei gibt es sehr viele verschiedene Einsatzgebiete.

Bevor jedoch die einzelnen Einsatzgebiete genannt werden, soll zuvor noch die grundsätzliche Funktionalität beschrieben werden. Chowdhury (2020) definiert die Blockchain wie folgt.

> *„A blockchain is an immutable distributed ledger secured by cryptographic techniques and managed by a decentralised community over a peer-to-peer network through incentivisation.“* (Chowdhury, 2020, S.8)

Um die Definition klarer zu machen, müssen die einzelnen genannten Attribute aufgeschlüsselt werden. Oft wird „distributed ledger“ als eine Art Synonym für den Namen Blockchain verwendet. Die Bezeichnung ergibt sich dabei aus dem Vergleich zu einem Register für Transaktionen, welches früher in der Buchhaltung verwendet wurde. Dabei werden die Transaktionen nicht nur in einem dieser Ledger gespeichert, sondern in vielen verteilten (Chowdhury, 2020). Eine Blockchain kann man sich wie eine große Datenbank vorstellen, die aus einzelnen Blöcken besteht. Jeder Block besteht dabei aus den gleichen Attributen. Ein Block beinhaltet hierbei immer den Hash des vorangegangenen Blocks, die gespeicherten Daten und den eigenen Hash-Wert (Klotz, 2016). In Abbildung 4.2 wird dies grafisch dargestellt. Sollte nun jemand den Inhalt eines Block verändern, würde sich damit auch der Hash-Wert ändern und somit nicht mehr zum Hash-Wert des nächsten Blocks passen. Natürlich wäre es nun möglich einfach alle Blöcke zu manipulieren, hierbei kommt jedoch das „distributed ledger“-Prinzip zum Einsatz. Der Zustand einer Blockchain wird auf vielen verschiedenen, verteilten Systemen gespeichert, somit müssten mehr als 50% der Systeme auf den gewünschten Zustand geändert werden. Deswegen hat Chowdhury (2020) in seiner Definition die Blockchain als „immutable“ (englisch für unveränderlich) beschrieben. Damit eine neue Transaktion, ein neuer Block, an die Kette angehängt werden kann, müssen die sogenannten *Miner* die Transaktion bestätigten

und hinzufügen. Welcher *Miner* nun genau den neuen Block erstellen darf, wird mithilfe von Konsensmechanismen (Consensus-Modell) festgelegt (Lang & Karlstetter, 2017). Dabei gibt es verschiedene Methoden, wie dieser ermittelt wird. Einer der ersten *Consensus*-Algorithmen ist der Proof of Work (POW), welcher auch bei Bitcoin eingesetzt wird (Chowdhury, 2020). Dabei müssen die *Miner* ein kryptografisches Rätsel lösen (Lang & Karlstetter, 2017). Wenn der neue Block bestätigt wurde, aktualisiert sich die Blockchain eines jeden Systems (Chowdhury, 2020).

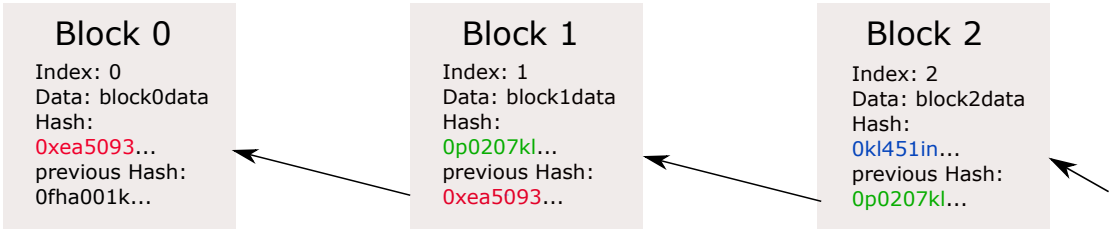

Abbildung 4.2: Beispielhafte Darstellung mehrerer Blockchain-Blöcke. (In Anlehnung an Hartikka (2017))

Da eine Speicherung von vielen verschiedenen Informationen möglich ist, kann die Blockchain-Technologie breit gefächert eingesetzt werden. Die Technik ist somit nicht nur auf die Finanzindustrie beschränkt. Zum Beispiel hat das US-amerikanische Unternehmen IBM im Jahre 2018 ein System zur Nachverfolgung von Lebensmitteln entwickelt. Dieses Netzwerk basiert auf der Blockchain (Rau, 2019). Auch in der Baubranche gibt es ein großes Anwendungsfeld für die Blockchain-Technologie. In der Veröffentlichung von Ye *et al.* (2018) werden verschiedene Einsatzmöglichkeiten beleuchtet. Unter anderem könnten mithilfe von *smart contracts* bestimmte Aktionen automatisiert durchgeführt werden. Der *smart contract* stellt einen Vertrag dar, der automatisch Ausgeführt wird, wenn festgelegte Bedingungen erfüllt werden. Der Kontrakt kommuniziert dabei mit der Blockchain (Klotz, 2016). Laut Ye *et al.* (2018) sind verspätete Zahlungen ein großes Problem der Bauindustrie, welche unter anderem auch zu verringerter Arbeitseffizienz führt. Um dem entgegenzuwirken können smart contracts eingesetzt werden. Dabei soll die Zahlungen automatisiert durchgeführt werden, sobald die Arbeit abgeschlossen ist (Ye *et al.*, 2018).

Dies sollte nur ein kleiner Überblick, über die möglichen Einsatzgebiete einer Blockchain sein. Für welche Anwendungen diese Technik verwendet werden kann, hängt auch stark von der Art der Blockchain ab. Im nächsten Kapitel werden die verschiedenen Arten beleuchtet.

4.4.2 Blockchain Arten

Public Blockchain

Die Beschreibungen des letzten Kapitels treffen hauptsächlich auf die öffentlichen (engl. public) Blockchains zu. Eine public Blockchain ist ein Netzwerk, das für jeden zugänglich ist. Somit kann jeder Beteiligte eine Transaktion erstellen, die Kette der Blöcke betrachten und mithilfe von bestimmten Methoden neue Blöcke erstellen. Die bekannteste öffentliche Blockchain wurde bereits genannt, nämlich Bitcoin. Eine weitere Kryptowährung, die auf eine öffentliche Methode setzt, ist Ethereum (Schiller, 2019). Die Besonderheit dieses Systems ist, dass Smart Contracts erstellt werden können. Ein großer Vorteil einer öffentlichen Blockchain ist die Dezentralisierung. Die Speicherung von Daten wird nicht zentral von einem Institut geregelt, sondern ist auf alle Beteiligten verteilt (SelfKey, 2020). Dadurch gibt es nicht den einen Angriffspunkt.

Private Blockchain

Im Vergleich zur öffentlichen Blockchain ist die private nur für bestimmte Parteien zugänglich. Verantwortlich für diese Blockchain ist meist nur eine Instanz (Binance, 2020). Deswegen eignet sich diese Art sehr gut für Unternehmen (Schiller, 2019). Diese bestimmte Partei kann unter anderem festlegen, welche Rechte andere Teilnehmer erhalten. Somit können spezielle Nutzer das Recht erhalten Transaktionen durchzuführen (Schiller, 2019). Sollten die Daten auch für andere User freigegeben werden, ohne dass diese neue Transaktionen tätigen, kann die verantwortliche Instanz bestimmten Nutzern ein Leserecht erteilen. Der Hauptvorteil einer privaten Blockchain ist die Geschwindigkeit. Da im Vergleich zur öffentlichen Kette, die Anzahl an Verantwortlichen um einiges verringert ist, wird weniger Zeit benötigt um einen Konsens zu erreichen. Der Grund für diesen Vorteil ist aber gleichzeitig auch der Grund für einen großen Nachteil. Aufgrund der geringen Anzahl von Verantwortlichen, wird die Idee eines großen dezentralen Systems reduziert. Die Blockchain wird somit zentraler. Dadurch leidet unter anderem auch die Sicherheit der Kette, da nur ein Knoten übernommen werden muss, um die Blockchain zu kontrollieren (SelfKey, 2020). Zu den bekanntesten privaten Blockchains zählen *Ripple* und *Hyperledger* (Schiller, 2019).

Konsortium Blockchain

Die Konsortium Blockchain ist eine erweiterte Version der privaten und öffentlichen Blockchain. Die Entscheidungsgewalt liegt hierbei nicht bei einem Verantwortlichen, sonder wird von mehreren Parteien getragen. Das Konsortium kann aus mehreren Unternehmen, Personen oder Instituten bestehen. Dabei werden die Vorteile beider Varianten (öffentlich und privat)

vereint (Binance, 2020). Somit sind schnelle Abwicklungen der Transaktionen möglich und die Sicherheit ist durch die Aufteilung auf mehrere Entscheidungsträger erhöht. Sinnvoll wäre es, wenn das Konsortium aus Parteien der gleichen Branche bestehen würde. Zum Beispiel könnten sich viele Finanzinstitute zusammenschließen, um eine gemeinsame Blockchain zu verwalten. Der Konsens könnte somit erzielt werden, wenn die Mehrzahl der Institute einer Entscheidung zustimmt. Beispiele für eine Konsortium Blockchain sind *R3* und *EWF* (Schiller, 2019).

Welche Variante von diesen drei nun die beste ist, lässt sich schwer sagen. Für jede Einsatzmöglichkeit muss abgewogen werden, welches Gesamtkonzept für dieses Anwendungsbeispiel am meisten Sinn macht.

Kapitel 5

Umsetzung der Methode des digitalen Planstempels

In diesem Kapitel soll die Methode des digitalen Planstempels genauer erläutert werden. Dabei wird zuerst ein allgemeiner Überblick zu diesem Konzept geliefert. Das Unterkapitel 5.1 und die Abschnitte von 5.2, die eine Beispielhafte Implementierung der Methode umfassen, sind dabei ähnlich aufgebaut. In beiden Fällen, werden die wichtigen Eckpfeiler der Methode nacheinander erläutert. In 5.1 wird dabei eher auf die allgemeine Funktionsweise eingegangen. Der technische Teil der Umsetzung wird mit der Auflistung der verwendeten Software gestartet (5.2.1). Im Unterkapitel (5.2.2) werden die Schritte zur Erstellung einer Schablone dargestellt. Dabei wird unter anderem mithilfe von Codeausschnitten erläutert, wie sich die Daten zum Erstellen der Schablone aus dem Modell herausfiltern lassen. Im Anschluss (5.2.3) wird beschrieben, wie sichergegangen werden kann, dass keine Verfälschung der Gebäudemodelldaten möglich ist. Dabei wird gezeigt, wie die ausgewählte Variante aus Abschnitt 5.1.2, bei der Implementierung der Methode eingesetzt wird. Die Schablone soll neben dem Festhalten des Gebäudestands auch dazu verwendet werden, um Vergleiche zwischen dem geprüften und einem anderen Modell durchzuführen. Im vorletzten Abschnitt (5.2.4) werden die Vergleichsvorgänge aufgezeigt. Der Abschluss (5.2.5) beschäftigt sich damit, wie die Implementierung und somit auch die Schablone erweitert werden kann.

5.1 Erläuterung des Konzepts

Das grundlegende Konzept des digitalen Planstempels ist recht einfach erklärt. Es soll eine Methode entwickelt werden, die bei der Prüfung bzw. Genehmigung von Bauwerken unterstützend eingesetzt werden kann. Als erster Schritt muss vom Prüfer das geprüfte Gebäudemodell freigegeben werden, wodurch eine Art Schablone für genau den Stand des

Modells erstellt wird und die prüfungsrelevanten Daten signiert werden. Die Schablone kann im Anschluss dazu verwendet werden, um aktuellere Versionen des Modells mit der geprüften Variante zu vergleichen. Diese kurze Erläuterung soll nun in den folgenden Unterabschnitten genauer erklärt werden.

5.1.1 Erstellen der Schablone

Damit ein Vergleich von verschiedenen Modellen möglich ist, müssen zuerst Schablonen der geprüften Bauwerke angefertigt werden. Die Schablone soll genau den Stand des Modells bei der Prüfung festhalten. Die Schablonen werden als eine JavaScript Object Notation (JSON)-Datei gespeichert. JSON ist ein Datenaustauschformat, das unabhängig von Programmiersprachen verwendet werden kann. Diese Textdatei wird dabei durch *Namen/Wert* Paare aufgebaut. Der *Name* stellt dabei eine Art Schlüssel dar. Mithilfe dessen kann der Inhalt des zugehörigen *Werts* abgerufen werden. Als *Wert* können verschiedene Inhalte festgelegt werden. So ist es möglich einen ganzen normalen String als Wert festzulegen oder durch ein Objekt die Struktur zu erweitern (JSON, 2020). Die daraus entstehende Baumstruktur und die Verwendung von Objekten macht dieses Datenformat zu einer optimalen Speichermöglichkeit der Schablone.

Die Struktur der Schablone soll, durch die Gliederung des Modells in Geschosse und Bauteile, sehr übersichtlich sein. Die Struktur der JSON-Datei würde somit ungefähr wie die des Listings 5.1 aussehen.

Listing 5.1: Grober struktureller Aufbau der Modellschablone.

```
Gesamtgebäude
|--- Geschoss 1
        |--- Wand 1
                |--- Eigenschaft 1
                |--- Eigenschaft 2
                |--- Tür 1
                        |--- Eigenschaft 1
                        |--- Eigenschaft 2

                |--- Fenster 1
                        |--- Eigenschaft 1
                        |--- Eigenschaft 2

        |--- Wand 2
|--- Geschoss 2
```

Das Listing zeigt dabei eine vereinfachte Darstellung der endgültigen Schablone. Die einzelnen Überpunkte besitzen noch einen Hash-Wert und weitere Attribute, die im Laufe des Kapitels

genauer beleuchtet werden. Als Bauteile werden für den Zweck der Arbeit nur Wände, Fenster und Türen berücksichtigt. Damit nun der Stand des Modells festgehalten werden kann und verschiedene IFC-Dateien verglichen werden können, muss den jeweiligen Oberpunkten der JSON-Datei ein Hash-Wert zugeordnet werden. Der Wert besteht dabei aus dem Inhalt der zugehörigen Unterpunkte. Zum Beispiel würde für den Hash-Wert der Wand 1 aus Listing 5.1, die Namen/Wert-Paare von den darin verbauten Fenstern und Türen, sowie den zugeordneten Eigenschaften, codiert werden. In Listing 5.2 wird ein Teil einer JSON-Datei dargestellt. Dabei werden nur die Strukturebenen von Gebäude, erstem Stockwerk (Storey), erster Wand und dessen Eigenschaften (Properties) aufgezeigt.

