Lehrstuhl für Computergestützte Modellierung und Simulation
TUM School of Engineering and Design
Technische Universität München

Modellbasierte Unterstützung der Verkehrssicherheitsarbeit in der Straßenplanung

Wissenschaftliche Arbeit zur Erlangung des Grades

Master of Science (M.Sc.)

an der TUM School of Engineering and Design der
Technischen Universität München.

Betreut von Prof. Dr.-Ing. André Borrmann
Simon Vilgertshofer M.Sc.
Lehrstuhl für Computergestützte Modellierung und Simulation

Christian Peetz Dipl.-Ing.
Landesbaudirektion Bayern

Eingereicht von Hans Lauro Schnittger [redacted]
[redacted]

Eingereicht am 30.09.2022

Abstract

This thesis deals with the model-based support of the road safety work in road design and investigates the benefits of Building Information Modeling (BIM)-based code compliance checking in combination with virtual reality for safety auditing. The goal is to demonstrate the benefits and limitations of automated detection and visualization of safety-relevant criteria using a case study with a certified safety auditor. On the basis of a selection of deficits, the necessary boundary conditions are elaborated that must be complied with in the modelling of the BIM model in order to be able to automatically read out required data such as geometries or attributes and thus create an efficient workflow that can be combined with the methodology of Landesbaudirektion Bayern (LBD)s System Bayerisches Straßeninformationssystem (BAYSIS). For this purpose, an insight into the techniques of BIM-based code compliance checking is given, and it is examined how Industry Foundation Classes (IFC) 4.3 can improve data discovery and shown how BIM technology in general has developed in traffic planning so far. The compliance tests are carried out with programs written in Java and Python via the Solibri-Application Programming Interface (API), the model is modified using the IFC-authoring tool IfcOpenShell, and it is documented how the result can be used in a virtual walk-through using VRex, interacted with and converted into a BAYSIS-compliant protocol.

Zusammenfassung

Die vorliegende Thesis befasst sich mit der modellbasierten Unterstützung der Verkehrssicherheitsarbeit in der Straßenplanung und untersucht den Nutzen von BIM-basiertem Code Compliance Checking in Kombination mit Virtual Reality für das Sicherheitsaudit. Das Ziel ist es, anhand eines Fallbeispiels mit einem zertifizierten Sicherheitsauditor die Vorteile und Grenzen der automatisierten Erkennung und Visualisierung sicherheitsrelevanter Kriterien zu demonstrieren. Anhand einer Auswahl von Defiziten werden die notwendigen Randbedingungen ausgearbeitet, die bei der Modellierung des BIM-Modells beachtet werden müssen, um erforderliche Daten wie Geometrien oder Attribute automatisiert auslesen zu können und somit einen effizienten Workflow zu erschaffen, der sich mit der Methodologie von BAYSIS der LBD kombinieren lässt. Hierzu wird ein Einblick in die Techniken des BIM-basierten Code Compliance Checkings gegeben und untersucht, wie IFC 4.3 die Datenermittlung verbessern kann und gezeigt wie sich generell die BIM-Technologie in der Verkehrsplanung bisher entwickelt hat. Die Compliance Tests werden mit in Java und Python geschriebenen Programmen über die API von Solibri durchgeführt, die Modifizierung des Modells erfolgt mittels des IFC-authoring tools IfcOpenShell und es wird dokumentiert, wie das Ergebnis in einer virtuellen Begehung mittels VRex genutzt, damit interagiert und in ein BAYSIS-konformes Protokoll überführt werden kann.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Algorithmenverzeichnis

Abkürzungen

ACCC	automated code compliance checking
API	Application Programming Interface
BAYSIS	Bayerisches Straßeninformationssystem
BCF	BIM Collaboration Format
BIM	Building Information Modeling
CAD	Computer-aided Design
CDE	common data environment
GUID	Globally Unique Identifier
IFC	Industry Foundation Classes
LBD	Landesbaudirektion Bayern
LOD	Level of Development
LOG	Level of Geometry
LOI	Level of Information

1. Einführung

1.1 Motivation

Die deutsche Bauindustrie befindet sich im Wandel. Building Information Modeling (BIM), also das Arbeiten mit einem Modell, das nicht primär durch seine geometrischen, sondern durch seine semantischen Aspekte geprägt ist, wird für die Beteiligten im AEC-Sektor (Architecture, Engineering and Construction) immer geläufiger. Im Masterplan BIM Bundesfernstraßen, der durch das Bundesministerium für Verkehr und digitale Infrastruktur veröffentlicht wurde, ist ein Stufenplan erstellt worden, der eine verpflichtende Einführung von BIM in allen Bundesfernstraßenbauten bis 2025 als Ziel setzt. International gesehen sind wir hierbei bei weitem keine Vorreiter. So hat das Vereinigte Königreich bereits 2016 BIM Level 2, das eine volle Kollaboration mit 4D und 5D BIM bedeutet (Bauablaufplanung/ Zeit- und Kostenfaktoren), für alle öffentlichen Projekte vorgeschrieben. Ebenso sind die skandinavischen Länder an der Führungsspitze. Finnland hat BIM seit 2002 eingeführt und bereits 2007 benutzten 93% der Architektur- und 60% der Ingenieurbüros BIM in ihrer Arbeitsroutine. Auch der Pionier der BIM-Anwender, die USA, die seit den Siebziger Jahren bereits begannen, die ersten Schritte mit BIM zu machen, zeigt einen starken Anwuchs in der BIM-Adaptierung in den letzten Jahren (UNITEDBIM, 2022).

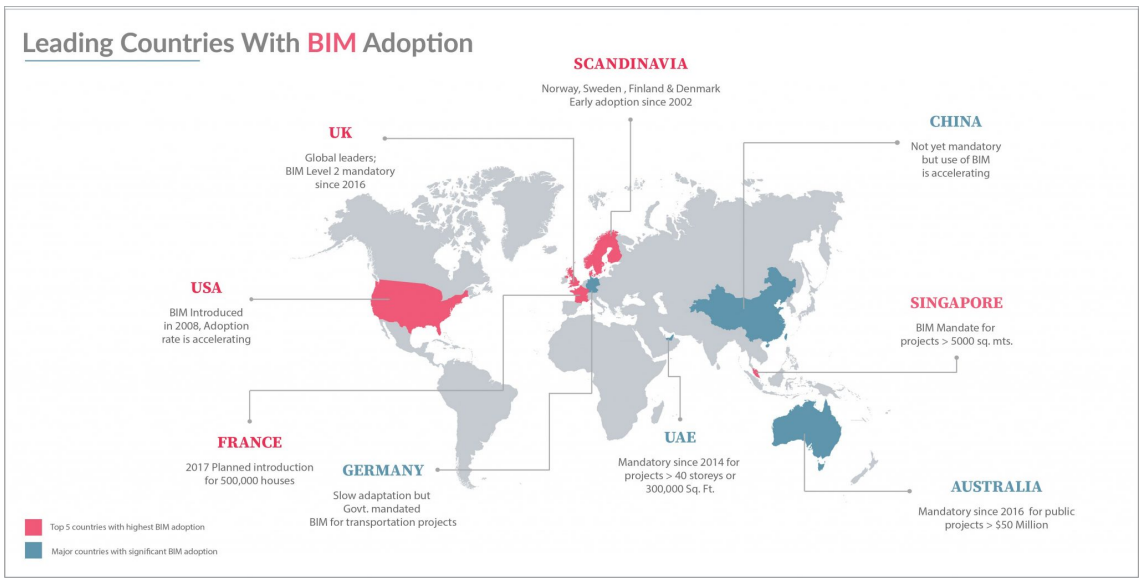


Abbildung 1.1: Die Übernahme von BIM aus globaler Sicht (UNITEDBIM, 2022)

Da der Bauprozess nicht wie die Fahrzeugherstellung komplett standardisiert werden kann, sondern von vielen lokalen Faktoren abhängig ist, die sich von Projekt zu Projekt und von Region zu Region unterscheiden, können nicht alle Erfahrungswerte aus dem Ausland einfach übernommen werden. Eigene Forschungen müssen betrieben werden, um Arbeitsprozesse und Standards zu entwickeln, die der hiesigen Industrie entspre-

chen. Einerseits werden so die Grenzen des Machbaren ausgetestet und andererseits bekommen die Beteiligten einen wachsenden Sinn für die Möglichkeiten, die BIM ihnen bereitstellt. So bietet die staatliche Planung im Infrastrukturbereich ein ideales Feld, um diese Vorteile zu ermitteln. Das Einsparungspotenzial ist bei den meist sehr großen Projekten enorm und durch die Zusammenarbeit der Ämter lassen sich Standards gut etablieren. Um zu diesem Wissenszuwachs beizutragen und dabei zu helfen den Stufenplan des Bundesministeriums einzuhalten, wurde in dieser Arbeit der Prozess des Sicherheitsaudits untersucht und dafür das Ziel verfolgt herauszufinden, ob und wie dieser durch die Nutzung von BIM profitieren kann, sowohl im Aspekt der Arbeitserleichterung, der Präzision als auch in der Einsparung von Zeit. Die Arbeit eines Sicherheitsauditors ist vergleichbar mit der eines Prüfstatikers und setzt eine korrekte und hinreichende Informationsgrundlage voraus. Oft sind jedoch jahrelange Erfahrung nötig, um anhand von 2D-Plänen und PDFs ein korrektes Urteil treffen zu können. Um den Auditor hierbei zu unterstützen, wurde demnach untersucht, wie das digitale 3D-Modell ihm seine Arbeit durch die Möglichkeiten, die BIM bietet, wie die automatisierte Konflikterkennung und das Arbeiten in der virtuellen Realität, erleichtert.

1.2 Umfang

Die Arbeit befasst sich mit der Entwicklung eines Modellinhaltsstandards, der eine Teilautomatisierung des Audits ermöglicht und mit dem Aufbau der Programme, die die Überprüfung durchführen. Weiterhin wird untersucht, welchen Mehrwert eine Konfliktsignalisierung mittels BCF-kompatiblen Konfliktmanagement im Vergleich zu im IFC-Schema hinzugefügter Hinweisobjekte hat und wie eine solche Modellveränderung erzielt werden kann. Im Anschluss werden die Methoden in einer Fallstudie getestet und ausgewertet.