Listing 5.2: Teildarstellung einer JSON-Datei. Dargestellt werden die Abstufungen von Gebäude zu erstem Stockwerk (Storey) zu einer Wand des Stockwerks und dessen Eigenschaften (Properties).

```
1  "BuildingHash": "230e1c7e787a7ea64960f0a999b64e4e..." ,
2  "AllStoreys": [
3    {
4      "StoreyHash": "de00c86d0e3f616a50bb0205bc468ba0c13a7..." ,
5      "StoreyName": "'Erdgeschoss'" ,
6      "AllWalls": [
7        {
8          "WallHash": "c8563b9fdd597c10524a0f5e461a8b6ef89..." ,
9          "GlobalId": "'1bzfVsJqn8De5PukCrqylz'" ,
10         "StartPoint": {
11           "PointHash": "FAE2B750AAD41970F41B62551424BAE7",
12           "X": "0,00",
13           "Y": "9,85",
14           "Z": "0,00"
15         },
16         "EndPoint": {
17           "PointHash": "b1711fa241c9320404989ec339a3c7f6",
18           "X": "12",
19           "Y": "9,85",
20           "Z": "0"
21         },
22         "AllProperties": [
23           {
24             "PropertyHash": "AC8121293DA64124812AF536C14FE72C",
25             "PropertyName": "Height",
26             "PropertyValue": "2,7"
27           },
```

In dieser gekürzten Auflistung fehlen die verbauten Elemente der Wand, wie Türen und Fenster. Beide Bauteile werden mit ähnlichen Attributen wie bei einer Wand aufgelistet. Nur der Endpunkt ist bei Fenster und Tür nicht aufgeführt. Die Hash-Werte werden mit

unterschiedlichen Verfahren ermittelt, deswegen sind manche großgeschrieben und manche kleingeschrieben.

Durch die hierarchische Struktur der JSON-Datei, sind Vergleiche von verschiedenen Modellen gut durchführbar. Dafür wird für beide IFC-Dateien eine extra Schablone erstellt. Da bereits die kleinsten Abweichungen einen anderen Hash-Wert hervorrufen (s. Abschnitt 4.2), muss schon bei der Erstellung der Schablone darauf geachtet werden, dass der Inhalt und der Aufbau der JSON-Datei gleiche Strukturen aufweisen. Zum Beispiel kann es vorkommen, dass Punktkoordinaten und geometrische Abmessungen mit vielen Nachkommastellen festgehalten werden. Damit keine Abweichung des Hash-Wertes durch eine Zahl z.B. an der fünften Nachkommastelle entsteht, werden diese Werte gerundet. Auch die Auflistungen in der JSON-Datei können zu Fehlern führen. Durch eine unterschiedliche Anordnung, der eigentlich übereinstimmenden Attribute, wird ein komplett veränderter Hash-Wert erstellt. Um dem entgegen zu wirken, werden die Geschosse (Storeys) und die Wandeigenschaften (Properties) alphabetisch nach dem Objektnamen sortiert. Ähnliches passiert bei der Sortierung der Wände eines Stockwerks oder der Fenster und Türen einer Wand. Diese werden jedoch nicht nach dem Namen gegliedert, sondern über den Startpunkt (StartPoint) geordnet. Dabei werden die Wände ausgehenden vom X-Wert der Wand aufsteigend sortiert. Sollten zwei Wände den gleichen X-Wert besitzen, wird die Wand mit dem größeren Y-Wert zu erst gelistet. Durch diese Maßnahmen sollte gewährleistet sein, dass Abweichungen der Hash-Werte nur noch durch veränderte Modelleigenschaften entstehen.

Damit der digitale Planstempel auch von Fachprüfern verwendet werden kann, muss es möglich sein die Bauteile des Modells nach bestimmten Kriterien in die Schablone einfließen zu lassen. Ein Prüfstatiker möchte z.B. nur die tragwerksrelevanten Bauteile des Bauwerks festhalten oder der Prüfer für Brandschutz nur die brandschutzrelevanten. Somit kann die Schablone je nach Notwendigkeit angepasst werden.

5.1.2 Gewährleisten der Datenintegrität

Die Datenintegrität kann durch verschiedene Mittel gewährleistet werden. In dieser Arbeit wurden bereits zwei dieser Methoden beschrieben, nämlich *Blockchain* und *digitale Signaturen*. Obwohl sich beide Varianten gut eignen die Verfälschung von Daten zu erkennen, wird bei der beispielhaften Implementierung der Methode des digitalen Planstempels auf die digitalen Signaturen gesetzt. Bevor jedoch genauer auf die Signatur eingegangen wird, soll zuerst noch die nicht verwendete Blochchain Variante erläutert werden.

Der Vorteil der Blockchain-Technologie ist, dass diese Technik zusätzlich zur Gewährleistung der Datenintegrität auch als Speichermöglichkeit eingesetzt werden kann. Beteiligte des Prüfungsprozesses erhalten dabei Zugang zur Blockchain. Im Falle einer Baugenehmigung bekommen die Baubehörde und die Gemeinde bestimmte Rechte. Die Baubehörde kann eine

Genehmigung durchführen und somit eine Transaktion erstellen, diese muss von einem Konsortium, aus anderen Baubehörden, bestätigt werden und die Transaktionen wird somit als neuer Block an die Kette angehängt. Der Block beinhaltet die gesamten Baugenehmigungsdaten und das Gebäudemodell. Dadurch ist eine Manipulation der Daten ausgeschlossen und die zuständigen Prüfer können jederzeit ermittelt werden. Somit entfällt auch der Sinn einer getrennten Speicherung durch Gemeinde und Baubehörde. Die Gemeinden können durch ein Leserecht Zugang zu den genehmigten Bauwerken in deren Gebiet erhalten. Die Umsetzung dieser Idee wäre jedoch sehr aufwendig und würde eher den gesamten Genehmigungs- bzw. Prüfungsprozess betreffen und nicht nur den kleinen Aufgabenbereich, der in dieser Arbeit betrachtet werden soll.

Unter anderem deswegen wird in dieser Arbeit die Authentizität der Daten mithilfe einer digitalen Signatur gewährleistet. In Abschnitt 4.3 wurde bereits die allgemeine Funktionsweise der Signatur erläutert. In diesem Absatz wird dies noch einmal aufgegriffen und die Verwendung beim digitalen Planstempel erklärt.
Die digitale Signatur soll für die geprüften Gebäudedaten und Prüfungsdaten erstellt werden. Somit ergibt sich ein Datenpaket aus Gebäudemodell, Schablone des Modells, Prüfername und dem Zeitpunkt der Prüfung. Wie bereits im Abschnitt 4.3 erwähnt wurde, gibt es Signaturarten, die nur eine begrenzte Datengröße signieren können. Um dieses Problem zu vermeiden, werden die Daten vor dem Signieren gehasht. Im Anschluss wird für den Hash-Wert eine Signatur erstellt. Das bedeutet, sobald eine Person das betrachtete Gebäudemodell als geprüft ansieht und dies im Programm bestätigt, wird mit dem privaten Schlüssel dieser Person eine digitale Signatur angefertigt und in Kombination mit den geprüften Daten gespeichert und weitergeleitet. Sollte nun bestimmt werden, wer dieses Modell genehmigt hat, muss das Paket aus Signatur und Prüfungsdaten analysiert werden. Aus der Signatur lässt sich mit dem öffentlichen Schlüssel des Prüfers, der ursprüngliche Hash-Wert ermitteln. Damit die Verifizierung abgeschlossen werden kann, muss die selbe Hash-Funktion auf die mitgelieferten Prüfungsdaten angewendet werden. Stimmen Hash-Wert der entschlüsselten Signatur und Hash der Datei überein, ist die Prüfung vom Besitzer des privaten Schlüssels durchgeführt worden. Die genannten Abläufe sind gut erkennbar in der Abbildung 4.1 dargestellt.

Zusätzlich wäre es noch möglich die geprüften Dateien zu verschlüsseln und danach erst die Signatur durchzuführen. Damit wäre es möglich zu verifizieren, dass die Dateien von dieser Person stammen ohne jedoch Zugang zum Inhalt zu erhalten.

5.1.3 Vergleich mit anderen Modellen

Wie bereits in Unterabschnitt 5.1.1 erwähnt wurde, wird für jede Ebene in der JSON-Datei ein Hash-Wert ermittelt. Durch vergleichen dieser Werte kann ermittelt werden, ob die identischen Daten gehasht wurden. Falls der Gebäudehash beider Modelle gleich ist, wird damit

bestätigt, dass seit der Genehmigung des Modells keine Änderungen an diesem durchgeführt wurden. Sollten die Werte nicht übereinstimmen, muss durch abgleichen der Hash-Werte auf einer der unteren Ebenen ermittelt werden, an welcher Stelle die Abweichungen entstanden sind. In den Gesamthash des Gebäudes fließen alle Stockwerke und dessen Bauteile, sowie die Eigenschaften dieser Bauteile mit ein. Falls nun der Hash-Wert auf oberster Ebene („BuildingHash“) von der anderen Schablone abweicht, müssen die einzelnen Stockwerke verglichen werden. Das praktische dabei ist, wenn der Hash-Wert eines Stockwerks mit dem des zu vergleichenden Modells übereinstimmt, fällt eine zusätzliche Betrachtung der Bauteile dieser Etage weg.
Sollte ein Stockwerk mit ungleichen Werten erkannt worden sein, muss der Ursprung dieser Veränderung gefunden werden. Dies ist möglich, indem die einzelnen Wände der Etage untersucht werden. Die Wände des geprüften Modells müssen dabei mit den Gegenstücken des neuen Gebäudemodells verglichen werden. Die übereinstimmenden Bauteile können dabei über die „GlobalId“ der Wand gefunden werden. Außerdem lassen sich, durch vergleichen der Bauteil-Ids, neu hinzugekommene Bauteile schnell ermitteln. Die Wände setzten sich aus einer GlobalId, einem Startpunkt, einem Endpunkt, verschiedenen Eigenschaften und den darin enthaltenen Bauteilen, wie Fenster und Türen, zusammen. Wird für eine Wand ein fehlerhafter Hash-Wert entdeckt, müssen die Attribute der Wand untersucht werden. Wird das abweichende Attribut gefunden, muss dies vermerkt werden. Die Suche nach weiteren Veränderungen wird durch einen Sprung auf die nächst höhere Ebene fortgesetzt. Abbildung 5.1 zeigt eine vereinfachte Version der gerade beschriebenen Vorgänge.

Durch die hierarchische Struktur der JSON-Datei ist auch ein Vergleich von partiell geprüften Gebäudemodellen möglich. Sollte bei einem Genehmigungsvorgang nur das Erdgeschoss genehmigt worden sein, muss bei zukünftigen Genehmigungen, für das selbe Gebäude, überprüft werden, ob Veränderungen in dem bereits genehmigten Erdgeschoss vorliegen. Möglich ist dies, indem der Vergleich der Hash-Werte auf der JSON-Ebene des gewünschten Stockwerks gestartet wird.

Die exemplarische Umsetzung dieses Konzepts und die damit verbunden Teilschritte werden im nächsten Abschnitt erläutert.

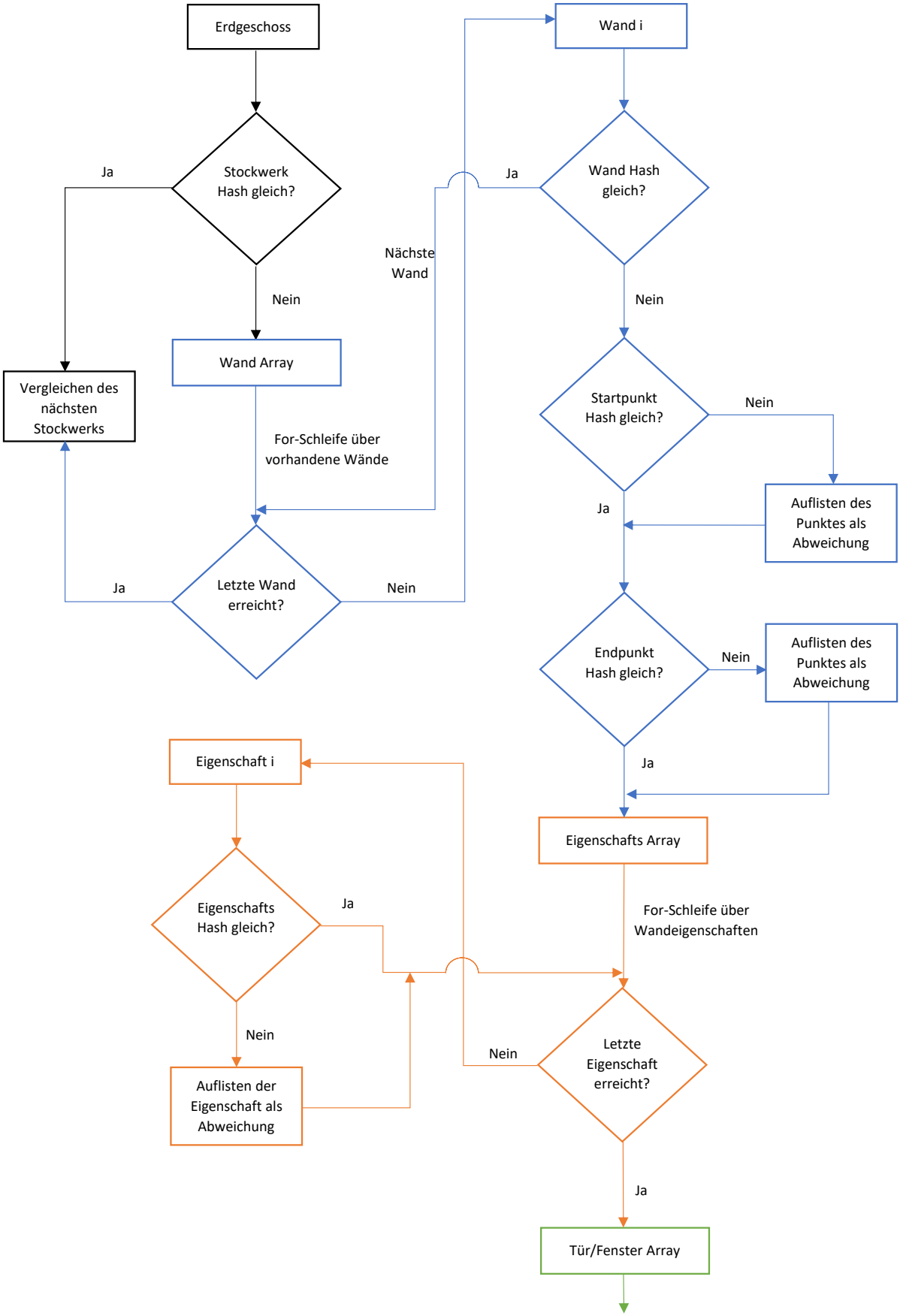

Abbildung 5.1: Vereinfachtes Flussdiagramm der Vergleichsvorgänge für die unterschiedlichen Modelle. Die Ermittlung von Abweichungen bei Türen und Fenstern ist ungefähr so aufgebaut wie bei den Wänden. Auf eine Darstellung ist deswegen verzichtet worden.

5.2 Beispielhafte Implementierung der Methode

Nachdem die konzeptionellen Grundlagen im Abschnitt zuvor gelegt wurden, soll in diesem die exemplarische Implementierung der Methode des digitalen Planstempels aufgezeigt werden. Bevor jedoch auf die genaue Umsetzung der wichtigen Funktionen eingegangen wird, liefert folgender Absatz einen kurzen Gesamtüberblick über die Anwendung.
Die Windowsanwendung ist auf das Nötigste, um die Methode umzusetzen, reduziert. Auf dem Startfenster kann eine Prüfung bestätigt werden. Durch Eingabe des Prüfernamens, dem Namen des Bauvorhabens, der Art der Prüfung und durch laden der IFC-Datei können die Daten bestätigt werden. Wird der Bestätigungsbutton gedrückt, wird die Modell-Datei in eine Graphdatenbank geladen und damit die Schablone des Bauwerks angefertigt. Alle notwendigen Daten zur Prüfung werden gespeichert und zusammen gehasht. Für den entstandenen Hash-Wert wird die digitale Signatur erstellt. Soll nun überprüft werden, ob Unterschiede zwischen der geprüften Datei und einer aktuelleren Version davon bestehen, muss nur die aktuelle Datei geladen werden. Dadurch wird für diese Version ebenfalls eine Schablone erstellt, wodurch sich beide Schablonen miteinander vergleichen lassen. Eine Kontrolle der Integrität der geprüften Daten kann mithilfe der digitalen Signatur durchgeführt werden. Dafür müssen wieder alle gespeicherten prüfungsrelevanten Daten gehasht und mit dem entschlüsselten Wert der Signatur verglichen werden. Stimmen beide Werte überein, steht einem Vergleich der Modelle nichts im Weg. Die Vergleiche werden auf den unterschiedlichen Ebenen der JSON-Datei durchgeführt. Treten Ungleichheiten bei Bauteilen auf, werden diese aufgelistet und lassen sich mithilfe einer Anwendung visualisieren.

Die Struktur des Abschnittes ähnelt dabei dem des vorangegangenen (5.1). Das Grundgerüst der Methode, Erstellen der Schablone (5.2.2), Gewährleisten der Datenintegrität (5.2.3) und Vergleich mit anderen Modellen (5.2.4), wird durch die Unterabschnitt „Verwendete Software“ und „Erweitern der Implementierung“ ergänzt. Zum Abschluss des gesamten Kapitels, wird das technische und grundlegende Verständnis für die Methode des digitalen Planstempels vorhanden sein, um das Konzept bei bestimmten Gebäudemodellen einzusetzen.

5.2.1 Verwendete Software

Um die technische Umsetzung des digitalen Planstempels durchzuführen, wurden einige zusätzliche Programme benötigt. In diesem Unterabschnitt werden die Programme aufgelistet und deren Einsatz bei der Umsetzung der Methode kurz beschrieben. Folgende aufgelistete Anwendungen wurden verwendet.

- Visual Studio 2017
- Neo4j

- xBIM WeXplorer

Visual Studio 2017

Visual Studio ist eine Integrated Development Environment (IDE), welche von Microsoft entwickelt wurde. Mithilfe dieser Entwicklungsumgebung können Anwendungen mit den verschiedensten Programmiersprachen entwickelt werden (Microsoft, 2019). In dieser Arbeit wurde Visual Studio zur Erstellung einer Windows Presentation Foundation (WPF)-Anwendung verwendet. Mit WPF ist es möglich, grafische Oberflächen für die Desktopanwendung zu erstellen. Im Falle der Implementierung des digitalen Planstempels wurden mehrere Seiten designt. Auf diesen Seiten können mithilfe von Buttons Funktionen ausgeführt werden. So ist es möglich die gewünschten IFC-Dateien zu laden, daraus die Schablonen anzufertigen und falls gewünscht, mit anderen Dateien zu vergleichen. Die Funktionen wurden dabei in der Programmiersprache C# programmiert.

Neo4j

Dieses Hilfsmittel wurde bereits im Kapitel 3 erwähnt. Es handelt sich dabei um eine Graphdatenbank. Wie der Name schon sagt, ist es möglich mithilfe dieser Anwendung einen Graph zu erstellen und die Knoten und Kanten des Graphen mit Informationen zu füllen. Durch die Abfragesprache *Cypher* können Knoten erzeugt, Knoten über Beziehungen verbunden werden oder Informationen aus dem Graphen gezogen werden (Neo4j, 2020b). In Unterabschnitt 3.3.2 wurde bereits auf die Verwendung von IFC-Dateien in Graphdatenbanken eingegangen. Neo4j kann dabei helfen die IFC-Datei nach Informationen zu filtern. Durch festgelegte Abfragen können Daten, wie die Länge einer Wand oder die Größe eines Fensters, für jedes einzelne Element herausgefunden werden. Deswegen wird diese Graphdatenbank hauptsächlich bei der Erstellung der Schablone (5.2.2) eingesetzt. In diesem Abschnitt werden unter anderem die verwendeten Abfragen dargestellt. Außerdem lassen sich mithilfe von Neo4j, die Verbindungen zwischen den verschiedenen IFC-Entitäten gut visualisieren. Die IFC-Datei wird mithilfe einer Implementierung des Lehrstuhls für Computergestützte Modellierung und Simulation der Technische Universität München in die Graphdatenbank geladen. Die Implementierung des Lehrstuhls ist grundsätzlich an den Parser von Ismail *et al.* (2017) angelehnt.

xBIM WeXplorer

WeXplorer stellt einen Web Viewer für IFC-Dateien dar und ist Teil des xBIM Toolkits. Mithilfe dessen können Bauwerke im Webbrowser betrachtet werden. Die IFC-Dateien können hierbei nicht direkt verwendet werden, sondern müssen in das wexBIM Format umgewandelt werden (xBIM, 2016). Mithilfe von WeXplorer können verschiedene Aktionen an dargestell-

ten Gebäuden durchgeführt werden. Diese Funktionen machen den Viewer interessant für die Darstellung von Abweichungen zwischen zwei Gebäudemodellen. Sollten beim Vergleichen Ungleichheiten auftreten, werden die betroffenen Bauteile farbig markiert. Näheres zum Einsatz des Visualisierungswerkzeugs kann im Unterabschnitt 5.2.4 gefunden werden.

Nachdem die Beschreibung der verwendeten Software abgeschlossen ist, soll in den anschließenden Unterabschnitten die Implementierung der einzelnen Funktionen erklärt werden.

5.2.2 Erstellen der Schablone

Gestartet wird die Erläuterung der technischen Umsetzung mit der Erstellung der Schablone, welche am Ende des Erstellungsvorgangs in einer JSON-Datei gespeichert wird. Der Aufbau des Unterabschnitts gliedert sich noch einmal in verschiedene Bereiche, damit genau erläutert werden kann, wie die verschiedenen Bauteile und Eigenschaften herausgefiltert wurden. Den Abschluss bildet dann die Zusammensetzung der einzelnen Informationen, um eine vollständige Schablone, mit Stockwerken und Bauteilen, zu erhalten.

Besonders wichtig ist, dass die Bauteile nur Stockwerksweise ermittelt werden dürfen, da auch eine partielle Prüfung der Gebäudemodelle möglich sein soll. Damit dies durchführbar ist, muss die hierarchische Struktur der JSON-Schablone ungefähr wie in Listing 5.1 aussehen. Der erste Schritt zur Anfertigung der Schablone ist, herauszufinden aus wie vielen Stockwerken ein Gebäude überhaupt besteht. Aufgrund solcher Abfragen wird die Graphdatenbank Neo4j verwendet. Folgendes Listing zeigt die Abfrage der Stockwerksnamen.

Listing 5.3: Abfrage aller im Gebäude vorhandenen Stockwerke und Ausgabe derer Namen in Neo4j.

```
MATCH(storey:IFCBUILDINGSTOREY) WHERE storey.model="templateModel"
RETURN storey.prop2
```

Das Property „prop2“ beinhaltet dabei die Namen der *IfcBuildingStorey*-Knoten. Als Ergebnis erhält man eine Liste mit allen Stockwerksnamen des Modells. Über die Verbindung *IfcRelContainedInSpatialStructure* zwischen *IfcBuildingStorey* und den abgeleiteten Entities von *IfcBuildingElement*, sowie dem Namen einer speziellen Etage, können alle Bauteile für dieses Geschoss mithilfe einer Query ermittelt werden. Die Abbildung 5.2 zeigt das Ergebnis dieser Abfrage.

Hinzufügen der Wände

Nachdem im Absatz zuvor alle Bauteile einer Etage ermittelt wurden, soll im folgenden beschrieben werden, wie sich die aufgelisteten Wände aus Abbildung 5.2 zur Schablone hinzufügen lassen. Dafür sind einige Arbeitsschritte notwendig um alle wichtigen Informationen

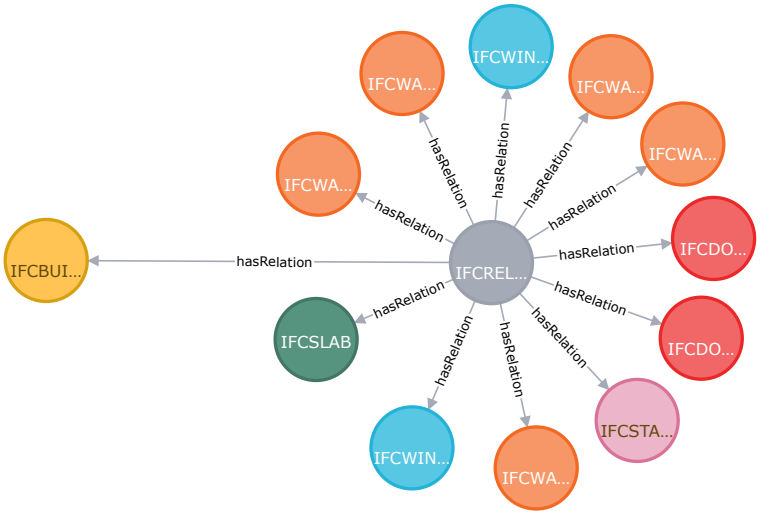

Abbildung 5.2: Darstellung aller Bauteile einer bestimmten Etage. Herstellen der Verbindung über IfcRelContainedInSpatialStructure (grau) und IfcBuildingStorey (gelb). Wände werden in orange, Fenster in blau, Türen in rot, Treppen in rosa und die Decke in grün dargestellt.

der Wand zu erhalten. Als erstes muss hierfür ermittelt werden, welche Nummern die Wände dieser bestimmten Etage besitzen. Codebeispiel 5.4 zeigt die Abfrage für diesen Vorgang.

Listing 5.4: Abfrage aller Wände der Etage „Level 1“ und Ausgabe derer Nummern in Neo4j.

```
MATCH(storey:IFCBUILDINGSTOREY)-[]-(:IFCRELCONTAINEDINSPATIALSTRUCTURE)
    -[]-(walls:IFCWALLSTANDARDCASE)
WHERE storey.model = "templateModel" AND storey.prop2 = "'Level 1'"
RETURN walls.EntityId
```

Das Ergebnis dieser Query, ist eine Liste mit Nummern aller Wände der Etage „Level 1“. In diesem Fall wurde der Name des Stockwerks und der Modellname direkt angegeben. Normalerweise werden die Namen über einen Platzhalter eingefügt, um mithilfe einer Schleife die Wandnummern für jedes Stockwerk zu ermitteln. Die ermittelten Wandnummern werden verwendet, um an alle notwendigen Informationen für diese Wandinstanz zu gelangen. Dazu zählen die GlobalId der Wand, Start- und Endpunkt der Wand, Eigenschaften, wie z.B. Höhe, Breite und Länge, sowie das Material. Damit jedoch alle gewünschten Eigenschaften berücksichtigt werden können, müssen die Namen der Eigenschaften festgelegt werden. Neben den bereits genannten Eigenschaften, sollen noch z.B. die Wärmedurchlässigkeit und die Angabe zur Tragfähigkeit hinzukommen. Folgender Codeausschnitt zeigt, wie das Material einer Wand ermittelt werden kann. Die Verknüpfung von Bauteilen und deren Materialien erfolgt in IFC, wie auch bereits in der Abbildung 5.2, mithilfe einer abgeleiteten Klasse von *IfcRelationship*. Um das Material zu erhalten, muss man somit über den Knoten *IFCRELASSOCIATESMATERIAL* gehen.

Im Gegensatz zu den zwei Listings zuvor, stellt das Codebeispiel 5.5 die Cypher Abfrage direkt in der Anwendung dar. Über die Verbindung zwischen der Neo4j-Datenbank und der WPF-

Listing 5.5: Ermitteln des Materials einer Wand in der Implementierung mithilfe der *Bolt-Connection* zur Datenbank und dem *Neo4j Driver*.

```
1 var result = tx.Run("MATCH(n1:IFCWALLSTANDARDCASE)-[]-(n2:
      IFCRELASSOCIATESMATERIAL)-[]-(n3:IFCMATERIALLAYERSETUSAGE)-[]-(n4:
      IFCMATERIALLAYERSET)-[]-(n5:IFCMATERIALLAYER)-[]-(n6:IFCMATERIAL) " +
2                         "WHERE n1.model = $ModelName AND n1.EntityId=$id " +
3                         "RETURN n6.prop0 AS material",
4 new { ModelName, id });
```

Anwendung können dort die Queries definiert und die Ergebnisse empfangen werden. Die Verbindung wird über einen *Bolt-Connector* erzeugt. Mithilfe von *Bolt* und dem Neo4j *Driver* für *.NET* lassen sich Anfragen an die Datenbank senden und Ergebnisse zurückbekommen (Bolt Protocol, 2020). Durch die Platzhalter *ModelName* und *id* kann das Material für beliebige Wände und Modelle herausgefunden werden. Nachdem das Material und die anderen Eigenschaften der Wand ermittelt worden sind, müssen noch der Start- und Endpunkt herausgefunden werden. Der Startpunkt kann über die Klasse *IfcLocalPlacement* herausgefunden werden. Durch die Verbindung zwischen *IfcWallStandardCase*, *IfcLocalPlacement* und *IfcAxis2Placement3D* kann der *IfcCartesianPoint* bestimmt werden, der die Koordinaten des Startpunktes der Wand beinhaltet. Der Endpunkt kann im Anschluss aus den Koordinaten des Startpunkts, sowie der Länge und dem Richtungsvektor der Wand berechnet werden.

Damit auch die einzelnen Eigenschaften einer Wand auf einen schnellen Blick verglichen werden können, muss auch jedes einzelne von diesen mit einem Hash-Wert versehen werden. Vereinfacht wird dieses Verfahren durch eine Funktion in Neo4j. Mithilfe von *apoc.hashing.fingerprint()* können Hash-Werte für Knoten, Beziehungen und sogar für den kompletten Graphen generiert werden (Neo4j, 2020a). Folgendes Codebeispiel zeigt, wie der Hash-Wert für das Material, mithilfe des Hash-Verfahrens Message-Digest Algorithm 5 (MD5), ermittelt werde kann.