1.3 Structure of Work

In Kapitel 2 wird ein grundsätzliches Verständnis von IFC vermittelt und ein Einblick in seine Entwicklungsgeschichte, seinen Aufbau und die Kernprinzipien gegeben. Danach werden die unterschiedlichen Möglichkeiten genannt, die es erlauben dieses Schema auf simple Weise zu bearbeiten und ihre Vor- und Nachteile gegenübergestellt. In Kapitel 3 wird das Thema Code Compliance Checking bearbeitet und nach einer Darstellung der Grundprinzipien dieser Checks auf ein paar der relevantesten Varianten der Regelerstellung eingegangen und für die jeweiligen Methoden exemplarische Softwareanwendungen genannt. Es wird die Plattform BAYSIS vorgestellt, die als digitale Schnittstelle und zentrale Informationsplattform für die LBD dient, sowie auf die Modellqualität eingegangen. Danach wird das Konzept des Level of Developments (LOD) erläutert, seine Bedeutung für den Bauablauf und der Zusammenhang zu den Auditphasen erklärt. Weiterhin wird auf Themen des digitalen Bauens wie BCF, common data environment (CDE), Augmented Reality (AR) und Virtual Reality (VR) eingegangen. Kapitel 4 dreht sich um das Thema Verkehrssicherheit an sich und um BIM an der LBD. Hierfür wird auf die geschichtlichen Hintergründe der

Verkehrssicherheit und auf ausgewählte Themen der Verkehrssicherheitstheorie eingegangen. Es werden die Richtlinien erläutert, die die Rahmenbedingungen für einen sicheren Verkehrsweg definieren, der Prozess des Sicherheitsaudits geschildert und die Kerndefizite veranschaulicht. Zusätzlich werden die Kernpunkte des BIM-Leitfadens zusammengefasst, um einen Überblick über die derzeitige Implementierungsphase in der LBD zu schaffen und die Arbeit dazu in einen Kontext zu setzen. Im fünften Kapitel geht es schließlich um das Konzept, mit der aus einem Modell Daten erhalten werden, die den Auditor zuverlässig unterstützen können. Hierfür wird erklärt, wie ein Teilmodell vorbereitet wurde, um den ausgearbeiteten Informationsanforderungen zu entsprechen. Es wird auf die einzelnen Arbeitsschritte eingegangen und die Lösungen für die Ex- und Importkonflikte präsentiert, die zwischen Civil3D und Revit auftraten und gezeigt, wie die Eigenschaftssätze eingepflegt werden konnten. Danach werden die Programme durchgegangen, die zur Modellanalyse entwickelt wurden und der Unterschied zwischen dem Konfliktmanagement mit BCF und mit einem angepassten IFC-Schema verdeutlicht. Diesbezüglich wird das Pythonscript, das mittels IfcOpenShell die Dateien verändert, erläutert. Kapitel 6 ist die Fallstudie, die mit den zuvor besprochenen Methoden ein Audit beschreibt und die einzelnen Schritte durchgeht. Hierbei wird das vorbereitete Modell von den Auditoren mit den automatisierten Prüfungen analysiert, in der virtuellen Realität begangen und das Konfliktmanagement mittels Modellveränderung und BCF-Kollaboration verglichen. Es wird beschrieben, worin die Vorteile der Checks und der VR-Anwendungen liegen, wo es Schwierigkeiten gab und welche Limitationen sich aufgezeigt haben. Das letzte Kapitel fasst alle Arbeitsschritte zusammen und zieht das Fazit aus den Resultaten. In einer Diskussion werden hierfür die Ergebnisse aufgegriffen und bewertet. Zuletzt erfolgt ein Ausblick auf die kommenden Entwicklungen und die notwendigen Verbesserungen, die für einen effektiven Einsatz der modellbasierten Unterstützung der Verkehrssicherheitsarbeit in der Straßenplanung notwendig sind.

2. IFC

Industry Foundation Classes ist eine international standardisierte open source Datenstruktur, entwickelt für Building Information Modeling (BIM), um alle Informationen eines Bauwerks modellieren und software-unabhängig teilen zu können. Also nicht nur visuelle Designaspekte, sondern auch Informationen, die für strukturelle Berechnungen, zeitliche Koordinierungen des Baumanagements und viele weitere Bereiche relevant sind. Aufgrund seiner weiten Verbreitung und Standardisierung ist IFC für die Nutzung durch die öffentliche Hand unerlässlich geworden (BORRMANN et al., 2020). Die in den späteren Kapiteln aufgeführten Modellmanipulationen setzen ein grundlegendes Verständnis dieses Datenmodells voraus, da die Modelle in diesem Schema vorliegen. Im Folgenden wird ein Einblick in die Entwicklungsgeschichte, den aktuellen Aufbau und ein Ausblick auf die weitere Entwicklung von IFC gegeben.

2.1 Hintergrund

Die Entwicklung der Computer-aided Design (CAD)-Systeme und Standards kann nach Laakso und Kiviniemi in drei Abschnitte geteilt werden. Erstens, der Beginn der Nutzung von CAD-Systemen in den fünfziger Jahren, die ausschließlich auf die jeweiligen Projekte ausgerichtet waren. Zweitens, die Einführung herstellerneutraler CAD Standards wie Initial Graphics Exchange Specification (IGES), das 1980 veröffentlicht wurde und drittens, die Entwicklung von Standard for the Exchange of Product Model Data (STEP) in 1984, definiert in der ISO-Norm 10303. Ein Standard, der sich für die Nutzung unterschiedlichster Industriezweige eignete und die Grundlage für die Entwicklung der EXPRESS-Sprache bildete, eine deklarative Sprache, die im Gegensatz zu existierenden Sprachen geeignet war ein Datenmodell zu definieren (LAAKSO & KIVINIEMI, 2012) (BORRMANN et al., 2020). Im Zuge dieser Entwicklung entstand ein Anwendungsprotokoll, genannt AP 225, das für eine einheitliche Modellierung von Gebäuden sorgen sollte, jedoch von der Bauindustrie wegen mangelnder Effizienz nicht angenommen wurde. Um dieses Problem zu lösen, wurde 1994 die International Alliance for Interoperability (IAI) gegründet, die 2005 in buildingSMART umbenannt wurde und der Schöpfer von IFC ist, dessen erste Version 1997 veröffentlicht wurde (BORRMANN et al., 2020). IFC basiert also auf den grundlegenden Prinzipien von STEP, welche durch die EXPRESS-Sprache definiert ist und im Teil 11 des STEP-Standards beschrieben ist.

Versionsverlauf

Erst in IFC 4.3, das zur Zeit dieser Arbeit am 07.03.2022, offiziell in der International Organization for Standardization (ISO)-Norm aufgenommen wurde, werden erstmals standardisierte Attribuierungen von Infrastruktureigenschaften ermöglicht. Dieser Entwicklung gehen viele Schritte voraus und die unterschiedlichen Versionen können entsprechend der Grafik zeitlich einsortiert werden.

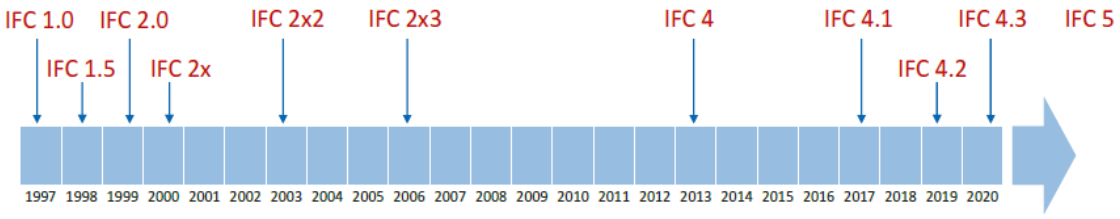


Abbildung 2.1: Versionsverlauf von IFC (BORRMANN et al., 2020)

IFC 1.0 legte den Grundstein für das heutige komplexe Datenmodell, war jedoch seinerseits noch sehr rudimentär aufgebaut und hauptsächlich auf architektonische Aspekte reduziert. Es beinhaltete zwei Architektur-, zwei HLS-Design-, zwei Bauleitungs- und einen Gebäudemanagementprozess. Diese Version wurde von gerade mal 17 Firmen implementiert, diente aber als einen ersten Testlauf und half IFC 1.5 stabiler zu gestalten, dessen überarbeitete Version IFC 1.5.1 1998 auf den Markt kam (LAAKSO & KIVINIEMI, 2012). Der aktuell am weitesten verbreitete Standard ist das 2006 veröffentlichte Schema IFC 2x3, welches jedoch durch IFC 4 und 4.3 stetig ersetzt wird. Das Schema befindet sich unter konstanter Entwicklung und Revision.

2.2 Aufbau

Die IFC-Datenstruktur ist in vier Schichten (engl. Layers) aufgebaut. Von oben nach unten sind diese: Domain, Interoperability, Core und Resource Layer (siehe Abb. 2.2), welche jeweils nur von oben nach unten Bezug nehmen können (BORRMANN et al., 2020).

Der Domain Layer beinhaltet Entitäten der unterschiedlichen Spezialisierungsbereiche. Diese können von keiner anderen Schicht aus referenziert werden (BUILDINGSMART, 2022b). Der IFC 4.3 Release fügte die drei neuen Klassen Road, Ports and Waterways und Rail hinzu (siehe Abb. 2.2). IfcTunnel ist ein Forschungsobjekt der TUM, war jedoch kein Bestandteil der Einreichung am 07. März 2022 (CMS, 2014). Der Domain Layer kann auf alle Klassen auf Höhe oder unterhalb des Interoperability Layers zugreifen.

Der Shared oder Interoparability Layer dient als Zwischenebene des Domain und Core Layers, mittels dem Beziehungen und spezifischere Objekte definiert werden können, die von mehreren Bereichen geteilt werden. So können sich zum Beispiel IfcArchitectureDomain und IfcConstructionMgmtDomain die Klasse IfcDoor teilen, die in IfcSharedBldgElements enthalten ist, einem Subtyp von IfcBuildingElement, der in der IfcProductExtension des Kernels definiert ist (BUILDINGSMART, 2022c).