Listing 5.6: Ermitteln des Hash-Wertes für die Verbindung von *IfcWallStandardCase* zu *IfcMaterial* in Neo4j.

```
MATCH(n1:IFCWALLSTANDARDCASE)-[]-(n2:IFCRELASSOCIATESMATERIAL)-[]-(n3:
    IFCMATERIALLAYERSETUSAGE)-[]-(n4:IFCMATERIALLAYERSET)-[]-(n5:
    IFCMATERIALLAYER)-[]-(n6:IFCMATERIAL)
WITH n1,n6
WHERE n1.model = "templateModel" AND n1.EntityId=33842
WITH collect([n1.name,n6.name, n6.prop0]) AS result
RETURN apoc.hashing.fingerprint(result) AS hash
```

Die Verbindung über die sechs Knoten kann dabei leider nicht komplett gehasht werden, da die Knoten je nach Stand der IFC-Datei andere Nummern besitzen und das Ergebnis des Hash-Verfahrens somit, trotz übereinstimmender geometrischer Eigenschaften, verändert

wird. Deswegen müssen die Attribute von *IfcMaterial* einzeln aufgeführt werden. Dieses Verfahren wird auch für die restlichen Eigenschaften der Wand verwendet. Somit kann die im Programm definiert Wandklasse (Listing 5.7) gefüllt werden.

Listing 5.7: In der WPF-Anwendung definierte Wall-Klasse.

```
1  public class Wall
2      {
3          public string WallHash { get; set; }
4          public string GlobalId {get; set;}
5          public CartesianPoint StartPoint { get; set; }
6          public CartesianPoint EndPoint { get; set; }
7          public List<Property> AllProperties { get; set; }
8          public List<Door> AllDoors { get; set; }
9          public List<Window> AllWindows { get; set; }
10     }
```

Das Codebeispiel 5.7 zeigt neben den bereits ermittelten Punkten und Properties, die noch leeren Auflistungen der in der Wand integrierten Bauteile. In den folgenden Ausführungen werden diese Elemente ermittelt und zur Wandklasse hinzugefügt.

Hinzufügen der Fenster und Türen

Das Ermitteln von Fenster und Türen einer Wand, besteht aus ungefähr den gleichen Arbeitsschritten wie im Absatz zuvor. Der erste Schritt, wie bereits in der „Hinzufügen einer Wand“-Passage, ist herauszufinden, wie viele Fenster und Türen in einer bestimmten Wand verbaut sind. Die Verbindung zwischen Wand und Tür/Fenster, wird in der IFC-Datei über die Klasse *IfcOpeningElement* hergestellt. Folgende Neo4j-Abfrage ermittelt alle Fensternummern für die Wand mit der Nummer (EntityId) 33842.

Listing 5.8: Ermittlung aller Fensternummern in der Wand 33842 in Neo4j.

```
MATCH(wall:IFCWALLSTANDARDCASE)-[]-(:IFCRELVOIDSELEMENT)-[]-(:
    IFCOPENINGELEMENT)-[]-(:IFCRELFILLSELEMENT)-[]-(element)
WHERE wall.model = "templateModel" AND wall.EntityId = 33842 AND element.
    name = " IFCWINDOW"
RETURN element.Entityid
```

Durch die Abfrage Bedingung von „element.name = “ können je nach Bedarf die Nummern der Türen oder Fenster ermittelt werden. In der Anwendung wird hierfür ein Platzhalter verwendet, um je nach Situation das spezifische Bauteil anzugeben.
Sobald alle Nummern der Türen/Fenster ermittelt worden sind, können nacheinander alle Elemente mit Informationen gefüllt werden. Die folgenden Arbeitsschritte gelten für Türen und Fenster. Damit sich die Eigenschaften der Bauteile herauslesen lassen, muss zuerst festgelegt

werden welche speziellen Eigenschaften überhaupt wichtig sind. Wie bereits bei den Wänden, fällt die Anzahl der zu betrachtenden Attribute eher gering aus. Für den Zweck der Arbeit werden nur die Eigenschaften Höhe, Breite, Material und Name des Bauteils berücksichtigt. Die Liste mit den gewünschten Eigenschaften kann jedoch je nach Belieben erweitert werden. Die geometrischen Abmessungen, wie Höhe und Breite, werden dabei direkt aus der IFC-Klasse des Bauteils herausgelesen. Im neunten bzw. zehnten Attribut der Klasse sind die Werte vermerkt (buildingSMART, 2020b). Eine Abfrage der Breite für ein spezifisches Fenster würde so aussehen:

Listing 5.9: Ermitteln der Breite eines Fensters in Neo4j.

```
MATCH(n1) WHERE n1.EntityId=32750 AND n1.model ="templateModel"
RETURN n1.prop9 AS value
```

Nachdem mit ähnlichen Abfragen die anderen Eigenschaften ermittelt wurden, müssen die Koordinaten der Bauteile ausfindig gemacht werden. Die Koordinaten werden dabei, ausgehend vom Nullpunkt des Stockwerks und dessen Ausrichtung, angegeben. Somit erhält man mit folgender Query lokale Koordinaten bezogen auf dieses Stockwerk.

Listing 5.10: Ermitteln der lokalen Koordinaten eines Fensters.

```
MATCH(n1)-[]-(n2:IFCLOCALPLACEMENT)-[]-(n3:IFCAXIS2PLACEMENT3D)-[]-(n4:
    IFCCARTESIANPOINT)
WHERE n1.model = "model3" AND n1.EntityId=23024
RETURN n4
```

Damit die Passage *Hinzufügen der Fenster und Türen* abgeschlossen werden kann, müssen noch die Hash-Werte der Bauteilattribute und der Wand ermittelt werden. Die Hash-Werte der Eigenschaften werden mit ähnlichen Queries, wie für die Ermittlung der jeweiligen Werte durchgeführt. Als Zusatz wird jedoch die bereits erwähnte Funktion *apoc.hashing.fingerprint()* benötigt. Für den Hash-Wert der Türen/Fenster bzw. der Wand muss aus den gesamten Bauteilinformationen ein JSON-String erstellt werden, welcher im Anschluss gehasht wird.

Dies bildet den Abschluss der Bauteilermittlung. Mithilfe der zuvor genannten Abfragen können alle Wände, Fenster und Türen in die Schablone übergeben werden. Somit ist es möglich Vergleiche auf Bauteilebene durchzuführen. Um die Stockwerke oder sogar das gesamte Gebäude zu vergleichen, muss nun die hierarchische Struktur wieder erklommen werden. Dafür müssen die Unterabschnitte der Schablone, bestehend aus Wänden, Türen und Fenstern, für jedes einzelne Stockwerk gehasht werden. Die Klasse *Storey*, bestehend aus den zuvor ermittelten Wänden, Türen und Fenstern, wird dabei in einen JSON-String umgewandelt und mit dem Hash-Verfahren *SHA256* gehasht. Die vorangegangene Umwandlung in einen JSON-string ist notwendig, um die Bauteile des Stockwerks in dessen Hash-Wert einfließen zu lassen. Das gleiche Verfahren wird dann für die Ermittlung des Gebäude Hash-Werts angewendet. In diesen Wert fließen somit alle Stockwerke und die darin beinhalteten Bauteile

ein. Der Hash-Wert in der Schablone wird mithilfe zwei verschiedener Funktionen ermittelt. Für die Eigenschaften der Bauteile wird die Funktion von Neo4j und somit das Verfahren MD5 verwendet und die restlichen Werte werden mit dem Verfahren *SHA256* ermittelt.

5.2.3 Gewährleisten der Datenintegrität

Wie bereits in 5.1.2 erläutert wurde, wird die Datenintegrität mithilfe einer digitalen Signatur gewährleistet. Verwendet wird die Signaturart RSA, auf welche schon im Unterabschnitt 4.3.2 genauer eingegangen wurde. Für die Verwendung von Signaturen sind zwei Schritte notwendig, einmal die Erstellung der digitalen Signatur für die gewünschten Daten und die Verifizierung der Signatur, um sicherzugehen, dass die Daten nicht verändert wurden.

Die digitale Signatur wird mithilfe von bereits vorgefertigten Klassen erzeugt. Folgende Codezeilen werden für die Signierung der Prüfungsdaten benötigt.

Listing 5.11: Erstellen einer digitalen Signatur mithilfe von RSA (Microsoft, 2017).

```
1  //In dieser variable wird der signierte Hash-Wert gespeichert
2  byte[] signedHashValue;
3
4  //Public und private key werden erstellt
5  RSACryptoServiceProvider rsa = new RSACryptoServiceProvider();
6
7  //Der private Key wird dem RSAPKCS1SignatureFormatter Objekt übergeben
8  RSAPKCS1SignatureFormatter rsaFormatter = new RSAPKCS1SignatureFormatter(
       rsa);
9
10 //Das Hash Verfahren SHA256 wird ausgewählt
11 rsaFormatter.SetHashAlgorithm("SHA256");
12
13 //Erstellen der Signatur
14 signedHashValue = rsaFormatter.CreateSignature(hashValue);
```

Die Variable *hashValue* beinhaltet in diesem Fall den Hash-Wert der geprüften Daten in einem Byte-Array. Das Array besteht somit aus den Modelldaten, der Schablone, den Namen des Prüfers und dem Prüfzeitpunkt. Dadurch ist es möglich den verantwortlichen Prüfer zu bestätigen und den Zeitpunkt der Prüfung nachzuvollziehen. Diese Verifizierung der Daten kann mit den Zeilen aus dem Codebeispiel 5.12 ausgeführt werden.

Listing 5.12: Verifizieren der erstellten digitalen Signatur mit RSA (Microsoft, 2017).

```
1  RSAPKCS1SignatureDeformatter rsaDeformatter = new
       RSAPKCS1SignatureDeformatter(rsa);
2  rsaDeformatter.SetHashAlgorithm("SHA256");
3  if (rsaDeformatter.VerifySignature(hashValue, signedHashValue))
```

```
4  {
5      Console.WriteLine("The signature is valid.");
6      return true;
7  }
8  else
9  {
10     Console.WriteLine("The signature is not valid.");
11     return false;
12 }
```

Wichtig zu beachten ist dabei, dass das selbe *RSACryptoServiceProvider* Objekt wie in 5.11 benutzt wird, damit der passende öffentliche Schlüssel zum bereits genutzten privaten Schlüssel verwendet werden kann. Eine neue Instanziierung der Klasse *RSACryptoServiceProvider* würde ein anderes Schlüssel Paar erzeugen und die Verifizierung ungültig machen. Der *hashValue* darf nicht, damit die digitale Signatur gültig ist, vom *hashValue* aus Listing 5.11 abweichen. Wird somit die Richtigkeit der digitalen Signatur und der damit verbundenen Prüfungsdaten bestätigt, können aktuellere Versionen des Gebäudemodells mit der geprüften Variante verglichen werden, um Abweichungen zu ermitteln.

5.2.4 Vergleich mit anderen Modellen

Wie auch schon im Unterabschnitt 5.1.3, wird in dieser Textpassage auf die Vergleiche der Gebäudemodelle eingegangen. Da bereits für beide Modelle eine Schablone erstellt wurde und die geprüften Daten verifiziert worden sind, steht dem Vergleich nichts mehr im Weg.
Da die JSON-Datei einer Baumstruktur gleicht, werden die Vergleiche auf der obersten Ebene der JSON-Schablone gestartet. Je nachdem, ob es sich bei der Prüfung um eine Teilprüfung oder eine Prüfung des gesamten Gebäudes handelt, ist der Startpunkt die Gebäudeebene (*BuildingHash*) oder das gewünschte Stockwerk (StoreyHash). Bei beiden Varianten werden die Hash-Werte der Schablonen verglichen. Stimmen beide Werte überein, wird das Vergleichsverfahren abgebrochen, da keine Unterschiede vorliegen. Sollten die Werte jedoch abweichen, muss die nächst niedrigere Ebene betrachtet werden, da in den Hash-Wert immer alle untergeordneten Objekte der JSON-Datei mit einfließen. Somit wird der iterative Vorgang ausgeführt, bis die unterste Ebene einer Abzweigung der Baumstruktur erreicht ist oder die Hash-Werte übereinstimmen. Technisch wird dies durch den Einsatz von Schleifen und if-else-Anweisungen erreicht. Abbildung 5.1 zeigt einen Teilausschnitt dieser Vorgänge.
Auf den unterschiedlichen Ebenen werden wichtige Informationen mitgenommen, um die Abweichungen genau zu beschreiben. Somit kann, falls die unterste Ebene einer Abzweigung erreicht wird, angegeben werden welches Bauteil auf welcher Etage den veränderten Hash-Wert verursacht hat. Zusätzlich dazu werden bestimmte Meldungen angehängt, um zu

erläutern wie die Abweichung entstanden ist. Folgende Auflistung ist ein kleiner Überblick der möglichen Meldungen.

- Die Bezeichnung des Stockwerks hat sich geändert.
- Startpunkt der beiden Türen stimmt nicht überein.
- Die Eigenschaft "Breite" hat sich in der aktuellen Version verändert.
- Die Eigenschaft "Höhe" hat sich in der aktuellen Version verändert.

Damit am Ende eine kompakte Übersicht über alle Abweichungen entsteht, werden die mitgenommenen Informationen, die Meldungen und die abweichenden Werte aus dem geprüftem Modell und der aktuellen Version in einer Tabelle gebündelt. Abbildung 5.3 zeigt eine Übersicht der Abweichungen zwischen zwei Modellen.

Stockwerk	Wand-Id	Wandeigenschaft	Status	Alter Wert	Neuer Wert
'Erdgeschoss'	'0Q$WeUXIL4J8BsYw		Diese Wand existiert nur im aktuellen Gebäudemodell.		
'Erdgeschoss'	'1bzfVsJqn8De5PukC	Material	Property: Material hat sich in der aktuellen Version verändert.	'Beton'	'Concrete Masonry Units'
'Erdgeschoss'	'1bzfVsJqn8De5PukC	'ThermalTransmittance'	Property: 'ThermalTransmittance' hat sich in der aktuellen Version verändert	IFCTHERMALTRANSMITTANCEMEASURE(0.4)	IFCTHERMALTRANSMITTANCEMEASURE(4.3
'Erdgeschoss'	'1$wmdwWPjDYuku_	Endpunkt	Endpunkt der beiden Wände stimmt nicht überein.	X = 3,8| Y = 4,13| Z = 0	X = 6,35| Y = 4,13| Z = 0
'Erdgeschoss'	'1$wmdwWPjDYuku_	Length	Property: Length hat sich in der aktuellen Version verändert.	3,5	6,05
'Dachgeschos	'0Q$WeUXIL4J8BsYw		Diese Wand existiert nur im aktuellen Gebäudemodell.		

Abbildung 5.3: Auflistung von Abweichungen zwischen zwei Modellen.

Zusätzlich zur Ausgabe der Abweichungen in Textform, soll es möglich sein die Veränderungen, an den Bauteilen im geprüften Modell, grafisch darzustellen. Dazu wird der bereits erwähnte Web Viewer *xBIM WeXplorer* eingesetzt. Die IFC-Datei muss dafür zuvor noch in eine wexBIM-Datei umgewandelt werden und kann im Anschluss als komplettes Modell visualisiert werden. Da die betroffenen Bauteile bereits ermittelt wurden, lassen sich die Elemente mithilfe der Bauteil-Ids im Web Viewer farbig markieren. Zusätzlich ist es möglich Bauteile zu verbergen, damit die markierten Elemente besser zu erkennen sind. Abbildung 5.4 zeigt ein Beispiel dafür. Das Ursprungsmodell ist von der Quelle Institut für Angewandte Information / Karlsruher Institut für Technologie (2020) heruntergeladen worden.

5.2.5 Erweitern der Implementierung

In der Implementierung werden zur Veranschaulichung bis jetzt nur die Wände, Fenster und Türen berücksichtigt. Auch deren Eigenschaften sind auf das Nötigste begrenzt. Das Programm ist jedoch so aufgebaut, dass der Code und somit die Schablone leicht erweitert werden können.

Durch ergänzen der Eigenschaftslist mit dem Namen der gewünschten Eigenschaft, können diese zusätzlich in der Schablone aufgeführt werden, da die Abfragen an Neo4j von innerhalb der Anwendung allgemein formuliert sind. Folgender Codeausschnitt zeigt, wie die Eigenschaft einer Wand ermittelt wird.

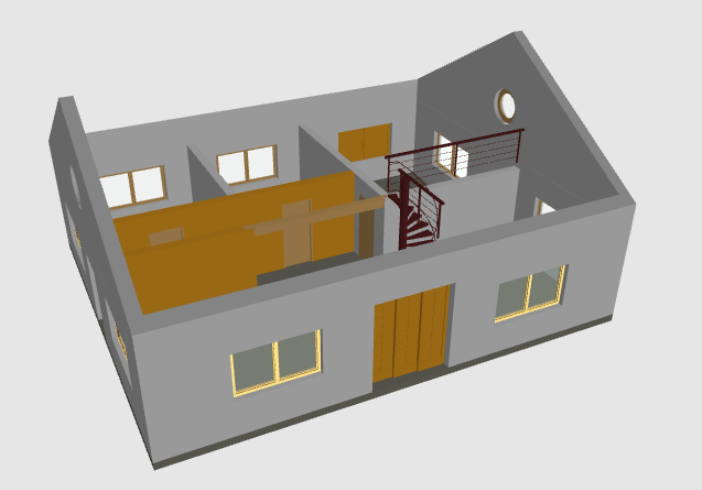

Abbildung 5.4: Darstellen der abweichenden Bauteile (gelb markiert) mit dem Web Viewer xBIM WeXplorer.

Listing 5.13: Ermitteln der Eigenschaften einer Wand in der Implementierung mithilfe der *Bolt-Connection* zur Datenbank und dem *Neo4j Driver*.

```
1 var result = tx.Run("MATCH(n1)-[]-(relProperties:
      IFCRELDEFINESBYPROPERTIES)-[]-(propertySet:IFCPROPERTYSET)-[]-(
      property:IFCPROPERTYSINGLEVALUE) " +
2                     "WHERE n1.model = $ModelName AND n1.EntityId = $id
                          AND property.prop0 = $name " +
3                     "RETURN property.prop2 as value",
4 new { ModelName, id, name });
```

Durch den Platzhalter *name* und einer Schleife wird die Eigenschaftsnamensliste abgearbeitet. Möchte man also zusätzlich die Feuerfestigkeit der Wand herausfinden, falls diese definiert wurde, muss nur die Liste ergänzt werden.

Neben Eigenschaften können auch ganz leicht zusätzliche Bauteile aufgeführt werden. Abbildung 5.2 führt neben den bereits integrierten Bauteilen (Wände, Türen und Fenster) auch noch Treppen auf. Eine Integration von Treppen und z.B. auch Stützen ist durch den modularen Aufbau des Programms möglich. Kleine Anpassungen im Code für diese Elemente, aufgrund von anderen Eigenschaften oder veränderter Ermittlung der geometrischen Abmessungen, lassen sich jedoch nicht vermeiden. Die Schablone würde somit, auf der gleichen hierarchischen Ebene, neben dem Array aus Wänden ein weiteres Array aus z.B. Stützen beinhalten.

Eine Erweiterung der Methode und Implementierung über den Hochbau hinaus ist natürlich auch möglich. Neben den bereits aufgeführten Bauwerken aus dem Hochbau, lassen sich somit auch Schablonen für Bauwerksmodelle aus den Teilbereichen Ingenieurbau, Tiefbau oder Straßenbau erstellen. Ein Vergleich zwischen den verschiedenen Versionen eines Straßenmodells

wäre somit auch durchführbar. Die Implementierung und somit der Aufbau der Schablone müsste jedoch für das spezielle Teilgebiet des Bauwesens angepasst werden.

Kapitel 6

Beispielhafte Verwendung der Methode

Die Methode des digitalen Planstempels und dessen Implementierung soll in diesem Kapitel anhand von zwei Beispielen aufgezeigt werden. Bei den Beispielen handelt es sich einmal um eine Gesamtprüfung eines Gebäudes und einmal nur um die Genehmigung des Erdgeschosses. Umgesetzt werden hierbei zwei der Szenarien aus dem Unterabschnitt 2.4.1.

6.1 Gesamtprüfung

Als erstes Beispiel wird die Gesamtprüfung eines einfachen zweistöckigen Hauses betrachtet. Angelehnt ist dies an das 1. Szenario aus 2.4.1. Am zu betrachtenden Gebäude ist ein Schaden festgestellt worden. Nun soll ermittelt werden, ob der gebaute Zustand von der geprüften Version des Gebäudes abweicht. Dadurch lässt sich feststellen ob ein Fehler bei der Prüfung gemacht wurde oder erste bei der Bauausführung. Unter anderem wird somit auch die Prüfung des Prüfstatikers untersucht. Im folgenden soll das vom Statiker geprüfte Modell mit dem aktuellen Gebäudemodell verglichen werden. Das Ursprungsmodell ist von der Quelle Institut für Angewandte Information / Karlsruher Institut für Technologie (2020) heruntergeladen worden.

In der gebauten Version des Modells wurden zu Testzwecken mehrere Bauteile verändert. Unter anderem ist eine tragende Wand verlängert, die geometrischen Abmessungen und Eigenschaften von Bauteilen verändert und zwei neue Wände, eine tragende und eine nichttragende Wand (Trennwand), eingebaut worden. In Abbildung 6.1 werden beide Bauwerke dargestellt, in der gebauten Version sind die abweichenden Bauteile markiert.

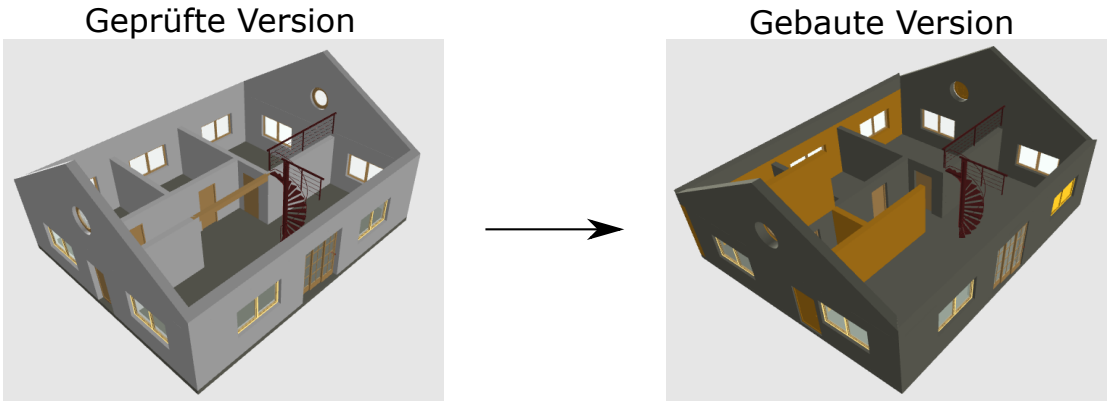

Abbildung 6.1: Darstellung der Veränderungen zwischen dem geprüften und gebautem Gebäude.

Für den Prüfstatiker sind nur die tragwerksrelevanten Bauteile von Interesse, deswegen werden auch nur diese in der Schablone der Modelle aufgeführt. Abweichungen können somit nur an diesen Bauteilen ermittelt werden. Im Fall dieses Beispiels müssten, bis auf die zusätzliche nichttragende Wand, alle Veränderungen vermerkt werden. Damit also nun die Abweichungen ermittelt werden können, muss zuerst die Signatur des genehmigten Modells überprüft werden. Durch die Speicherung der Modell- und Prüfungsdaten kann die digitale Signatur verifiziert werden. Sobald die Richtigkeit bestätigt wurde, ist es möglich eine Schablone für das aktuelle Modell zu erstellen. Sollten beide Schablonen erstellt worden sein, können die Veränderungen ermittelt und in einer Tabelle aufgelistet werden. Die Ausgabe des Programms wir in Abbildung 6.2 dargestellt. Zu beachten ist dabei, dass die zusätzliche Wand im Dachgeschoss (siehe Abbildung 6.1) nicht unter den Veränderungen aufgeführt wird, da diese keine tragwerksrelevante Wand darstellt.

Stockwerk	Wand-Id	Wandeigenschaft	Wandbauteil	Wandbauteil-Id	Wandbauteileigenschaft	Status	Alter Wert	Neuer Wert
'Erdgeschoss'	'3e9rWi4v13J9DADE7qDOE9'					Diese Wand existiert nur im aktuellen Gebäudemodell.		
'Erdgeschoss'	'1bzfVsJqn8De5PukCrqylz'	Material				Property: Material hat sich in der aktuellen Version verändert.	'Beton'	'Concrete Masonry Units'
'Erdgeschoss'	'3rPX_Juz59peXXY6wDJI18'		Tür	'2jTRqchjf7oB0yhC	Startpunkt	Startpunkt der beiden Türen stimmt nicht überein.	X = -0,17 | Y = 5,61 | Z = 0,00	X = -0,17 | Y = 4,41 | Z = 0,00
'Erdgeschoss'	'1$wmdwWPjDYuku_ghVkynE'	Endpunkt				Endpunkt der beiden Wände stimmt nicht überein.	X = 3,8 | Y = 4,13 | Z = 0	X = 6,3 | Y = 4,13 | Z = 0
'Erdgeschoss'	'1$wmdwWPjDYuku_ghVkynE'	Length				Property: Length hat sich in der aktuellen Version verändert.	3,5	6
'Erdgeschoss'	'16DNNqzfP2thtfaOflvsKA'		Fenster	'3BFcylCsX74PQAc	Startpunkt	Startpunkt der beiden Fenster stimmt nicht überein.	X = 10,50 | Y = -0,17 | Z = 0,80	X = 10,90 | Y = -0,17 | Z = 1,0
'Dachgeschos	'3VCarUKgH1buLo22Ozxe6J'		Fenster	'2ACmFFQhT1Ouf	Startpunkt	Startpunkt der beiden Fenster stimmt nicht überein.	X = 12,11 | Y = 5,50 | Z = 0,80	X = 12,11 | Y = 6,85 | Z = 0,42

Abbildung 6.2: Auflistung der Veränderungen zwischen dem geprüften und gebautem Gebäude durch die Anwendung.

Neben der Auflistung der Veränderungen können diese auch noch im Bauwerk visualisiert werden. Die aufgeführten Bauteile aus Abbildung 6.2 werden dort farbig dargestellt. Ein Vergleich der markierten Bauteile aus Abbildung 6.3 und der gebauten Version aus Abbildung 6.1 bestätigt, dass nur die Bauteile, welche verändert wurden, farbig markiert sind. Bauteile die neue dazugekommen sind, werden in der Visualisierung nicht aufgezeigt, sind jedoch in der Auflistung zu finden.

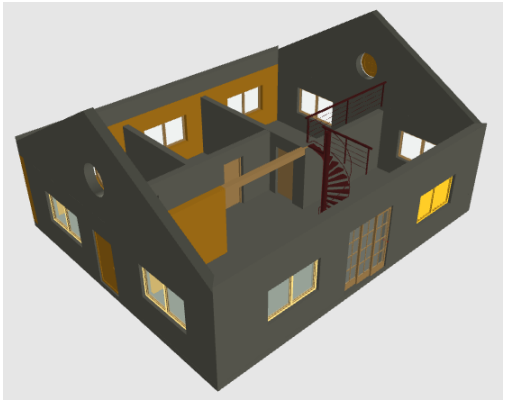

Abbildung 6.3: Grafische Darstellung der Veränderungen am geprüften Modell mithilfe von xBIM WeXplorer.

6.2 Teilprüfung

Beim zweiten Beispiel handelt es sich um eine Teilprüfung eines Gebäudes, genauer gesagt soll das 3. Szenario aus 2.4.1 dargestellt werden. Dabei soll nur ein Stockwerk des Bauwerks genehmigt werden, damit die Arbeiten am Gebäude starten können. Die restlichen Stockwerke werden nachträglich genehmigt. Bei der Genehmigung der restlichen Etagen ist dabei zu beachten, dass der bereits geprüfte Teil, in diesem Fall das Erdgeschoss, sich nicht verändern darf. Deswegen mussen bei jeder erneuten Prüfung alle schon abgesegneten Bereich auf Veränderungen überprüft werden. Dies soll im folgenden Beispiel veranschaulicht werden.

Als Bauwerk wird für dieses Beispiel ein zweistöckiges Haus verwendet. Für dieses kleine Haus würde normalerweise keine Teilgenehmigung durchgeführt werden. Zum veranschaulichen der Methode wird jedoch das Erdgeschoss getrennt geprüft. Wenn also nun das Obergeschoss zur Prüfung eingereicht wird, muss das bereits genehmigte Erdgeschoss auf Veränderungen untersucht werden. Die Abbildung 6.4 zeigt die zwei Versionen des Erdgeschosses.

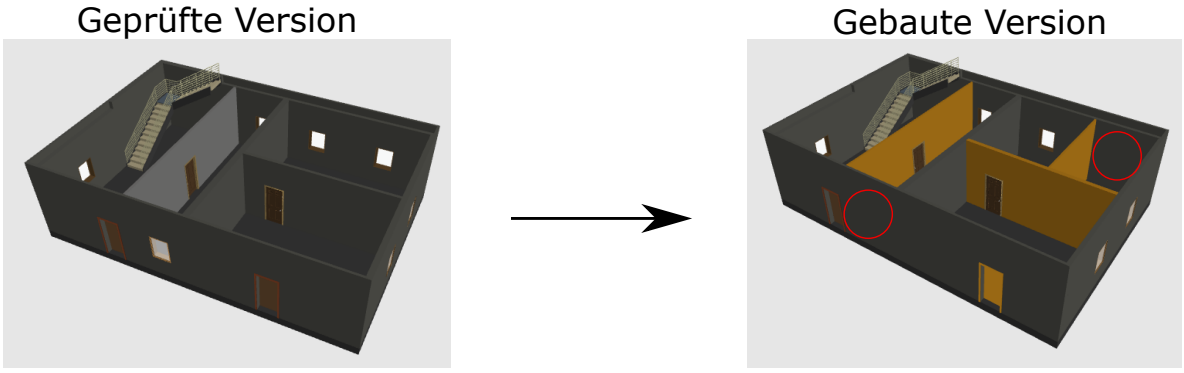

Abbildung 6.4: Darstellung der Veränderungen (gelb) zwischen dem geprüften und gebautem Gebäude. Fehlende Elemente in der gebauten Version sind durch rote Kreise markiert.