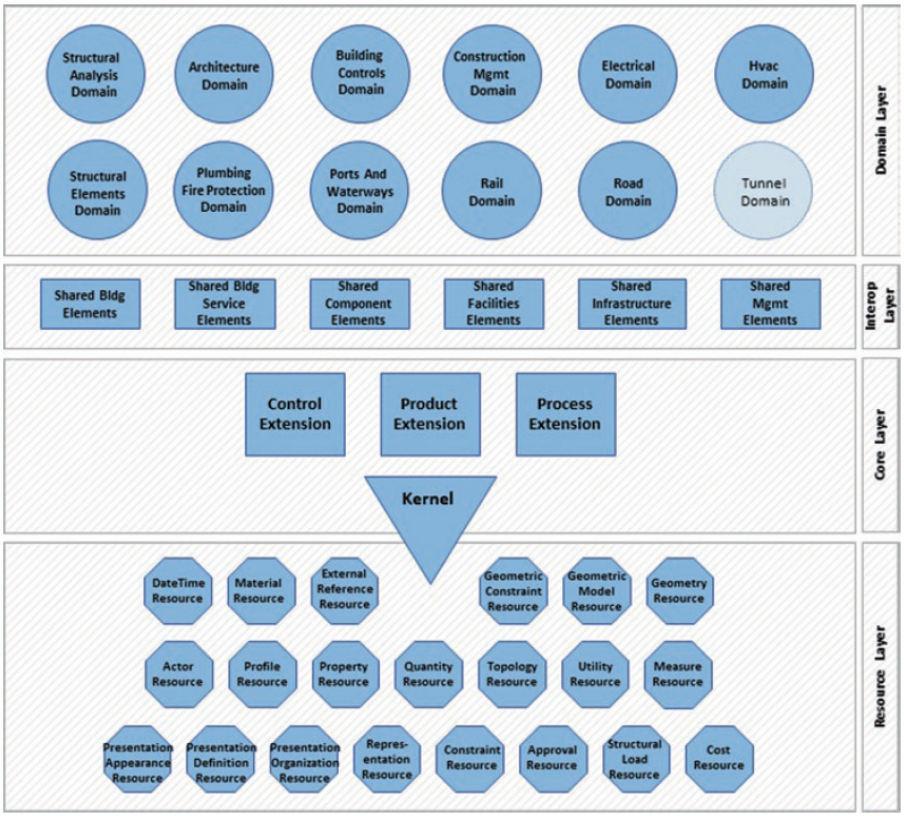


Abbildung 2.2: IFC Layer (BORRMANN et al., 2020)

Der Core bildet die Grundstruktur in der die grundlegenden Beziehungen und allgemeinen Konzepte für alle weiteren Spezialisierungen liegen. Alle Entitäten, die im Core Layer oder darüber definiert werden, werden von IfcRoot abgeleitet (BUILDINGSMART, 2022a), das die Globally Unique Identifier (GUID), den Namen, die Beschreibung und die Besitzerhistorie (owner history) enthält. Alle Entitäten, die im Kernel definiert sind, sind Subtypen von IfcRoot und können unabhängig genutzt werden wohingegen alle, die im Resource schema definiert sind nicht unabhängig genutzt werden können (BUILDINGSMART, 2022d). Weitere Bestandteile des Kernels sind IfcProductExtension, IfcControlExtension und IfcProcessExtension. IfcProductExtension definiert die Repräsentation des physischen Objektes weiter. Es enthält Konzepte für zum Beispiel die Definition der Baustelle, des Raumes oder der Anordnung in einer Gitternetzstruktur. IfcControlExtension enthält die Klassen für Kontrollinstanzen. IfcControll dient dabei als Klasse für einschränkende oder kontrollierende Konzepte. Beispiele hierfür sind gesetzliche Regulierungen oder Produktanforderungen. IfcProcessExtension stellt eine der Kernideen des IFC-Schemas dar, nämlich Prozesse in logischer Sequenz zu strukturieren und zu planen und sie mit den den entsprechenden Ressourcen zu verknüpfen (BUILDINGSMART, 2022a) (BORRMANN et al., 2020).

Wie bereits zuvor erwähnt, können die Klassen des Resource Layers nicht eigenständig genutzt werden. Sie befinden sich im untersten Layer und können somit nicht auf das im Kernel enthaltene IfcRoots verweisen und besitzen daher keine Identität. Auf sie kann nur direkt oder indirekt durch die darüberliegenden Klassen verwiesen werden, die so

die grundlegenden Datenstrukturen verwenden können (BUILDINGSMART, 2022e). Gut veranschaulicht wird dieses Konzept durch die Material- oder Geometry- Resource. Alle Klassen darüber können auf ein Material verweisen, welches dadurch Bedeutung erhält, jedoch alleine für sich in einem Modell keinen Sinn machen würde (BORRMANN et al., 2020).

Vererbungshierarchie

Neben dem Aufbau ist die Vererbungshierarchie von IFC für das Verständnis des Schemas wichtig. Hier bildet IfcRoot die Wurzel des Vererbungsbaums, das ausgehend vom Core Layer für alle Entitäten, die nicht im Resource layer liegen den Supertype bildet. Jede Entität wird mittels der GUID identifiziert, kann einem Eigentümer zugeordnet, gelabelt und beschrieben werden. Jede Klasse, die sich daran anschließt erbt nun diese Attribute und gibt seine eigenen ebenso weiter, sodass die letzte Instanz in der Hierarchie Zugriff auf alle darüberliegenden Attribute hat (siehe Abb. 2.3).

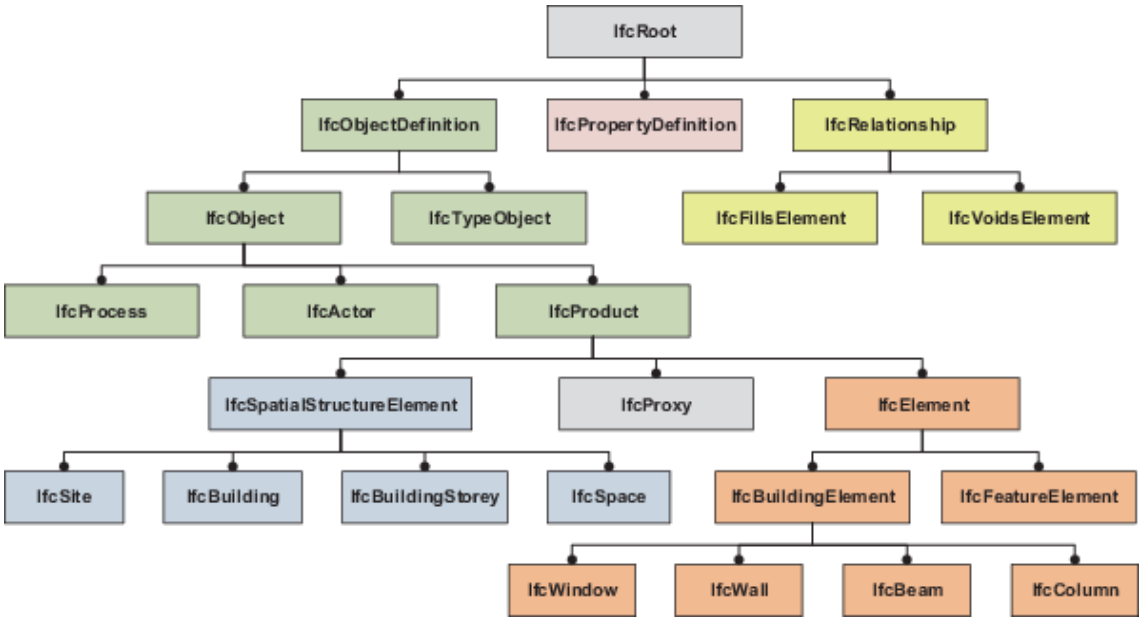


Abbildung 2.3: Vererbungshierarchie (BORRMANN et al., 2020)

Direkt an IfcRoot schließen die Klassen IfcPropertyDefinition, IfcRelationship und IfcObjectDefinition an. Ersteres dient der Beschreibung von Attributen, für die im IFC-Schema keine vorgegebene Definition vorliegt. So gibt es festgelegte Attribute, die zur Bestimmung der Höhe und Breite eines Fensters dienen, aber keine, die die Querneigung einer Straße bestimmen. Hierzu können IfcPropertySingleValue erstellt werden, die entsprechenden IfcPropertySet untergeordnet sind. Für das Beispiel der Straße könnte der Name des IfcPropertySingleValue "Querneigung"lauten und darin den Datentyp IfcRatioMeasure mit dem Wert "5.2ünd der Einheit "%ënthalten. Zusammen mit anderen Eigenschaften wie Oberflächenmaterial und Radius würde dies nun unter die erstellte Kategorie "LBD Straßenattributeëinsortiert werden. Für oft vorkommende Eigenschaften gibt es hierfür jedoch noch Eigenschaftssätze, die für allgemeine Fälle vordefiniert sind und mit dem Anhang "Common"gekennzeichnet sind. Im Falle des genannten Beispiels wäre das

Pset_RoadDesignCriteriaCommon, das für die Steigung eine Vorlage enthält. Seit IFC 4 gibt es auch das Konzept der IfcPropertySetTemplate, das aus mindestens einem IfcPropertyTemplate besteht, eine Vorlage, die im Projekt eine einheitliche Bezeichnung erleichtern soll.

2.3 IFC 4.3

Durch die Arbeit im Infrastrukturbereich spielt die Entwicklung von IFC 4.3 eine wichtige Rolle. Diese Version wurde am 07. März 2022 veröffentlicht und ist seitdem offizieller Standard von buildingSmart. Seit IFC 4.1 wurde das Schema vermehrt für die Bereiche Gleis-, Tunnel-, Straßen-, Hafen- und Wasserstraßenbau überarbeitet. Mit der aktuellen Version wurden nun sogar drei komplett neue Domänen erstellt, die diese Bereiche besser repräsentieren sollen. IfcPortsAndWaterwaysDomain dient hierbei der Klassifizierung von Hafenanlagen und allen Konstruktionen, die im Wasser angelegt werden. IfcRailDomain dient der Spezifizierung des Gleisbaus und IfcRoadDomain dem Straßenbau.

Mit IFC 4.3 wird auch die Klasse IfcLinearPlacement weiter ausgebaut, die in IFC 4.1 eingeführt wurde. Ähnlich zu IfcLocalPlacement dient es der Platzierung eines Objekts im Raum, wird aber statt mit Koordinaten mittels linearer Referenzierung platziert. Um dies besser zu verstehen, werden die beiden Konzepte im Folgenden näher erklärt.

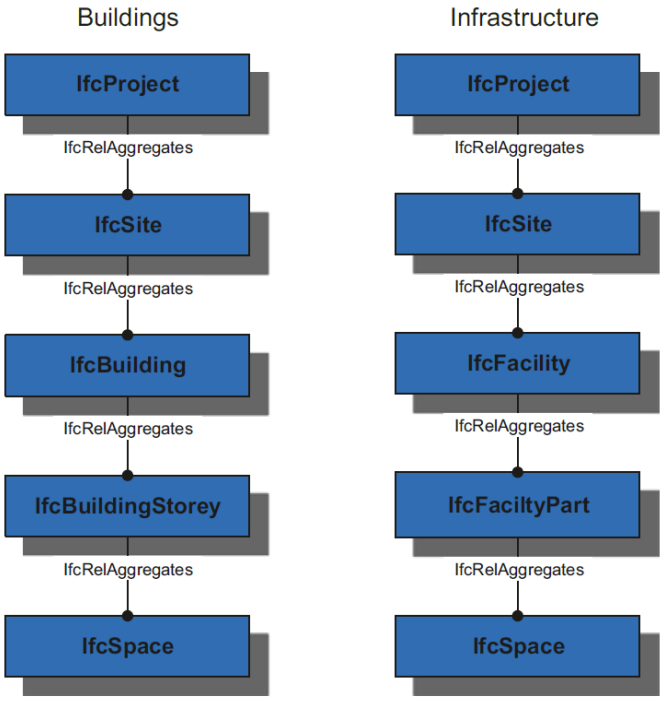


Abbildung 2.4: IFC Layer (BORRMANN et al., 2020)

Ein konventionelles Hochbauprojekt besitzt eine Raumhierarchie-Struktur, bei der das Projekt durch IfcSite relativ zum globalen Koordinatensystem ausgerichtet wird (siehe 3.8). Diesem Prinzip wird bis zur kleinsten Entität gefolgt, bedeutet, dass der Raum

(IfcSpace) relativ zum Stockwerk, dieses relativ zum Gebäude und dieses wiederum relativ zur Baustelle platziert wird. Hierfür wird die bereits erwähnte Klasse IfcLocalPlacement verwendet. Diese erstellt im Falle einer Platzierung im dreidimensionalen Raum mittels IfcAxis2Placement3D ein Koordinatensystem, dessen Ursprung relativ zum gegebenen Koordinatensystem gesetzt und dessen Achsenausrichtung durch eine Richtungsangabe der z- und x-Achse definiert wird (siehe 2.5).

Für den Infrastrukturbereich wurde jedoch die Raumhierarchie anders definiert. Statt über Gebäude und Stockwerke ist das Projekt in Einrichtungen (IfcFacility) und Einrichtungsteile (IfcFacilityPart) gegliedert, was eine generisch anwendbare Klassifizierung ermöglicht. Diese wurden in IFC 4.2 eingeführt und bilden den Supertype für nützliche Entitäten wie IfcRoad und IfcRoadPart. Dementsprechend werden Objekte nicht in Stockwerke platziert, sondern beispielsweise entlang vieler Kilometer langer Gleisbauprojekte. Dementsprechend platziert IfcLinearPlacement Objekte in einem definierten Abstand und Rotation entlang einer Kurve. Die Klasse verweist hierbau auf die Attribute IfcCurve, das die Kurve darstellt, IfcDistanceExpression, das den Abstand von der Kurve definiert und IfcOrientationExpression, das die Richtung der lateralen und vertikalen Achse definiert. Hierdurch können, wie in Abbildung 2.5 gezeigt, Hinweisschilder relativ zu einer Straße ausgerichtet werden.

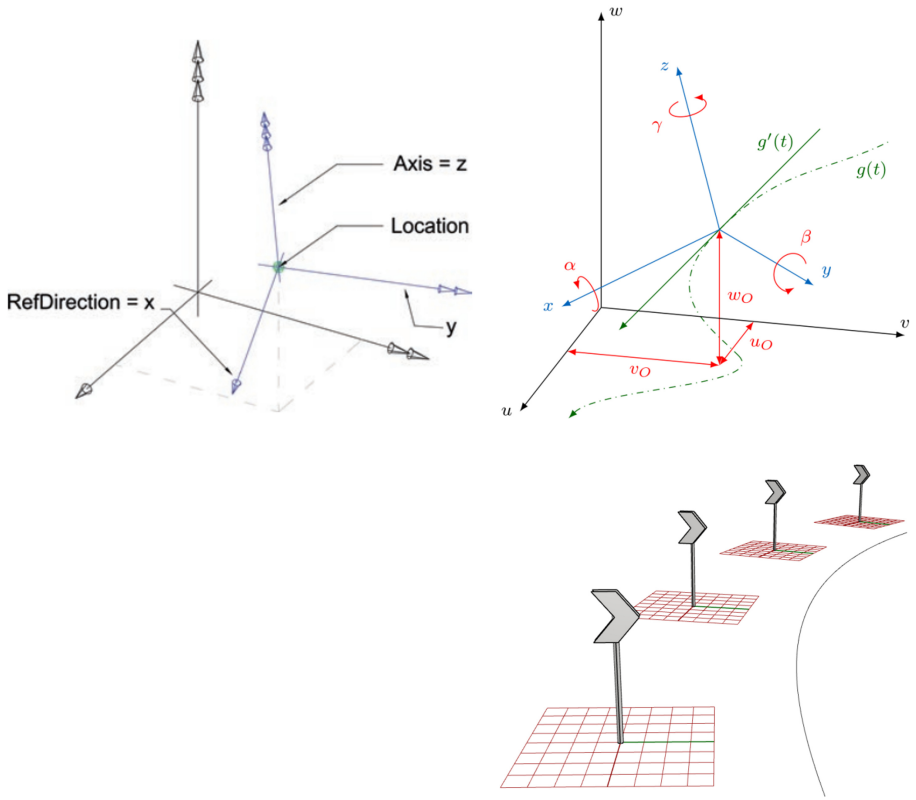


Abbildung 2.5: [links] Illustration von Local (BORRMANN et al., 2020) und [rechts] Linear Placement (JAUD et al., 2021)

Die meisten gängigen Softareanwendungen sind jedoch noch nicht kompatibel mit IFC 4.3, weshalb dieser Standard vorerst nicht genutzt werden kann, außer mit wenigen Ausnahmen, die nur der Modellansicht dienen, wie der OpenIfcViewer. Eine Anwendung wie etwa Autodesk AutoCAD Civil3D wird demnach beim Entwurf ein Straßenobjekt nicht automatisiert als ein IfcRoadPart definieren, sondern als die generische Entität IfcBuildingElementProxy.

2.4 IFC authoring tools

Da es in dieser Arbeit um die Unterstützung des Auditors durch die Nutzung von virtueller Realität geht, stellte sich die Frage, was der beste Weg sei, um die erkannten Defizite möglichst gut darzustellen und gleichzeitig möglichst plattformunabhängig zu bleiben. Die Antwort war, das IFC-Modell direkt zu manipulieren, sodass eine Datei exportiert werden kann, die von jeder IFC-kompatiblen VR-Software importiert werden könnte. Um dies zu tun, boten sich mehrere Möglichkeiten beziehungsweise Softwareanwendungen an. Eine Option wäre das Programmieren eines Codes, der die jeweiligen Visualisierungen direkt in die IFC-Datei schreibt. Dies wäre zwar durchaus möglich, doch ist IFC sehr kompliziert und der Code wäre entsprechend aufwendig zu entwickeln, um möglichst alle Fälle und unterschiedlichen IFC-Schema abzudecken. Abgesehen davon, gibt es sehr nützliche IFC authoring tools, die diese Arbeit sehr viel angenehmer gestalten. Im Folgenden werden die unterschiedlichen Möglichkeiten vorgestellt und diskutiert.

JSDAI

JSDAI ist eine frei erhältliche Programmierschnittstelle (API), mit der objektorientierte Daten, die durch ein Expressdatenmodell definiert sind, also unter anderem Modelle, die dem STEP-Standard entsprechen, modelliert werden können. Von Vorteil ist, dass sie für JAVA ausgelegt ist, nachteilig ist jedoch, dass sie nicht auf IFC spezifiziert und dementsprechend auch nicht für die DIN EN ISO 16739-1, der aktuellen Norm seit IFC 4 ausgelegt ist (LKSoftWare, 2022). Abgesehen davon ist sie inkompatibel mit der Solibri API, auf welche im Kapitel 3 eingegangen wird.

Apstex

Apstex ist ein IFC-Framework, das ein IFC-toolkit besitzt, mittels dessen in Kombination mit Java IFC Building Information Modeling bearbeitet werden können. Für Studien- und Forschungszwecke bietet es eine kostenlose Lizens an, die jedoch nicht kommerziell genutzt werden darf. Apstex hat eine gut strukturierte Dokumentation, die IFC 2x3 bis IFC 4.2 unterstützt und somit sehr aktuell ist (Apstex, 2022). Auch hier bestand jedoch das Problem, dass die Solibri API, die Apstex-Bibliothek nicht akzeptierte.

IfcOpenShell

IfcOpenshell ist eine open source Software-Bibliothek, die es Nutzern erlaubt IFC Dateien zu bearbeiten. Auf diese kann per C++ oder Python zugegriffen werden. Der Vorteil hierbei

ist, dass IfcOpenShell frei verfügbar ist und die Bibliothek nur in den C++ oder Python Code importiert wird und nicht in den Java Code (IFCOPENSHELL, 2022). Durch die Nutzung eines Process- builders im Java Code, konnte somit das Problem der inkompatiblen Solibri API umgangen werden. Darauf wird im Kapitel 5 näher eingegangen.

3. Code Compliance Checking und digitales Bauen

Die folgenden Abschnitte geben einen Einblick in Code Compliance Checking, dessen Methoden genutzt werden, um Modelle auf Korrektheit und Einhaltung entsprechender Regulierungen zu prüfen und sich daher auch für die Verifizierung von Sicherheitsdefiziten einer Planung eignen. Die Checks können durch unterschiedliche Ansätze erstellt und umgesetzt werden, welche im Folgenden vorgestellt werden.

3.1 Einleitung

Grundsätzlich sind alle Compliance Checks nach dem gleichen Prinzip aufgebaut, welches sich in 4 Segmente einteilen lässt:

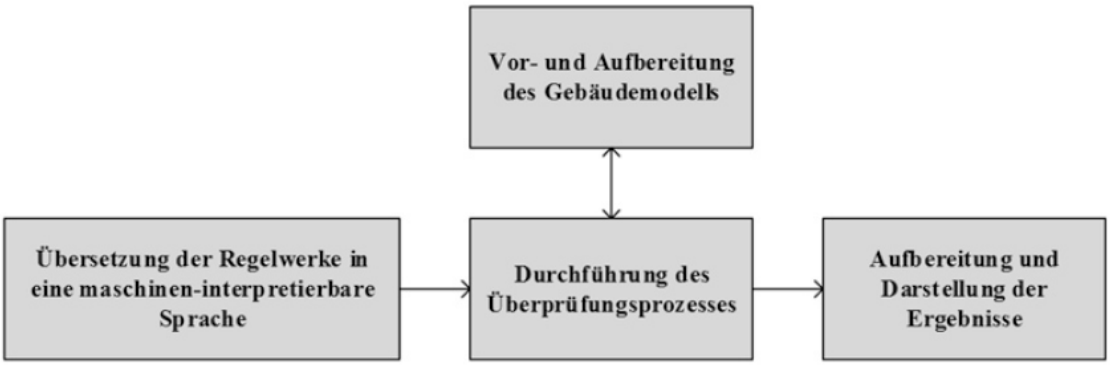


Abbildung 3.1: Aufbau eines ACCC (BORRMANN et al., 2020)

Im ersten Schritt müssen die Normen und Richtlinien in eine maschineneninterpretierbare Sprache übersetzt werden. Ein Beispiel wäre die maximale Höhe einer Bordsteinkante, dessen Wert dem Programm übergeben werden muss. An dieser Stelle treten für den Anwender die größten Unterschiede bezüglich der Nutzung der Applikation auf, da hier einerseits der Grad seines Einflussbereichs und die Art und Weise der Datenübergabe deutlich variieren können.