Die veränderten Bauteile sind in der neuen Version markiert. Als Test wurden geometrische Abmessungen und Eigenschaften von Bauteilen verändert und neue Bauteile hinzugefügt. Im Gegensatz zum Beispiel in 6.1, werden alle Bauteile, auch die nichttragenden, aufgeführt. Abbildung 6.5 zeigt die aufgelisteten Veränderungen in der Anwendung. Abbildung 6.6 zeigt die grafische Darstellung des Erdgeschosses. Die abweichenden Bauteile sind farbig markiert. Zu beachten ist dabei, dass in Abbildung 6.6 im Vergleich zu Abbildung 6.4 die zwei Türen der beiden gelben Wände auch farbig markiert sind. Der Grund hierfür ist, dass durch die veränderte Dicke der Wand auch die Koordinaten der Tür leicht verändert wurden. Die Abweichung ist zwar sehr gering, führt aber beim hashen zu einem anderen Wert. Damit solche Kleinigkeiten nicht auftauchen, müssen bestimmte Toleranzen im Code festgelegt werden.

Stockwerk	Wand-Id	Wandeigenschaft	Wandbauteil	Wandbauteil-Id	Wandbauteileigenschaft	Status	Alter Wert	Neuer Wert
'Erdgeschoss'	'2RGUgLUd1FYRf					Diese Wand existiert nur im aktuellen Gebäudemodell.		
'Erdgeschoss'	'3xbJkdANf7Zgd\	Startpunkt				Startpunkt der beiden Wände stimmt nicht überein.	X = -13190,17 | Y = 5018,30 | Z = 0,00	X = -13165,17 | Y = 5018,30 | Z = 0,00
'Erdgeschoss'	'3xbJkdANf7Zgd\	Endpunkt				Endpunkt der beiden Wände stimmt nicht überein.	X = -13190,17 | Y = -2219,2 | Z = 0	X = -13165,17 | Y = -2219,2 | Z = 0
'Erdgeschoss'	'3xbJkdANf7Zgd\	Material				Property: Material hat sich in der aktuellen Version ver.	'Default Wall'	'Concrete Masonry Units'
'Erdgeschoss'	'3xbJkdANf7Zgd\	'ThermalTransmittance'				Property: 'ThermalTransmittance' hat sich in der aktuel		IFCTHERMALTRANSMITTANCEMEASU
'Erdgeschoss'	'3xbJkdANf7Zgd\	Width				Property: Width hat sich in der aktuellen Version verän	200	150
'Erdgeschoss'	'3xbJkdANf7Zgd\		Tür	'3xbJkdANf7Zg	Startpunkt	Startpunkt der beiden Türen stimmt nicht überein.	X = 2005,00 | Y = 100,00 | Z = 0,00	X = 2005,00 | Y = 75,00 | Z = 0,00
'Erdgeschoss'	'3xbJkdANf7Zgd\	Startpunkt				Startpunkt der beiden Wände stimmt nicht überein.	X = -17977,67 | Y = 8093,30 | Z = 0,00	X = -17977,67 | Y = 8068,30 | Z = 0,00
'Erdgeschoss'	'3xbJkdANf7Zgd\	Endpunkt				Endpunkt der beiden Wände stimmt nicht überein.	X = -8202,67 | Y = 8093,3 | Z = 0	X = -8202,67 | Y = 8068,3 | Z = 0
'Erdgeschoss'	'3xbJkdANf7Zgd\	Material				Property: Material hat sich in der aktuellen Version ver.	'Concrete Masonry Units'	'Default Wall'
'Erdgeschoss'	'3xbJkdANf7Zgd\	'ThermalTransmittance'				Property: 'ThermalTransmittance' hat sich in der aktuel	IFCTHERMALTRANSMITTANCEMEASU	
'Erdgeschoss'	'3xbJkdANf7Zgd\	Width				Property: Width hat sich in der aktuellen Version verän	150	200
'Erdgeschoss'	'3xbJkdANf7Zgd\		Tür	'3xbJkdANf7Zg	Startpunkt	Startpunkt der beiden Türen stimmt nicht überein.	X = 4432,50 | Y = -75,00 | Z = 0,00	X = 4432,50 | Y = -100,00 | Z = 0,00
'Erdgeschoss'	'0grKb8u8L7$AH		Tür	'0eEZggiBzBEO	Startpunkt	Startpunkt der beiden Türen stimmt nicht überein.	X = 10932,50 | Y = -150,00 | Z = 0,00	X = 12332,50 | Y = -150,00 | Z = 0,00
'Erdgeschoss'	'0grKb8u8L7$AH		Fenster	'2wBpHCZDr2c		Dieses Fenster existiert nicht mehr im aktuellen Gebäu		
'Erdgeschoss'	'0grKb8u8L7$AH		Fenster	'2wBpHCZDr2c		Dieses Fenster existiert nicht mehr im aktuellen Gebäu		

Abbildung 6.5: Auflistung der Veränderungen zwischen dem geprüften und gebautem Erdgeschoss durch die Anwendung.

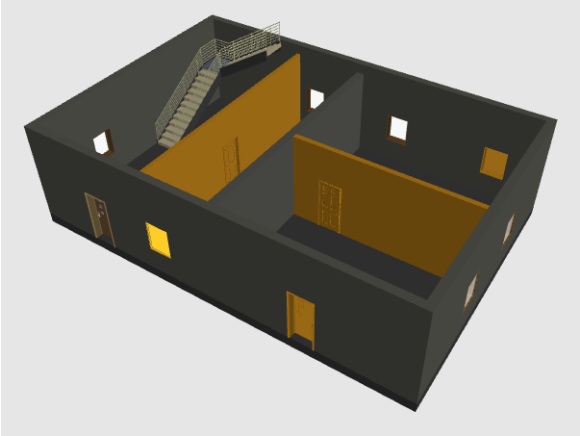

Abbildung 6.6: Grafische Darstellung der Veränderungen im Erdgeschoss des geprüften Modell mithilfe von xBIM WeXplorer.

Kapitel 7

Forschungsstand und alternativ Methoden zur Umsetzung des digitalen Planstempels

In diesem Kapitel wird untersucht, mithilfe welcher anderen Ansätze die Methode des digitalen Planstempels umgesetzt werden kann. Unter anderem sollen Möglichkeiten aufgezeigt werden, um Modelldaten zu vergleichen. Außerdem werden verschiedene Optionen betrachtet, wie BIM-Daten verwendet werden können. Doch bevor diese Thematiken behandelt werden, wird zuerst der Forschungsstand zur digitalen Baugenehmigung dargestellt.

7.1 Digitale Baugenehmigung

Das erstellen von Konzepten und Methoden zum Vergleichen von geprüften Gebäudemodellen macht dann Sinn, wenn es möglich ist Modelle in das Bauantragsverfahren einfließen zu lassen. Deswegen muss betrachtet werden, wie der Forschungsstand zum digitalen Baugenehmigungsverfahren ist.

Für die Thematik der digitalen Baugenehmigung wurde das Förderprojekt „BIM-basierter Bauantrag“ gegründet. Dabei wird eine Integration von BIM-Modellen in das Baugenehmigungsverfahren erforscht (Terfehr, 2019). Das Projekt wird von *planen-bauen 4.0 GmbH* in Zusammenarbeit mit zwei Projektpartnern und mehreren Verbänden und Kammern durchgeführt. Ziel des Projektes ist es dabei die effektive Arbeitsweise eines BIM-basierten Bauantrags hervorzuheben und gewisse Rahmenbindungen abzustecken. Wichtige Rahmenbedingungen sind dabei, die bundesweite Vereinheitlichung (z.B. Modellierungsrichtlinien), die Betrachtung von Software-neutralen Formaten und die Klärung von rechtlichen Aspekten (Mittelstand 4.0-Kompetenzzentrum Planen und Bauen, 2020). Seit April liegen erste Ergeb-

nisse zu den gestellten Fragen vor. Unter anderem wurde der Ablauf des Gesamtprozesses definiert, sowie ein Konzept zu verwendbaren offenen Datenstandards erstellt. Zusätzlich wurde eine Prototypische Software vorgestellt, welche beim erstellen von Anträgen und dem bearbeiten dieser verwendet werden kann (BIM-basierter Bauantrag, 2020a).
Ein möglicher Ablauf des BIM-basierten Bauantrags könnte wie in Abbildung 7.1 aussehen. Für die Erstellung des Antrags wird dabei ein Modell benötigt, welches in Software-neutralen Formaten (z.B. IFC) übergeben wird. Zusammen mit der Datei und den entnommenen Formulardaten kann ein Bauantrag eingereicht werden (Mittelstand 4.0-Kompetenzzentrum Planen und Bauen, 2020).

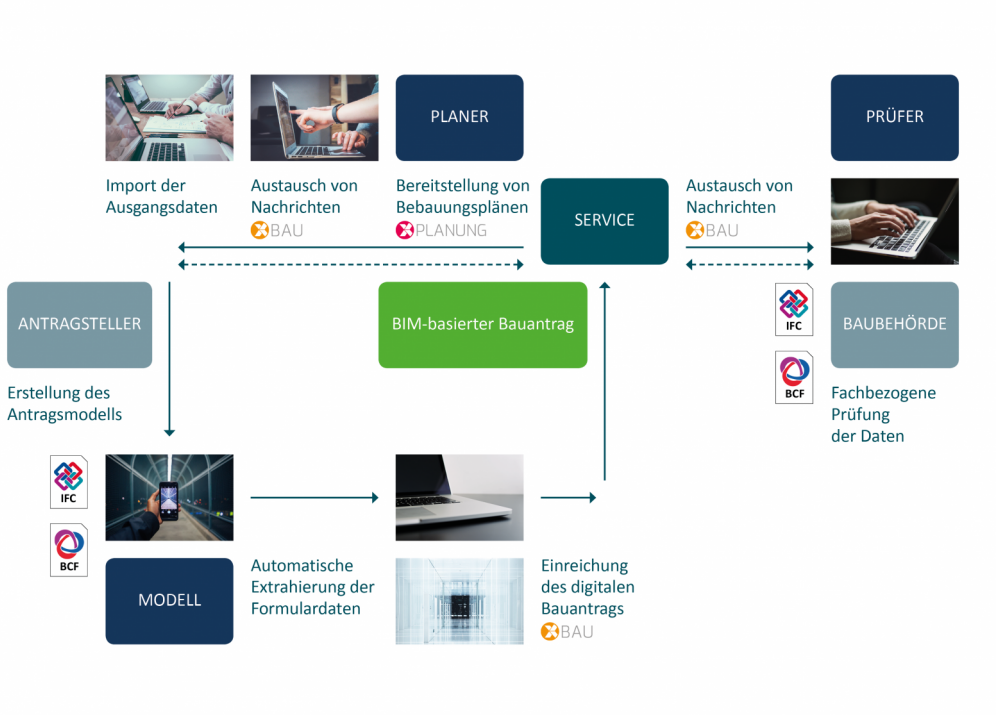

Abbildung 7.1: Darstellung des Ablaufs eines BIM-basierten Bauantrags. (Mittelstand 4.0-Kompetenzzentrum Planen und Bauen, 2020)

Genauere Aussagen zum gesamten Vorhaben und möglichen zeitlichen Aspekten können wahrscheinlich erst später getroffen werden. Das Projekt läuft noch bis 20.06.2020 (BIM-basierter Bauantrag, 2020b).

7.2 Zugang zu BIM-Daten

Die Methode des digitalen Planstempels baut auf der Verwendung von IFC-Dateien im STEP Format auf. Diese Daten werden zusätzlich in eine Graphdatenbank geladen um Informationen herauszufiltern. Doch es gibt noch andere Methoden, wie die BIM-Daten zugänglich gemacht werden können. Bezogen auf die Verwendung von Graphdatenbanken würde sich ein

Einsatz von ifcOWL anbieten. Bei ifcOWL wird das IFC-Schema für die Web Ontology Language (OWL) abgebildet. Eine Darstellung der Daten erfolgt mithilfe von Resource Description Framework (RDF). RDF zu ifcOWL kann dabei wie SPF zu EXPRESS angesehen werden (buildingSMART, 2019b). Abbildung 7.2 listet die verschiedenen Schemamöglichkeiten für IFC auf.

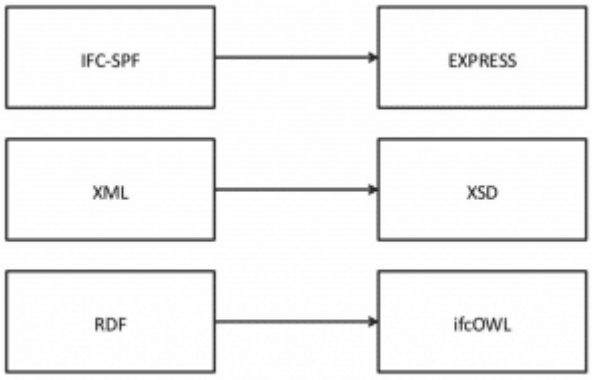

Abbildung 7.2: Darstellung der verschiedenen Schemamöglichkeiten für IFC. (buildingSMART, 2019b)

RDF stellt die Daten in einem gerichteten Graphen dar. Durch den Einsatz von ifcOWL würde der Zwischenschritt von IFC-SPF zur Graphdatenbank entfallen. Die Dateigröße würde jedoch, im Vergleich zu SPF, um mehr als 816% ansteigen. Der Nutzeranteil von IFC-SPF ist, im Gegensatz zu RDF mit ifcOWL , weit größer (buildingSMART, 2019a). Ein umschwenken würde somit wenig Sinn ergeben.

Neben ifcOWL gibt es noch andere Methoden, wie BIM-Daten mit einem Ontologie Ansatz zugänglich gemacht werden können. Zum Beispiel gehen Niknam & Karshenas (2017) in ihrer Veröffentlichung darauf ein, wie BIM-Daten als „Semantic Web“ Format, also wie ein Netz aus verbundenen Daten, abgebildet werden kann. Das genannte Format soll die Verknüpfung von verschieden Informationen erleichtern, weil es als Graph gespeichert wird (Niknam & Karshenas, 2017).

Zusätzlich zu IFC-SPF, RDF und anderen offiziellen Formaten, wird gerade ein weiterer Kandidat untersucht (buildingSMART, 2019a). Das *IFCJSON-Team* arbeitet daran mithilfe von JSON das Schema und die Instanzen von IFC darstellen zu können (Shelden *et al.*, 2020). Durch die Baumstruktur der JSON-Datei könnte die Lesbarkeit der Daten stark erhöht werden (Brouwer & Pauwels, 2020). Im Falle des digitalen Planstempels könnte somit direkt aus der *IFCJSON*-Datei die Schablone erstellt werden. Laut buildingSMART (2019a) wird JSON jedoch noch nicht als offizielles Format aufgeführt.

7.3 Ermitteln von Veränderungen

Veränderungen zwischen zwei Versionen des gleichen Modells können durch viele verschiedene Methoden ermittelt werden. Im Falle dieser Arbeit werden Schablonen der Modelle zum betrachteten Zeitpunkt ermittelt und verglichen. Eine andere Möglichkeit ist es, jede Veränderung am Modell zu vermerken. Eine Analyse der notierten Änderungen lässt erkennen ob gravierende Veränderungen vorgenommen wurden. In der Veröffentlichung von Pilehchian *et al.* (2015) wird auf diese Thematik eingegangen. Es wird ein Konzept dargelegt, wie Designänderungen eines BIM-Projekts ermittelt werden können.