Die Aufbereitung des Gebäudemodells sorgt dafür, dass die entsprechenden vorhanden sind beziehungsweise korrekt ausgelesen werden können. So kann die Höhe der Bordsteinkante nur gefunden werden, wenn der Name korrekt geschrieben ist und der Wert existiert. Eine generelle Voraussetzung für ein aussagekräftiges Ergebnis ist ein korrektes Modell, eine IFC-Datei mit hoher Datenqualität und die Entsprechung an die Vorgaben des jeweiligen Level of Developments (LOD). Weiterhin sorgen zu große Modelle oft für Leistungsprobleme, weshalb eine Reduktion auf Teilmodelle nötig sein kann (C. F. D. PREIDEL, 2020).

Die Durchführung des Überprüfungsprozesses ist der Vergleich der Werte aus den Regelwerken mit den im Modell vorhandenen und unterscheidet sich hauptsächlich in seiner Transparenz. Da letztlich der Anwender Verantwortung über seine Arbeit trägt, ist es vorteilhaft nachvollziehen zu können, wie die Überprüfungen im Detail durchgeführt worden sind (BORRMANN et al., 2020).

Die Aufbereitung und Darstellung der Ergebnisse sind üblicherweise eine Mischung aus einer visuellen Hervorhebung der betroffenen Objekte beziehungsweise Bereichen und einer schriftlichen Beschreibung in Tabellen- oder Listenform. Konflikte können hierbei über Kanäle wie etwa BCF kommuniziert werden, in dieser Arbeit wird jedoch gezeigt, dass es auch noch weitere Möglichkeiten gibt.

Im Folgenden wird eine Auswahl der unterschiedlichen technischen Umsetzungen dieser Prinzipien vorgestellt und ihre Vor- und Nachteile diskutiert.

3.2 Methoden des Code Compliance Checking

3.2.1 Hartkodierte Tests

Solibri

Der Solibri Model Checker ist eine Java-basierte Software des finnischen Unternehmens Solibri, wurde im Jahr 2000 veröffentlicht und wird in dieser Arbeit für die Entwicklung der später aufgeführten Programme verwendet. Die kommerzielle Software beinhaltet einen Model Viewer, ist IFC-kompatibel und kommt mit einer Auswahl von über 50 Rulesets, mit denen Modelle auf unterschiedlichste Kriterien geprüft werden können. Neben diesen bereits vorhandenen Regeln bieten sie mit ihrer API eine große Bandbreite an Möglichkeiten neue Überprüfungsregeln oder Fenster mit selbst definierten Funktionalitäten zu entwerfen (C. F. D. PREIDEL, 2020).

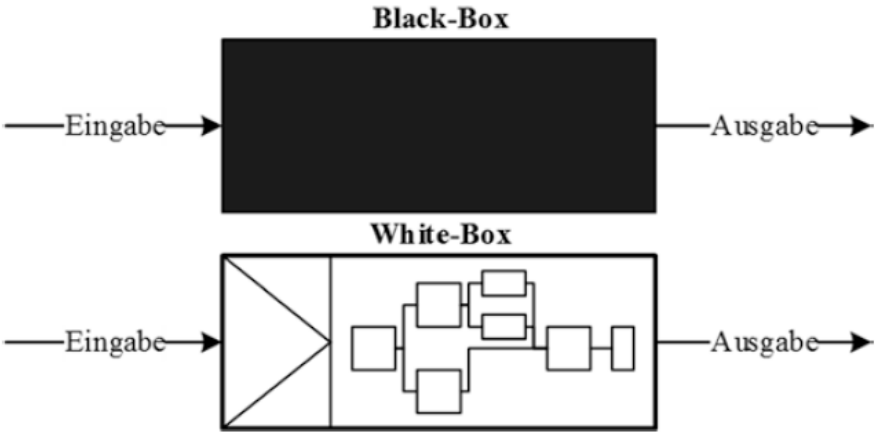


Abbildung 3.2: Veranschaulichung der Black-Box und White-Box Methode (BORRMANN et al., 2020)

Solibri ist eine weit verbreitete Anwendung für BIM-Projekte, hat eine sehr gut strukturierte API-Dokumentation sowie einen Kundensupport. Außerdem ermöglicht es die eigene Klassifikation von Modellen, was im Projektmanagement gerade für sehr große Projekte außerordentlich hilfreich sein kann. Nachteilig ist jedoch, dass Solibri keinen IFC-Export ermöglicht, bei sehr großen Projekten Leistungsprobleme zeigt und kostenpflichtig ist.

Die Software arbeitet mit fest implementierten (hartkodierten) Tests bedeutet, dass die Checks bis auf die variablen Parameter und Filter statische Programmroutinen sind, die keinen Einblick in die Prozesschritte oder die Algorithmen bieten. Obwohl das Ergebnis mit einem Text erläutert wird und somit einen gewissen Grad an Transparenz und Nachvollziehbarkeit bietet, fällt die Anwendung entsprechend der allgemeinen Systemtheorie in die Kategorie Black-Box-Prozess (BORRMANN et al., 2020). Im Gegensatz zu White-Box-Prozessen erlauben diese keinen Einblick in den Programmablauf (siehe Abb. 3.2), haben jedoch den Vorteil aufgrund der Geschlossenheit weniger fehleranfällig zu sein (BORRMANN et al., 2020). Ein weiterer Vorteil von fest implementierten Programmen ist, dass es praktisch keine Grenzen bezüglich ihrer Komplexität und Vielfalt gibt. Außerdem sorgt ihre statische Auslegung für einen Programmentwurf, der oft in mehreren Projekten Anwendung findet. Einmal entwickelt, ist die Überprüfung effizient und leicht durchzuführen. Nachteilig ist jedoch, dass für die Entwicklung der Tests Programmierkenntnisse nötig sind, die sehr viele Anwender in der Bauindustrie nicht haben.

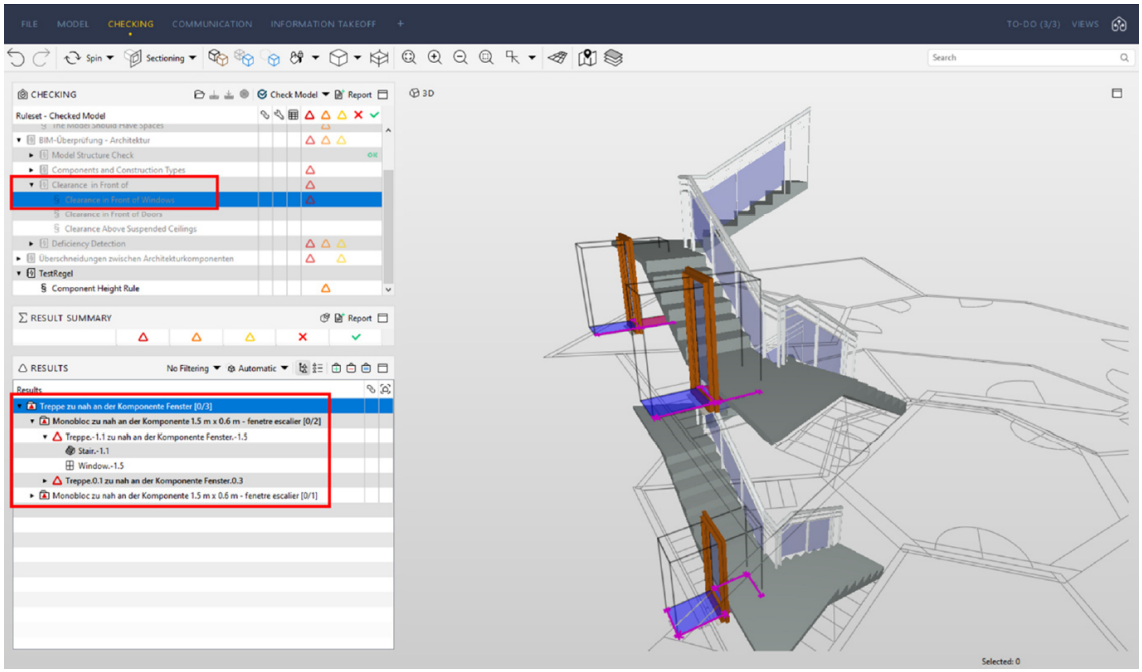


Abbildung 3.3: Beispiel einer Konformitätsprüfung (Freiraum vor Fenster)

Corenet

Eine weitere Anwendung, die hartkodierte Programmroutinen verwendet, ist die Plattform Construction and Real Estate Network (CORENET). Sie ist eine der bekanntesten Anwendungen im Bereich des automated code compliance checking (ACCC) und gleichzeitig wie BAYSIS eine staatliche Plattform. Sie wurde 1995 von der Building Construction Authority (BCA) in Singapur eingeführt und ist einer der ersten digitalen Plattformen für die Einreichung von Bauplanungsdokumenten. Ziel ist es den Genehmigungsprozess mithilfe digitaler Methoden zu optimieren. Die erste Anwendung zur Überprüfung von Vorschriften in Plänen war BP-Expert und war ausschließlich auf 2D-basierte Zeichnungen ausgelegt. Erst in 1998 wurde das System grundlegend neu aufgebaut und auf IFC ausgelegt und somit die Konformitätsprüfung von 3D-Modellen ermöglicht (C. F. D. PREIDEL, 2020). 2002 wurde die Prüfanwendung unter dem Namen e-PlanCheck veröffentlicht und bildet bis heute zusammen mit e-Submission und e-Info das digitale Portal für das Einreichen von Entwurfsplänen. Im Folgenden ein kurzer Überblick über die drei Module:

e-Submission

e-Submission ist eine Webplattform, die es ermöglicht bauprojektrelevante Pläne und Dokumente entsprechenden Behörden für den Genehmigungsprozess zukommen zu lassen. Die Vorteile sind eine schnelle Bearbeitung, da es einen zentralen Sammelpunkt bildet, keine Papierdokumente mehr versendet werden müssen und rund um die Uhr verfügbar ist.

e-Info

e-info ist eine zentrale Quelle für Gebäude- und Konstruktionsrelevante Informationen in Singapur, das über das Internet zugänglich ist. Es beinhaltet unter anderem Information über singapurische Richtlinien, Standards bis hin zu Auftragnehmerperformance und nutzt zur Strukturierung der Daten Extensible Markup Language (XML) eine international standardisierte Auszeichnungssprache.

e-PlanCheck

Eine zentrale Rolle in diesem System spielt die Bibliothek namens FORNAX. Diese wurde von der Firma novaCITYNETS entwickelt und beinhaltet fest implementierte Programme, in der Programmiersprache C++, die zur Datenaufbereitung aus dem IFC-Datenschema und der Konformitätsprüfung dienen. Neben den bereits vorhandenen Regeln können mittels einer API neue entwickelt und bestehende verändert werden (KHEMLANI, 2005). CORENET und FORNAX sind bis heute eine der fortschrittlichsten Ansätze des ACCC und haben deutlich gezeigt, welche Vorteile dies mitbringt. Die durchschnittliche Zeit des Prüfverfahrens konnte in Singapur von 103 Tagen in 2007 auf 26 Tage in 2014 gesenkt werden. Ein Nachteil ist jedoch, dass die Erstellung von individuellen Regeln sehr viel aufwändiger ist als mit dem Solibri Model Checker (SMC) und dass dies auch nur mit Zugriffsrechten möglich ist (C. F. D. PREIDEL, 2020).

3.2.2 Visual Code Checking Language

Eine wichtige Ergänzung in den Reihen der ACCC-Methoden ist die Visual Code Checking Language (VCCL), eine Sprache zur Konformitätsprüfung, die mittels den Prinzipien einer Visual Programming Language (VPL) durch C. Preidel (C. F. D. PREIDEL, 2020) entworfen wurde. Entgegen den fest implementierten Tests kann hier der Anwender mittels einer Art Bauklotzsystem eigene Regeln entwerfen. Hierzu benötigt er keine Programmierkenntnisse, sondern kann die jeweiligen Methoden, auch Knoten genannt (siehe Abb. 3.4), miteinander verbinden und so eine Abfolge von Prozessen erzeugen, die eine entsprechende Aufgabe erfüllen. Bekannte VPL-Anwendungen sind Grasshopper3D von Rhinoceros3D, Dynamo, ein Plug-in für Revit, Marionette von Vectorworks und Googly Blockly. Die meisten VPL-Anwendungen in der AEC-Industrie konzentrieren sich jedoch hauptsächlich auf die Unterstützung bei intuitiven Designaufgaben, weshalb C. Preidel in seiner Arbeit die Prinzipien des VCCL für die Entwicklung von ACCC nutzte (C. F. D. PREIDEL, 2020).

VCCL ist eine stark typisierte, objektorientierte Sprache, die auf der semantischen Ebene die zwei Elemente Methoden und Schnittstellen besitzt. Die Methoden werden durch abgerundete Rechtecke dargestellt, die auf der Nutzeroberfläche je nach Bedarf verschoben werden können. Die Eingangsgrößen werden Operanden genannt und wie die Schnittstellen, die als Ports bezeichnet werden, sind sie Datenobjekte eines bestimmten Datentyps, die durch farbliche Kreise differenziert werden. Nur Datenobjekte des gleichen Typs können über die gerichteten Kanten miteinander verbunden werden. Die visuellen Programme werden von links nach rechts gelesen. Abbildung 3.4 zeigt ein simples Beispiel für eine arithmetische und eine vergleichende Operation.

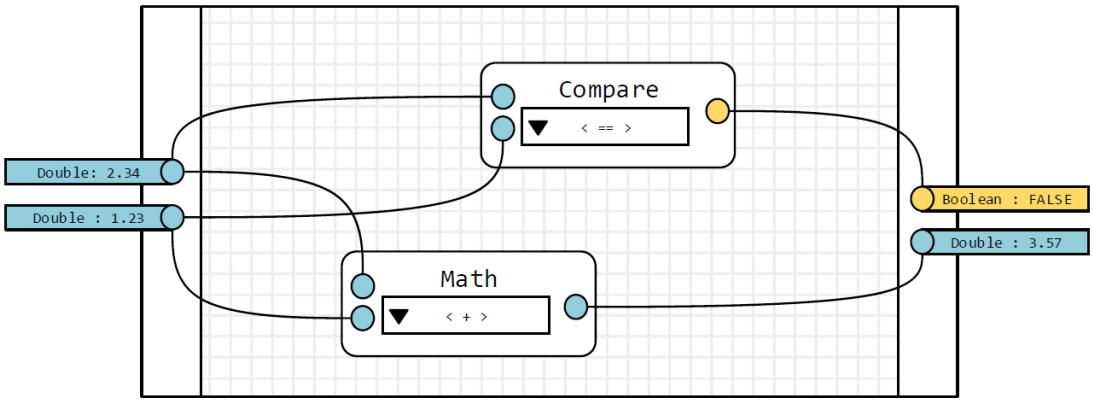


Abbildung 3.4: Beispielhaftes visuelles Programm (C. F. D. PREIDEL, 2020)

Im Kontext von ACCC wird zusätzlich noch der Objektknoten eingeführt. Dieser repräsentiert ein in der realen Welt existierendes Objekt, wie etwa eine Wand oder ein Straßenschild. Wie im Beispiel zuvor könnte nun damit eine Operation aufgestellt werden, die die Dicke der Wand abruft, sie mit einem gegebenen Wert vergleicht und somit die Konformität mit einer Norm abgleicht. Neben einer solchen mathematischen Funktion kann die Methode

aber auch aus einem Zugriff auf eine bestimmte Gruppe an Objekten bestehen. Die Abbildung 3.5 zeigt hierzu eine Auswahl (Set) an Wänden, die aus einem Modell die Wände filtert, die den gegebenen Kriterien entsprechen.

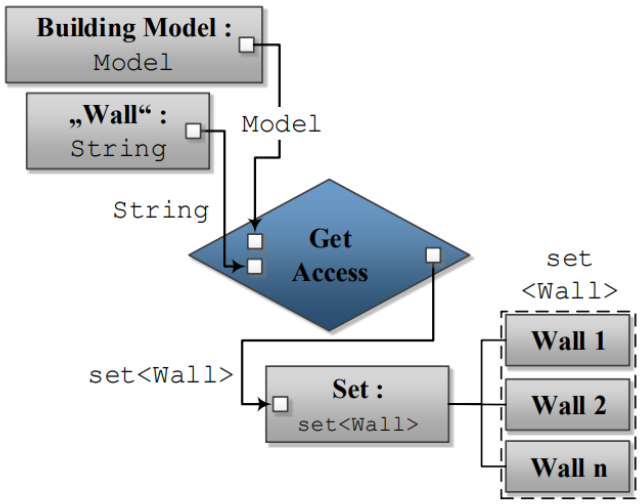


Abbildung 3.5: Zugriff auf bestimmte Wandobjekte eines Modells (C. PREIDEL & BORRMANN, 2015)

Die VCCL-Bibliothek enthält grundlegende Methoden, mit denen komplexere Programme entworfen werden können. Diese sogenannten atomaren Methoden können vom Nutzer nicht eingesehen werden und bilden somit Black-Boxes (siehe Abb. 3.2). Es wird jedoch davon ausgegangen, dass ein gewisser Mangel an Einsehbarkeit der Verständlichkeit zuträglich ist, weshalb VCCL als Whiteboxprozess klassifiziert wird, da der Ablauf im Gesamtkontext sehr transparent und gut nachvollziehbar ist.

Zusammenfassend kann gesagt werden, dass VCCL sehr anfängerfreundlich ist, da es keiner Programmierkenntnisse bedarf und eine große Variation an Möglichkeiten bietet. Jedoch ist der mögliche Grad der Komplexität im Vergleich zu hartkodierten Regeln geringer, da Schleifen und rekursive Funktionen nicht ohne weiteres implementierbar sind. VCCL zeigt einen vielversprechenden Ansatz für die Kombination von visuellen Bausteinen mit textuellen Programmen und könnte in der Zukunft eine effektive Unterstützung im Compliance-Prozess sein, der gleichzeitig, das Verständnis von Programmentwicklung für AEC-Fachkräfte ohne Programmierkenntnisse, um ein vielfaches erleichtert.

3.2.3 Semantic NLP-based Automated Compliance Checking (SNACC)

Normen und Richtlinien in maschineninterpretierbare Regeln zu überführen bedarf eines erheblichen Aufwands, unabhängig von der Art und Weise, wie der jeweilige Model Checker dies ermöglicht. Der SNACC-Prototyp soll diesen Prozess daher automatisieren. Es ist ein vereinheitlichtes ACCC-System, dessen Konzept auf der Integration von Natural Language Processing (NLP) -Algorithmen, semantischen BIM-Daten-Prozessalgorithmen und einer automatisierten Konformitätsargumentation basiert (ZHANG & EL-GOHARY, 2017). Die

Motivation hierfür ist die Einsparung von Zeit und Kosten, aber auch die Fehleranfälligkeit im Compliance Checking Prozess, da es eine höhere Flexibilität für Änderungen bietet als hartkodierte oder visuell programmierte Checks.

Das System besteht aus drei Hauptmodulen. Das Regulatory Information Extraction and Transformation Module erkennt Muster basierend auf einem Satz an Extraktionsregeln, generiert Informationstupel und transformiert diese in logische Regelsätze. Das Design Information Extraction and Transformation Module analysiert das Building information Model und kreiert auf logikbasierte Informationsrepräsentationen. Zuletzt vergleicht das Compliance Reasoning Module die Regelsätze und die Informationen des Modells und erstellt einen Compliance-Report. Bild 3.6 fasst die wichtigsten Prozesse zusammen. SNACC ist sehr nutzerfreundlich und bedarf keinerlei Programmierkenntnisse.

Obwohl die Tests, die durch die Entwickler durchgeführt wurden, ein vielversprechendes Ergebnis mit einer Präzision von 87,6% vorwiesen (Anteil der erkannten nonkonformen Instanzen), muss sich erst noch zeigen, wie das System unter realen Umständen performt, bedenkt man menschliches Versagen wie Rechtschreib- und Grammatikfehler. Weiterhin ist unklar, wie mit projektspezifischen Begriffen im jeweiligen IFC-Schema umgegangen werden muss. Noch ist das System also eher ein Forschungsansatz, der noch viel Entwicklung bedarf, jedoch im Angesicht der aktuellen Entwicklung der künstlichen Intelligenz von Spracherkennungssystemen immer greifbarer wird.