Das Nachverfolgen von Änderungen am Model wird mithilfe eines Graphen möglich gemacht. Die zu betrachtenden Elemente werden als Knoten im Graphen dargestellt und mit anderen Komponenten verbunden. Durch die Verbindung untereinander, können bei der Veränderung eines Knotens die zusätzlich betroffenen Elemente ausfindig gemacht werden. Der Einsatz von IFC bietet sich zur Erstellung des Graphen an (Pilehchian *et al.*, 2015). Das Speichern der Veränderungen und betroffenen Knoten erlaubt es die Unterschiede von genehmigtem Modell zu aktuellem Modell zu erkennen.

Eine weitere Möglichkeit zur Ermittlung von Unterschieden zwischen Gebäudemodellen beschäftigt sich auch mit Graphen. Dabei sollen die unterschiedlichen Modelle in Graphenform auf ihre Ähnlichkeit verglichen werden. Laut Koutra *et al.* (2011) gibt es verschiedene Methoden, wie die Ähnlichkeit von zwei Graphen bestimmt werden kann. Zum Beispiel mit der „Eigenvalue“ Methode (Koutra *et al.*, 2011). Ein Feld das diese Thematik schon früh betrachtet hat ist die Chemieinformatik. Die Atome einer chemischen Verbindung können als Knoten eines Graphen dargestellt werden. Durch Vergleichen zweier chemischer Graphen kann ermittelt werden ob die chemischen Verbindungen identische sind (Akutsu & Nagamochi, 2013). Zum Beispiel könnte damit die Verbindung zwischen *IfcBuildingStorey* über *IfcRelContainedInSpatialStructure* zu den verschiedenen *IfcBuildingElements* auf Ähnlichkeit überprüft werden. Ein zusätzliches Bauteil in dieser Knotenkette würde somit direkt auffallen.

Kapitel 8

Fazit und Ausblick

Zum Abschluss der Arbeit werden die entscheidenden Thematiken der Methode des digitalen Planstempels noch einmal aufgegriffen, um damit ein Fazit für die dargestellte Methode zu ziehen. Außerdem wird darauf eingegangen, wie das Konzept in Zukunft verwendet werden kann und welche Anpassungen dafür notwendig sind.

8.1 Fazit

Damit die voranschreitende Digitalisierung auch in der Baubranche Einzug erhält, werden Anwendungen und Konzepte benötigt, die Planern und Prüfern unter die Arme greifen. Vor allem die Verwendung von BIM trägt dazu bei, dass viele Bereiche zwangsweise modernisiert bzw. digitalisiert werden müssen und teilweise neue Anwendungen notwendig sind. Es gibt bereits viele Beispiele solcher neu entwickelten Anwendungen, sei es ein Modellchecker, ein Modellviewer oder viele andere Applikationen, die den Nutzer bei der Arbeit mit Bauwerksmodellen unterstützen und die sonst aufwendigen manuellen Tätigkeiten ersetzen.

Auch die Methode des digitalen Planstempels geht in eine solche Richtung. Mithilfe von BIM-Daten und technologischen bzw. kryptografischen Verfahren (Blockchain, digitale Signatur) werden Tätigkeiten ausgeführt, die den Prüfer beim Prüfverfahren unterstützen und bestimmte Vorgänge vereinfachen können.

Natürlich entsteht auch oft ein Mehraufwand durch den Einsatz von BIM-Methoden und neuen Anwendungen. Neben dem Einführen von neuen Arbeitsprozessen und Konzepten, müssen auch oft die technologischen Strukturen einer Firma oder Behörde angepasst werden. Auch das Erlernen der neuen Werkzeuge und Konzepte kann ein großes Hindernis für manche darstellen. Jedoch lohnt sich, meiner Meinung nach, in vielen Fällen dieser Mehraufwand.

Im Falle des digitalen Planstempels würden viele Bereiche davon profitieren. Ich denke, dass diese Methode gut in den Planung- und Genehmigungsprozess integriert werden kann. Überall dort, wo ein bestimmter Stand des Modells festgehalten werden möchte, kann die Methode dazu verwendet werden eine Schablone dieses Bauwerks zu erstellen. Ein Einsatz ist somit immer dann sinnvoll, wenn zwei Modelle verglichen werden müssen oder man den Fortschritt von einer alten Version des Gebäudemodells zu einer aktuellen Version dokumentieren möchte. Durch die digitale Signatur der Prüfungsdaten, können die Daten dem zuständigen Prüfer zugeordnet werden. Zuständigkeiten lassen sich somit auch nach Jahren noch verifizieren.

Im Lauf der Arbeit sind bereits verschiedene Einsatzszenarien vorgestellt worden. Besonders durch die Funktion der partiellen Betrachtung des Modells, werden viele verschiedene Einsatzszenarien ermöglicht. Durch die Auswahl von bestimmten Kriterien bei der Prüfung ist es möglich, das Modell nach diesen Punkten zu filtern und die Schablone nur mit diesen essentiellen Daten zu füllen. Die zwei aufgeführten Anwendungsbeispiele befassen sich auch mit dieser partiellen Betrachtung. In 6.2 wurde aufgezeigt, wie die Teilgenehmigung des Erdgeschosses bei späteren Genehmigungen verifiziert werden kann. Das zweite Anwendungsbeispiel geht auf die Prüfung des Bauwerks durch einen Statiker ein. Alle tragwerksrelevanten Bauteile werden dabei in der Schablone festgehalten. Die Methode ist dabei aber nicht nur auf diese genannten Beispiele festgelegt. Durch den modularen Aufbau der Anwendung und der Schablone sind Ergänzung bezüglich bestimmten Filterkriterien gut durchführbar. Somit ergeben sich neue Einsatzmöglichkeiten. Neben tragwerksrelevanten Bauteilen könnten somit auch brandschutzrelevante Bauteile in der JSON-Schablone festgehalten werden.

Abschließend lässt sich sagen, dass die Methode des digitalen Planstempels eine gute digitale Erweiterung des traditionellen Planstempels darstellt und eine gute Ergänzung im Planungsprozess wäre. Durch das breitgefächerte Einsatzgebiet kann es von vielen Prüfern dazu verwendet werden den Stand des zu prüfenden Modells festzuhalten. Mithilfe der daraus entstandenen Schablone lassen sich somit Vergleiche zwischen dem festgehaltenen Stand und einer veränderten Version des Modells ermitteln. Die beispielhafte Implementierung liefert dabei einen ersten Einblick in die Möglichkeiten dieser Methode.

8.2 Ausblick

Die beispielhafte Implementierung des digitalen Planstempel ist, im Vergleich zu den vielen Bereichen der Baubranche, bis jetzt nur für einen sehr kleinen Bereich einsetzbar. Wie bereits im Abschnitt zuvor beschrieben wurde, kann die Methode und Implementierung leicht ergänzt werden. Dies ist jedoch nicht nur auf zusätzliche Prüfungsszenarien begrenzt. Ein weiteres Einsatzgebiet könnte ein Planungsbüro sein, welches sich die Unterschiede beim Detaillierungsgrad zwischen zwei Versionen anzeigen lassen möchte. In dieser Arbeit wurde vor allem auf die Verwendung der Methode im Hochbau eingegangen, jedoch kann das Konzept

auch auf die anderen Teilgebiete des Bauwesens angewendet werden. Beispielsweise könnte damit der geprüfte Stand eines Brücken- oder Straßenmodells festgehalten werden. Für eine Ausrichtung der Methode auf diese Teilgebiete muss jedoch die Implementierung und somit auch die Struktur der Schablone angepasst werden.

Besonders wichtig für die Methode bzw. Modellschablone ist eine einheitliche Darstellung des Gebäudemodells. Die Ermittlung von Eigenschaften und Abmessungen in der Implementierung sollte so allgemein wie möglich gehalten sein, deswegen ist die einheitliche Darstellung zwingend notwendig. Durch das Austauschformat IFC war dies gut möglich, jedoch kam es auch hierbei zu Differenzen durch eigentlich unwichtige Veränderungen in der Datei. Eine Feinabstimmung der Vergleichsmethoden ist deswegen notwendig, um solche unwichtigen Differenzen in den IFC-Dateien abzufangen. Außerdem wäre eine Überprüfung der alternativen IFC-Formate (RDF oder JSON) eine gute Option, um die Erstellung der Schablone zu vereinfachen und zu beschleunigen.

Ein anderer Aspekt, der sich auf die zukünftige Nutzung der Methode des digitalen Planstempels auswirkt, ist die Thematik des „BIM-basierten Bauantrags“. Sollte ein Verfahren, wie es in Abschnitt 7.1 thematisiert wird, möglich gemacht werden, würde sich der Einsatz der Methode für die Baugenehmigung lohnen. In welchem Umfang und wann solch ein Vorgehen jedoch überhaupt zum Einsatz kommen könnte, steht noch nicht fest.

Anhang A

Digitaler Anhang

Bei der Abgabe werden dieser Arbeit in digitaler Form beigefügt:

- Einmal die originale Ausarbeitung in PDF-Format
- Einmal die editierte Ausarbeitung in PDF-Format für die Webseite
- Den deutschen und englischen Titel der Arbeit in einer Textdatei
- Abstract und Zusammenfassung in einer Textdatei
- Abbildungen die in der Arbeit verwendet wurden
- Das Visual Studio Projekt der Implementierung
- Verwendete IFC-Modelle und das Revit-Projekt
- Erstellte Schablonen für diese IFC-Modelle
- Die Graphdatenbanken der IFC-Modelle

Literaturverzeichnis

Akutsu, T. & Nagamochi, H. (2013). Comparison and enumeration of chemical graphs. *Computational and structural biotechnology journal* 5.

Albrecht, M. (2015). *Building Information Modeling (BIM) in der Planung von Bauleistungen.* Hamburg: Disserta Verlag.

Amann, J., Tauscher, E. & Borrmann, A. (2015). BIM-Programmierwerkzeuge. In: A. Borrmann, M. König, C. Koch, & J. Beetz (Hrsg.), *Building Information Modeling*, VDI-Buch, S. 193–204. Wiesbaden: Springer Vieweg.

Autodesk (2018). Revit IFC Handbuch - Ausführliche Anleitung für den Umgang mit IFC-Dateien.

Autodesk (2020). Liste aller Produkte. URL: https://www.autodesk.de/products [Letzter Zugriff am: 14.06.2020].

Azhar, S. (2011). Building Information Modeling (BIM): Trends, Benefits, Risks, and Challenges for the AEC Industry. *Leadership and Management in Engineering* 11(3), S. 241–252.

BauInfoConsult GmbH (2019). BIM 2019: Studie zeigt, wie es aktuell am Bau um BIM bestellt ist. URL: https://www.pressebox.de/pressemitteilung/bauinfoconsult-gmbh/BIM-2019-Studie-zeigt-wie-es-aktuell-am-Bau-um-BIM-bestellt-ist/boxid/963427 [Letzter Zugriff am: 14.06.2020].

Baunetz_Wissen (2020a). Digitales Bauwerksmodell — BIM — Glossar — Baunetz_Wissen. URL: https://www.baunetzwissen.de/glossar/d/digitales-bauwerksmodell-5314315 [Letzter Zugriff am: 14.06.2020].

Baunetz_Wissen (2020b). Was bedeutet LOD/LOI? — BIM — Modellinhalte — Baunetz_Wissen. URL: https://www.baunetzwissen.de/bim/fachwissen/modellinhalte/was-bedeutet-lod-loi-5285890 [Letzter Zugriff am: 14.06.2020].

Behaneck, M. (2015). Teil 1: Was kann BIM tatsächlich, was nicht und was ändert sich für Planer? Intelligenter planen, bauen und nutzen. URL: https://www.db-bauzeitung.de/db-themen/technik/was-kann-bim-tatsaechlich/ [Letzter Zugriff am: 14.06.2020].

Berger, C. (2017). BIM reduziert vielerlei Kosten. URL: https://www.springerprofessional.de/building-information-modeling/bau-projektmanagement/bim-reduziert-vielerlei-kosten/15055460 [Letzter Zugriff am: 14.06.2020].

Bergwanger, J., Steven, M., Krommes, W. & Winter, E. (2019). Prozess. URL: https://wirtschaftslexikon.gabler.de/definition/prozess-45614 [Letzter Zugriff am: 14.06.2020].

Beutelspacher, A., Neumann, H. B. & Schwarzpaul, T. (Hrsg.) (2010). *Kryptografie in Theorie und Praxis: Mathematische Grundlagen für Internetsicherheit, Mobilfunk und elektronisches Geld* (2., überarbeitete Auflage Aufl.). Wiesbaden: Vieweg + Teubner.

BIM-basierter Bauantrag (2020a). BIM-basierter Bauantrag. URL: http://www.bimbauantrag.de/ [Letzter Zugriff am: 14.06.2020].

BIM-basierter Bauantrag (2020b). Evaluierung Ausblick. URL: https://bim-bauantrag.blogs.ruhr-uni-bochum.de/wp-content/uploads/2020/04/004_Evaluierung_Ausblick.pdf [Letzter Zugriff am: 14.06.2020].

Binance (2020). Private, öffentliche und Konsortium-Blockchains – Was ist der Unterschied? URL: https://www.binance.vision/de/blockchain/private-public-and-consortium-blockchains-whats-the-difference [Letzter Zugriff am: 14.06.2020].

bitcoin.de (2020). Das Bitcoin Whitepaper von Satoshi Nakamoto. URL: https://www.bitcoin.de/de/bitcoin-whitepaper-deutsch [Letzter Zugriff am: 14.06.2020].

Bolt Protocol (2020). Bolt Protocol. URL: https://boltprotocol.org/ [Letzter Zugriff am: 14.06.2020].

Borrmann, A., Beetz, J., Koch, C., Liebich, T. & Muhic, S. (2018). Industry Foundation Classes: A Standardized Data Model for the Vendor-Neutral Exchange of Digital Building Models. In: A. Borrmann, M. König, C. Koch, & J. Beetz (Hrsg.), *Building Information Modeling*, S. 81–126. Cham: Springer International Publishing.

Brouwer, J. & Pauwels, P. (2020). IFCJSON-Team/IFC.JSON-4. URL: https://github.com/IFCJSON-Team/IFC.JSON-4/tree/master/Documentation [Letzter Zugriff am: 14.06.2020].

buildingSMART (2019a). IFC Formats - buildingSMART Technical. URL: https://technical.buildingsmart.org/standards/ifc/ifc-formats/ [Letzter Zugriff am: 14.06.2020].

buildingSMART (2019b). ifcOWL - buildingSMART Technical. URL: https://technical.buildingsmart.org/standards/ifc/ifc-formats/ifcowl/ [Letzter Zugriff am: 14.06.2020].

buildingSMART (2019c). IfcWallStandardCase. URL: https://standards.buildingsmart.org/IFC/RELEASE/IFC4/FINAL/HTML/schema/ifcsharedbldgelements/lexical/ifcwallstandardcase.htm [Letzter Zugriff am: 14.06.2020].

buildingSMART (2019d). Industry Foundation Classes (IFC) - buildingSMART Technical. URL: https://technical.buildingsmart.org/standards/ifc [Letzter Zugriff am: 14.06.2020].

buildingSMART (2020a). buildingSMART International. URL: https://www.buildingsmart.org/ [Letzter Zugriff am: 14.06.2020].

buildingSMART (2020b). IFC4 Documentation. URL: https://standards.buildingsmart.org/IFC/DEV/IFC4_2/FINAL/HTML/ [Letzter Zugriff am: 14.06.2020].

buildingSMART (2020c). Model View Definitions (MVD) - buildingSMART Technical. URL: https://technical.buildingsmart.org/standards/ifc/mvd/ [Letzter Zugriff am: 14.06.2020].