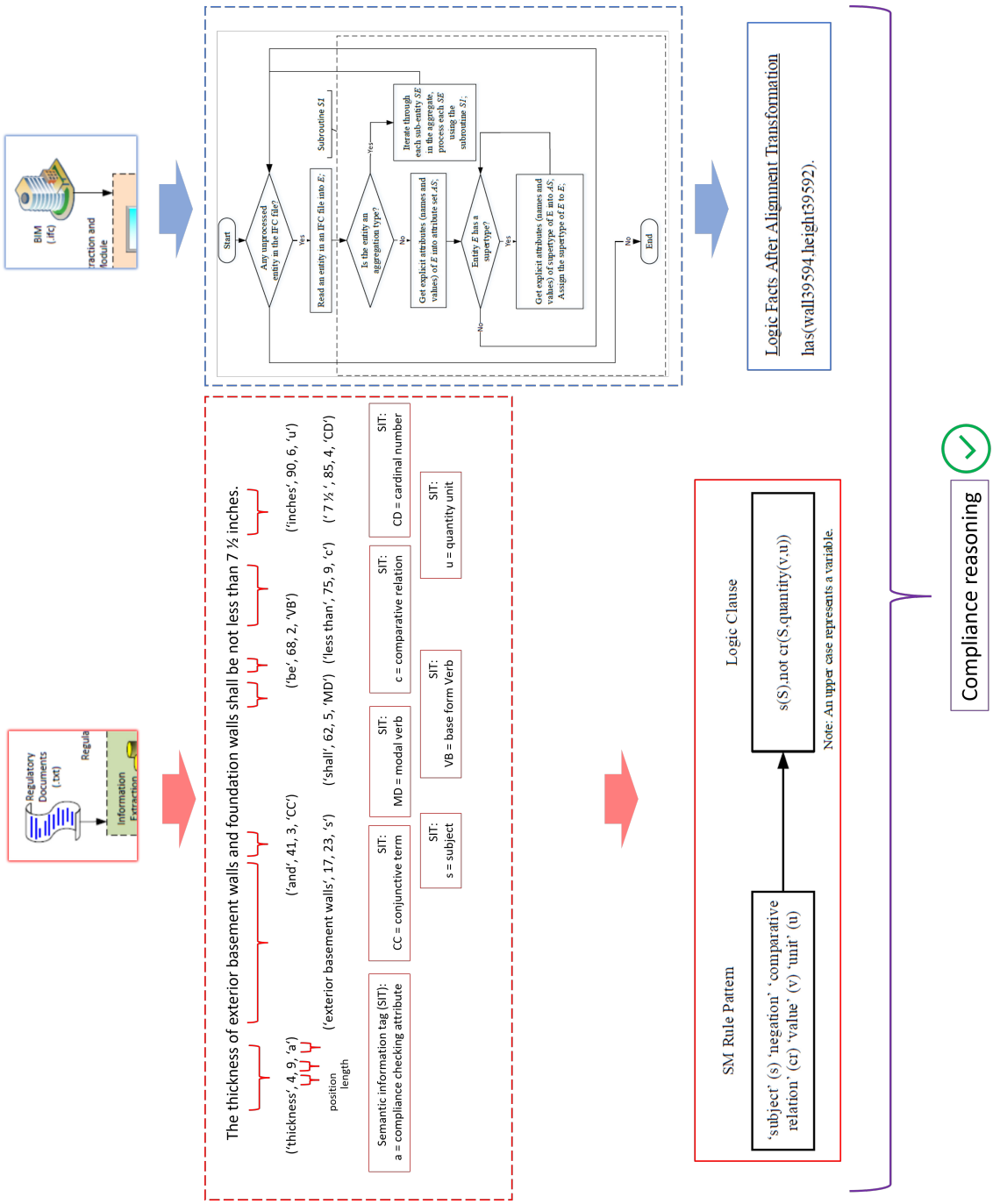


Abbildung 3.6: Die Kernprozesse von Snacc in einem Bild (ZHANG & EL-GOHARY, 2017)

3.3 Baysis

Das Bayerisches Straßeninformationssystem (BAYSIS) dient als zentrale Informationsplattform für die bayerische Straßenbauverwaltung. Es beinhaltet zwar kein Prüfsystem für ACCC wurde aber wie CORENET aus staatlicher Hand entwickelt und soll unterschiedliche Bereiche miteinander vernetzen und Prozesse effizienter gestalten. Die Federführung für BAYSIS liegt beim Staatsministerium für Wohnen, Bau und Verkehr und die Datenspeicherung und Zurverfügungstellung der Programme sowie die Betreuung und Weiterentwicklung ist Verantwortung der Zentralstelle Straßeninformationssysteme (ZIS),

angesiedelt bei der Landesbaudirektion Bayern. Abbildung 3.7 gibt einen Überblick über die verschiedenen Schnittstellen des Systems.

Abbildung 3.7: Schnittstellen zu BAYSIS

Die Zentralstelle Verkehrssicherheit (ZVS) trägt die Verantwortung über den gleichnamigen Bereich und verwaltet Unfalldaten, Publikationen und Auditverfahren. Hier sind die Sicherheitsauditoren in der Lage ihre Arbeit in Form eines Berichtes, der die Defizite beschreibt und die Vollständigkeit der Dokumente bewertet, digital zu erstellen und einzureichen. Hierbei müssen die Defizite über ein Drop-down-Menü immer eindeutig benannt werden und können erst dann zusammen mit der Erläuterung eingereicht werden. Danach erhält der verantwortliche Planer über BAYSIS eine Benachrichtigung, die ihn zu einer Stellungnahme und einer Entscheidung auffordert, die festlegt, ob die Planung entsprechend dem Defizit angepasst oder beibehalten wird. Das System erstellt aus beiden Beiträgen einen Auditbericht, der die Ergebnisse zusammenfasst und den Bericht und die Stellungnahme übersichtlich gruppiert. Hierdurch wird die Kommunikation zwischen Auditoren und Planern beschleunigt, die Prozesse standardisiert und Papier eingespart.

3.4 Modellqualität

Da der BIM-Planungsprozess das digitale 3D-Model im Zentrum hat, ist eine hohe Modell- und Datenqualität unerlässlich. Dies gilt noch mehr für die Anwendung von ACCC, denn bevor ein Programm Daten automatisiert interpretieren und verwerten kann, müssen diese korrekt vorhanden sein. Da es noch keinen allgemeinen Standard für die Modellqualität gibt, muss dieser projektspezifisch definiert werden. Durch eine Einteilung in drei Ebenen wird eine klare Struktur erzeugt. C. Preidel definiert diese als datentechnische, inhaltliche und planerische Qualität (C. F. D. PREIDEL, 2020). Im Folgenden wird auf diese eingegangen.

Bei der datentechnischen Qualität handelt es sich um die syntaktische Ebene des Modells. Entspricht es den Anforderungen der jeweiligen Syntaxregeln, wird von einer wohlgeformten Datenqualität gesprochen. Ein treffendes Beispiel hierfür ist das IFC-Schema, das in jeder Datei in der ersten und letzten Zeile auf die ISO-10303-21 verweist. Hiermit bezieht es sich auf die STEP Part 21 und darauf, dass es dessen Syntax als Basis nutzt. Fehler in dieser Ebene des Modells sind meist softwaretechnischer Natur, da bis auf Softwareentwickler die Projektbeteiligten im AEC-Sektor eher selten die Dateien manuell beschriften. Konfliktursachen können fehlerhaft ausgefüllte Parameterlisten sein, die zum Beispiel zu viele oder die falschen Attribute enthalten. Ein solches Problem kann unter anderem vorkommen, wenn eine Anwendung für das IFC-Schema 2x3 ausgelegt ist, jedoch das Modell auf IFC 4 basiert. Neben der Wohlgeformtheit werden durch C. Preidel noch die Begriffe Vollständigkeit, Aktualität und Verfügbarkeit definiert (C. F. D. PREIDEL, 2020). Ersteres bezieht sich auf die Integrität des Datensatzes und kann durch abgebrochene Exportvorgänge kompromittiert werden. Der zweite Punkt soll sicherstellen, dass die Pläne alle auf dem neuesten Stand sind und Letzteres bezieht sich auf die Pünktlichkeit der Modelleinreichung der jeweiligen Fachplaner zu den gegebenen Data-Drops.

Die inhaltliche Qualität ist das, was in BIM-Projekten häufig als die Modellierungsqualität verstanden wird. Diese besteht aus zwei Kriterien, der Semantik und der Geometrie. Ersteres lässt sich gut mit dem Infofenster assoziieren, das bei manchen Model Viewer erscheint, wenn ein Objekt angeklickt wird. Hier erscheinen alle Daten zur entsprechenden Komponente. Für jedes Projekt ist vorgegeben, welche Information in den entsprechenden Leistungsphasen enthalten sein muss. Der Detaillierungsgrad kann hierbei zwischen den Teilmodellen und Objekten variieren. So kann bei einem Tunnelbauprojekt in einer Leistungsphase die Betongüte wichtiger sein als der Hersteller der Notfallausgangstüren. Diese Daten werden über Attributslisten an die Objekte angehängt. Zusätzlich können Klassifikationen helfen, mit denen das Modell beispielsweise in unterschiedliche Projektphasen eingeteilt werden kann. Die Semantik ist einer der wertvollsten Aspekte in der Nutzung von BIM, in ihr steckt einer der Kernelemente der modellbasierten Planung. Hier wird deutlich, warum es so wichtig ist, diese Daten korrekt einzupflegen. Redundante oder widersprüchliche Informationen und fehlerhafte Klassifikationen können zu inkorrekter Interpretation führen, sowohl durch Menschen, als auch durch Programme.

Der visuelle Aspekt des Modells wird am meisten durch die geometrische Qualität geprägt. Viele Anwendungen beinhalten bereits eine integrierte Kollisionsprüfung, um sich überschneidende Teilmodelle und Materialverlust zu vermeiden. Weiterhin spielt die Positionierung eine wichtige Rolle und die Wahl des Koordinatensystems. Dieses muss von allen Planern für alle Teilmodelle konsistent übernommen werden, um unter anderem Skalierungsfehler zu vermeiden. Weiterhin muss auf einen adäquaten Detaillierungsgrad geachtet werden, der bestimmt aus wie viel einzelnen Objekten etwas besteht. So bestimmt dies, ob ein Bauteil, das aus mehreren Schichten aufgebaut ist, auch so detailliert dargestellt wird. Ein weiterer Aspekt ist die semantisch-geometrische Ebene, die Schnittstelle, der beiden bereits erläuterten Bereiche. Hier wird darauf abgezielt, Diskrepanzen zwischen der geometrischen Darstellung und den Informationen in der Attribuierung zu vermeiden. Beispiele hierfür sind Volumen oder Positionen relativ zum Stockwerk, die direkt aus der Geometrie des Objekts abgeleitet werden können, aber unter Umständen unabhängig davon auch als semantische Informationen vorliegen.