Bundesministerium für Verkehr und digitale Infrastruktur (2015). Stufenplan Digitales Planen und Bauen. Auftragnehmer: planen-bauen 4.0 - Gesellschaft zur Digitalisierung des Planens, Bauens und Betreibens mbH.

Chen, K. (2015). Creating a simple wall with property set and quantity information – IfcOpenShell Academy. URL: http://academy.ifcopenshell.org/creating-a-simple-wall-with-property-set-and-quantity-information/ [Letzter Zugriff am: 14.06.2020].

Chowdhury, N. (2020). *Inside blockchain, Bitcoin, and cryptocurrencies*. Boca Raton, FL: CRC Press, Taylor & Francis Group.

Daum, S. (2017). SimonDaum/QL4BIM. URL: https://github.com/SimonDaum/QL4BIM/wiki [Letzter Zugriff am: 14.06.2020].

Daum, S. & Borrmann, A. (2014). Processing of Topological BIM Queries using Boundary Representation Based Methods. *Advanced Engineering Informatics* 28(4), S. 272–286.

Deutsche Telekom AG (2018). Digitalisierungsindex Mittelstand 2018 - Der Digitale Status Quo des deutschen Mittelstands. URL: https://www.digitalisierungsindex.de/wp-content/uploads/2018/11/Telekom_Digitalisierungsindex_2018_GESAMTBERICHT.pdf [Letzter Zugriff am: 14.06.2020].

DocuSign (2017). Signaturen sind bereits seit mehr als 5000 Jahren ein kulturelles Phänomen. URL: https://www.docusign.de/blog/signaturen-sind-bereits-seit-mehr-als-5000-jahren-ein-kulturelles-phanomen/ [Letzter Zugriff am: 14.06.2020].

DocuSign (2020). Was ist eine digitale Signatur und wie wird sie erstellt? URL: https://www.docusign.de/wie-es-funktioniert/elektronische-signatur/digitale-signatur/digitale-signatur-faq [Letzter Zugriff am: 14.06.2020].

Elektronik-Kompendium (2020a). Asymmetrische Kryptografie (Verschlüsselung). URL: https://www.elektronik-kompendium.de/sites/net/1910111.htm [Letzter Zugriff am: 14.06.2020].

Elektronik-Kompendium (2020b). Kryptografische Hash-Funktionen. URL: https://www.elektronik-kompendium.de/sites/net/1909041.htm [Letzter Zugriff am: 14.06.2020].

Elektronik-Kompendium (2020c). RSA - Rivest, Shamir und Adleman. URL: https://www.elektronik-kompendium.de/sites/net/1910121.htm [Letzter Zugriff am: 14.06.2020].

Elektronik-Kompendium (2020d). Symmetrische Kryptografie (Verschlüsselung). URL: https://www.elektronik-kompendium.de/sites/net/1910101.htm [Letzter Zugriff am: 14.06.2020].

Esser, S. & Aicher, K. (2019). IfcBridge Model Generation using VisualProgramming. In: *31. Forum Bauinformatik, Berlin, 2019*. URL: https://publications.cms.bgu.tum.de/2019_Esser_FBI.pdf [Letzter Zugriff am: 14.06.2020].

Franke, L., Deckelmann, G., Franke, M., Henninger, D. & Stehr, H. (Hrsg.) (2002). *Baukonstruktion im Planungsprozess: Vom Entwurf zur Detailplanung*. Wiesbaden and s.l.: Vieweg+Teubner Verlag.

FundingUniverse (2006). Autodesk, Inc. History. URL: http://www.fundinguniverse.com/company-histories/autodesk-inc-history/ [Letzter Zugriff am: 14.06.2020].

Hartikka, L. (2017). A blockchain in 200 lines of code. URL: https://medium.com/@lhartikk/a-blockchain-in-200-lines-of-code-963cc1cc0e54 [Letzter Zugriff am: 14.06.2020].

Havenstein, A. (2007). Digitale Unterschriften mit ElGamal - Seminar Kryptographie und Datensicherheit. URL: https://www.cs.uni-potsdam.de/ti/lehre/06-Kryptographie/slides/slides-11.pdf [Letzter Zugriff am: 14.06.2020].

Helmus, M., Meins-Becker, A., Kelm, A., Bodtländer, C., Kaufhold, M., Kesting, H., Khorrami, N., Klusmann, B., Pütz, C., Scarpino, P. & Zibell, M. (2017). Building Information Modeling und Prozesse: Grundlagenbericht Teil 1. Forschungsbericht, Bergische Universität Wuppertal. URL: http://www.biminstitut.de/files/bim_institut/media/01_Forschung/Downloads/BIM%20-%20Prozesse%20%20-%20Grundlagenbericht.pdf [Letzter Zugriff am: 14.06.2020].

IfcOpenShell (2019). IfcOpenShell: v0.6.0. URL: http://blog.ifcopenshell.org/2019/12/v060.html [Letzter Zugriff am: 14.06.2020].

Institut für Angewandte Information / Karlsruher Institut für Technologie (2020). KIT IFC Examples - IfcWiki. URL: http://www.ifcwiki.org/index.php?title=KIT_IFC_Examples [Letzter Zugriff am: 14.06.2020].

Ismail, A., Nahar, A. & Scherer, R. (2017, 07). Application of graph databases and graph theory concepts for advanced analysing of BIM models based on IFC standard.

JSON (2020). Einführung in JSON. URL: https://www.json.org/json-de.html [Letzter Zugriff am: 14.06.2020].

Kaspersky (2020). Was ist ein Brute-Force-Angriff? URL: https://www.kaspersky.de/resource-center/definitions/brute-force-attack [Letzter Zugriff am: 14.06.2020].

Klotz, M. (2016). Gar kein Mysterium: Blockchain verständlich erklärt. URL: https://www.it-finanzmagazin.de/gar-kein-mysterium-blockchain-verstaendlich-erklaert-27960/ [Letzter Zugriff am: 14.06.2020].

Koonce, D., Huang, L. & Judd, R. (1998). EQL an Express Query Language. *Computers & Industrial Engineering* 35(1-2), S. 271–274.

Koutra, D., Ramdas, A. & Xiang, J. (2011). Algorithms for Graph Similarity and Subgraph Matching.

Krauth, W. (2018). Siegel - LEO-BW. in: Südwestdeutsche Archivalienkunde, URL: https://www.leo-bw.de/themenmodul/sudwestdeutsche-archivalienkunde/archivalienelemente/siegel [Letzter Zugriff am: 14.06.2020].

Kryptowissen.de (2020a). Asymmetrische Verschlüsselung. URL: https://www.kryptowissen.de/asymmetrische-verschluesselung.html [Letzter Zugriff am: 14.06.2020].

Kryptowissen.de (2020b). Symmetrische Verschlüsselung. URL: https://www.kryptowissen.de/symmetrische-verschluesselung.html [Letzter Zugriff am: 14.06.2020].

Lang, M. & Karlstetter, F. (2017). Consensus-Modelle in der Übersicht. URL: https://www.dev-insider.de/consensus-modelle-in-der-uebersicht-a-631671/ [Letzter Zugriff am: 14.06.2020].

Luber, S. & Litzel, N. (2019). Was ist eine Graphdatenbank? URL: https://www.bigdata-insider.de/was-ist-eine-graphdatenbank-a-788834/ [Letzter Zugriff am: 14.06.2020].

Manz, O. (2019). *Verschlüsseln, Signieren, Angreifen: Eine kompakte Einführung in die Kryptografie* (1st ed. 2019 Aufl.).

Mensch und Maschine Deutschland GmbH (2019). BIM für Tragwerksplaner. URL: https://www.wirmachenbim.com/de/industrien/tragwerksplanung/ [Letzter Zugriff am: 14.06.2020].

Microsoft (2017). Kryptografische Signaturen. URL: https://docs.microsoft.com/de-de/dotnet/standard/security/cryptographic-signatures [Letzter Zugriff am: 14.06.2020].

Microsoft (2019). Übersicht über Visual Studio. URL: https://docs.microsoft.com/de-de/visualstudio/get-started/visual-studio-ide?view=vs-2019 [Letzter Zugriff am: 14.06.2020].

Mittelstand 4.0-Kompetenzzentrum Planen und Bauen (2020). BIM-basierter Bauantrag – Ergebnisse aus dem Forschungsprojekt ZukunftBAU vorgestellt. URL: https://www.kompetenzzentrum-planen-und-bauen.digital/kos/WNetz?art=News.show&id=642 [Letzter Zugriff am: 14.06.2020].

Nahar, A. (2017). Applying graph theory concepts for analyzing BIM models based on IFC standards. Masterarbeit, Technische Universität Dresden.

NBS (2019). National BIM Report 2019. Unterüberschrift: UK BIM survey 2019 findings Download: https://www.thenbs.com/knowledge/national-bim-report-2019 [Letzter Zugriff am: 14.06.2020].

Neo4j (2020a). 10.15. Fingerprinting - Chapter 10. Utility Functions. URL: http://neo4j-contrib.github.io/neo4j-apoc-procedures/3.5/utilities/fingerprinting/ [Letzter Zugriff am: 14.06.2020].

Neo4j (2020b). Cypher Query Language Developer Guides & Tutorials. URL: https://neo4j.com/developer/cypher-query-language/ [Letzter Zugriff am: 14.06.2020].

Niknam, M. & Karshenas, S. (2017). A shared ontology approach to semantic representation of BIM data. *Automation in Construction* 80, S. 22–36.

N+P Informationssysteme GmbH (2018). Vergleich von open BIM, closed BIM, little BIM, big BIM und connected BIM. URL: https://blog.nupis.de/vergleich-open-closed-little-big-bim-connected-bim/ [Letzter Zugriff am: 14.06.2020].

Paar, C. & Pelzl, J. (2016). *Kryptografie verständlich: Ein Lehrbuch für Studierende und Anwender*. Berlin and Heidelberg: Springer Vieweg.

Pilehchian, B., Staub-French, S. & Nepal, M. (2015, 02). A conceptual approach to track design changes within a multi-disciplinary building information modeling environment. *Canadian Journal of Civil Engineering* 42.

Rau, S. (2019). Food Trust — Blockchain-Netzwerk zur Nahrungsmittelkontrolle. URL: https://blockchainwelt.de/food-trust-blockchain-netzwert-zur-nahrungsmittelkontrolle/ [Letzter Zugriff am: 14.06.2020].

SAS (2018). Blockchains - Was sie sind und warum man das wissen sollte. URL: https://www.sas.com/de_de/insights/analytics/blockchain.html [Letzter Zugriff am: 14.06.2020].

Schatz, K. (2020). Was bedeutet LOD/LOI? — BIM — Modellinhalte — Baunetz_Wissen. URL: https://www.baunetzwissen.de/bim/fachwissen/modellinhalte/was-bedeutet-lod-loi-5285890 [Letzter Zugriff am: 14.06.2020].

Schiller, K. (2019). Die Blockchain Typen im Überblick. URL: https://blockchainwelt.de/blockchain-typen-ueberblick/ [Letzter Zugriff am: 14.06.2020].

SelfKey (2020). Understanding Public vs. Private Blockchain. URL: https://selfkey.org/understanding-public-vs-private-blockchain/ [Letzter Zugriff am: 14.06.2020].

Serlo (2020). Caesar-Verschlüsselung. URL: https://de.serlo.org/informatik/verschluesselung/caesar-verschluesselung-48121 [Letzter Zugriff am: 14.06.2020].

Shelden, D., Brouwer, J., Pauwels, P., Afsari, K., Sparks, D., McGinley, T. & Stefanescu, D. (2020). Technical Room 2: IFC JSON and IFC HDF5 (IFC Binary) [Video File]. vimeo. URL: https://vimeo.com/423525651 [Letzter Zugriff am: 14.06.2020].

Softtech (2020). Was ist BIM? URL: https://www.softtech.de/service/was-ist-bim [Letzter Zugriff am: 14.06.2020].

Sommer, H. (2016). *Projektmanagement im Hochbau: Mit BIM und Lean Management* (4. Auflage Aufl.). Berlin, Heidelberg: Springer Vieweg.

STEP Tools, Inc. (2019). Introduction to Standard Data Access Interface (SDAI). URL: http://www.steptools.com/stds/step/sdai.html [Letzter Zugriff am: 14.06.2020].

Stöltzel, Thomas und Zander, F. (2010). Skytale. URL: http://www.mathe.tu-freiberg.de/~hebisch/Praktikum10-4/skytale.html [Letzter Zugriff am: 14.06.2020].

Tauscher, E., Bargstädt, H. & Smarsly, K. (2016). Generic BIM queries based on the IFC object model using graph theory.

Technische Bauaufsicht der Stadt Landsberg am Lech (2020, Mai). Fragen zum Baugenehmigungprozess. Persönliche Kommunikation.

Tekla (2019). Open BIM: Offener Datenaustausch für optimale Zusammenarbeit. URL: https://www.tekla.com/de/bim/open-bim-offene-zusammenarbeit [Letzter Zugriff am: 14.06.2020].

Terfehr, S. (2019). BIM im Baugenehmigungsverfahren. URL: https://www.build-ing.de/fachartikel/detail/bim-im-baugenehmigungsverfahren/ [Letzter Zugriff am: 14.06.2020].

Trimble MEP (2020). LOD - Fakten, die Sie kennen sollten — Trimble. URL: https://mep.trimble.de/blog/LOD-Fakten-die-sie-wissen-sollten [Letzter Zugriff am: 14.06.2020].

xBIM (2016). xBIM WeXplorer. URL: http://docs.xbim.net/XbimWebUI/ [Letzter Zugriff am: 14.06.2020].

xbim (2020a). Basic model operations. URL: https://docs.xbim.net/examples/basic-model-operations.html [Letzter Zugriff am: 14.06.2020].

xbim (2020b). xbim toolkit - making building information flow. URL: https://docs.xbim.net/ [Letzter Zugriff am: 14.06.2020].

Xu, S. (2018). Investigation of graph-databases for storing and analysing building models. Masterarbeit, Technische Universität München.

Ye, Z., Yin, M., Tang, L. & Jiang, H. (2018). Cup-of-Water Theory: A Review on the Interaction of BIM, IoT and Blockchain During the Whole Building Lifecycle. In: J. Teizer (Hrsg.), *Proceedings of the 35th International Symposium on Automation and Robotics in Construction (ISARC)*. International Association for Automation and Robotics in Construction (IAARC).