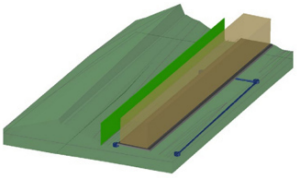
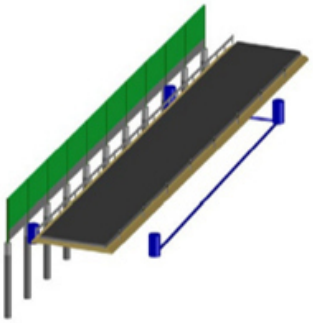
Die dritte Ebene der Modellqualität ist die planerische Qualität. Die Effektivität des ACCC-Prozesses hängt von den zwei zuvor erläuterten Ebenen ab, richtet sich aber primär auf diesen Bereich aus. Hier wird nämlich bestimmt, ob und wie weit ein Modell den Normen und Richtlinien entspricht. Weiterhin definiert diese Ebene die Einhaltung von internen Bürostandards und Auftraggeberanforderungen. Viele Büros haben ihren eigenen BIM-Standard eingeführt, der für gewisse Objekte und Räume einen eigens definierten Katalog enthält. Das Projektmanagement vieler Firmen kann dadurch effektiver arbeiten und nutzt dementsprechend teilweise generell gültige Definitionen, aber auch projektspezifische Standards.

3.5 LOG, LOI und LOD

Das Building Information Model besitzt im Gegensatz zur konventionellen Planung, die hauptsächlich auf zweidimensionalen Bauteilzeichnungen basiert, keine Maßstäbe. Diese definieren entsprechend dem Planungsfortschritt im Wesentlichen den Ausarbeitungsgrad, weshalb dieser in BIM-Projekten explizit vorgegeben werden muss. Hierfür wurde das Konzept des Level of Developments (LOD) eingeführt, das aus dem Level of Information (LOI) und dem Level of Geometry (LOG) besteht. Der LOI bezieht sich auf den mit dem Projekt wachsenden Grad des semantischen Informationsgehalts. Somit ist für die jeweiligen Projektphasen festgelegt, ob und wie detailliert Daten wie Materialeigenschaft, Einbaudatum und ähnliches einzupflegen sind. Der LOG legt fest, wie detailreich die Modellierung in Bezug auf Geometrie, Bemaßung und Ausrichtung sein soll. Ein Objekt wie beispielsweise eine sanitäre Einrichtung kann demnach nur ein Quader oder eine komplett visualisierte Keramikschüssel mit Zu- und Abwasserleitung sein. Solche Details sind ausschlaggebend für die Präzision der Mengen- und Kostenermittlung. Über die Auftraggeber-Informations-Anforderungen (AIA) werden die projektspezifischen Anforderungen der LOI und LOG festgelegt und auf angehängte Dokumente verwiesen, die die Planer zur Orientierung nutzen können.

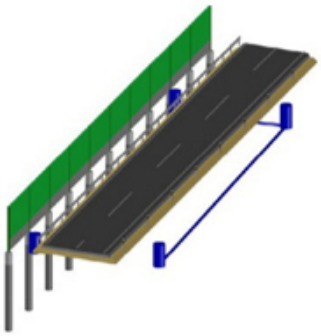
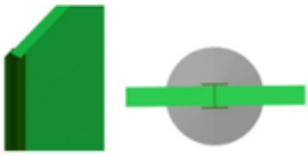
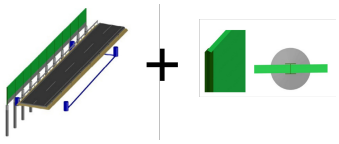
Die erste LOD Version wurde durch die amerikanische Dachorganisation BIMForum 2011 zusammen mit dem American Institute of Architects herausgegeben und wird seither als Orientierungsmaßstab für viele andere Standards gesehen (BORRMANN et al., 2020). Unter anderem durch den 2015 veröffentlichten Stufenplan des Bundesverkehrsministeriums, der die Nutzung von BIM für alle staatlichen Infrastrukturprojekte bis 2020 verpflichtend machen sollte (BMVI, 2015), begannen verschiedene Institutionen eigene LOD-Spezifikation zu entwickeln. Einige strebten hierbei die Verknüpfung der LOD mit der Honorarordnung für Architekten und Ingenieure (HOAI) an, was im LOD Konzept nicht so vorgesehen war. Dieses sollte lediglich als Grundlage für den Fertigungsgrad einzelner Modellbestandteile dienen, unabhängig von der Leistungsphase des Gesamtprojekts (BORRMANN et al., 2020). Ein gutes Beispiel hierfür ist BIM4INFRA2020, ein vom Bundesverkehrsministerium beauftragtes Konsortium, das in ihrer Handreichung eine strukturierte Visualisierung der LOD und HOAI Leistungsphasen darstellt. Da die Arbeit des Sicherheitsauditors direkt abhängig von dem Detaillierungsgrad der Planung ist, sind die Auditphasen ebenfalls aufgeführt (siehe Tabelle 3.1). Es sei noch angemerkt, dass der Level of Development nicht mit Level of Detail verwechselt werden darf. Letzteres bezieht sich primär auf den visuellen Ausarbeitungsgrad, gibt aber keinerlei Auskunft über den semantischen Informationsgehalt und eignet sich somit nicht für die "[...] Beschreibung der Informationsreife von Modellen in Bezug auf fortschreitende Planungsprozesse[...]"(BORRMANN et al., 2020, S. 168).

Tabelle 3.1: LOD, Lphs und Auditphasen mit Beispielen (BMVI, 2019a)

LOD und Lphs nach HOAI	Beschreibung	Modellbeispiele
LOD 100 Vorplanung	Lph 2 und Auditphase 1, grobe Bauwerksparameter, auch Vorentwurfsmodelle genannt	
LOD 200 Entwurfsplanung und Genehmigungsplanung	Lph 3 und 4 und Auditphase 2, Wesentliche Modellelemente mit korrekter Geometrie und - alphanumerischen Daten, Auch Entwurfsmodelle genannt	

Cont'd on following page

Table 3.1, cont'd

LOD und Lphs nach HOAI	Beschreibung	Modellbeispiele
LOD 300 Ausführungsplanung, Vorbereitung der Vergabe und Mitwirkung bei der Vergabe	Lph 5 bis 7 und Auditphase 3 und 4 (Ausführungsplanung und Vor Verkehrsfreigabe), Ausführungsreif modellierte Bauteile, größere alphanumerische Informationstiefe	
LOD 350	Ergänzende Detailzeichnungen	
LOD 400 Objektüberwachung	Lph 8 Zusätzlich Montage- und Installiationsdetails modelliert, noch größere alpahnumerische Informationstiefe, auch Bau- und Montagemodelle genannt	
LOD 500 Objektbetreuung und Dokumentation	Lph 9 und Auditphase 5 (Nach Verkehrsfreigabe), vollständige visuelle und alphanumerische Information, auch "Wie-gebaut Modell genannt	siehe LOD 400

3.6 BCF und CDE

Das BIM Collaboration Format ist eines der Standards im buildSMART Toolkit und spielt in der Zusammenarbeit der Projektbeteiligten eine wichtige Rolle. Da die Planung ein iterativer Prozess ist müssen die unterschiedlichen Fachdisziplinen ständig kommunizieren. Ein effizienter Ablauf dieses Austauschs ist hierbei klar von Vorteil. Screenshots und Protokolle sind zwar eine Möglichkeit des Austauschs, kosten aber sehr viel Zeit und sind fehleranfällig. Eine direktere Lösung ist daher um ein vielfaches praktischer. Sei es ein Hinweis oder ein Problem mit dem Modell, mittels BIM Collaboration Format (BCF) ist es möglich direkt im Modell einen Konflikt zu erstellen, der über einen Server für alle Beteiligten zugänglich ist. Selbst die meisten nativen Anwendungen können auf diese

Daten zugreifen, wodurch die Konflikte von allen überall eingesehen und kommentiert werden können.

Um für einen zentralen Ablageort zu sorgen, wird eine CDE genutzt. Diese dienen einerseits als Zugang zu den Modellen und andererseits als Protokoll für den Projektablauf und die Aufgabenverteilung. Hier werden die Koordinationsmodelle abgelegt, die für die regelmäßigen Projektbesprechungen genutzt werden und sorgen somit für mehr Transparenz. Über die CDE kann ebenfalls BCF genutzt, wodurch alle erstellten Konflikte eingesehen und neue erstellt werden können.

3.7 AR und VR im AEC-Sektor

Wird von AR und VR gesprochen, müssen diese klar unterschieden werden. AR wird als die Echtzeiterweiterung der Realität mittels virtueller 3D-Elemente verstanden und VR als ein Eintauchen in eine vollständig virtuell erzeugte Welt (SCHRANZ et al., 2020). Ersteres ist dabei bisher weiter verbreitet, vor allem in der Bauausführung. Mittels einer AR-Anwendung können Baustellen mit den Modellen und den darin enthaltenen geometrischen und semantischen Informationen überlagert und so Arbeitsschritte besser koordiniert, kontrolliert und dokumentiert werden. Genutzt werden die Funktionen über Apps auf mobilen Geräten und Head Mounted Display (HMD)-Geräten (siehe Abb. 3.8). Aktuell wird AR vermehrt durch Schalungstechniker, in der TGA-Planung und in der Baudokumentation bei der Abnahme und dem Qualitätsmanagement genutzt. Beispiele der Anwendungen sind unter anderem Gamma AR oder S-Construct. Durch die Überlagerung der Baustelle mit dem Teilmodell kann die Planung besser in einen realen Kontext eingeordnet werden, was für Behörden von Vorteil sein kann, die so Bürger, Baupolizei und andere Beteiligte besser in den Prozess involvieren können. Weiterhin gibt es noch viele vielversprechende Ansätze, die sich in der Entwicklung befinden. So könnte der Baugrubenaushub in der Zukunft durch die Verwendung von AR unterstützt werden, was die Notwendigkeit von Messpunkten reduzieren und durch das Anzeigen von Bestandsleitungen die Sicherheit steigern würde. Das Remote-Expert-System ist ein weiteres Programm, das sich in Entwicklung befindet. Statt ein Problem mit Fotos, Skizzen und E-Mail zu übermitteln, kann es direkt mittels Videotelefonie in Kombination mit einem Tracking-System kommuniziert und im Echtzeitbild markiert werden.

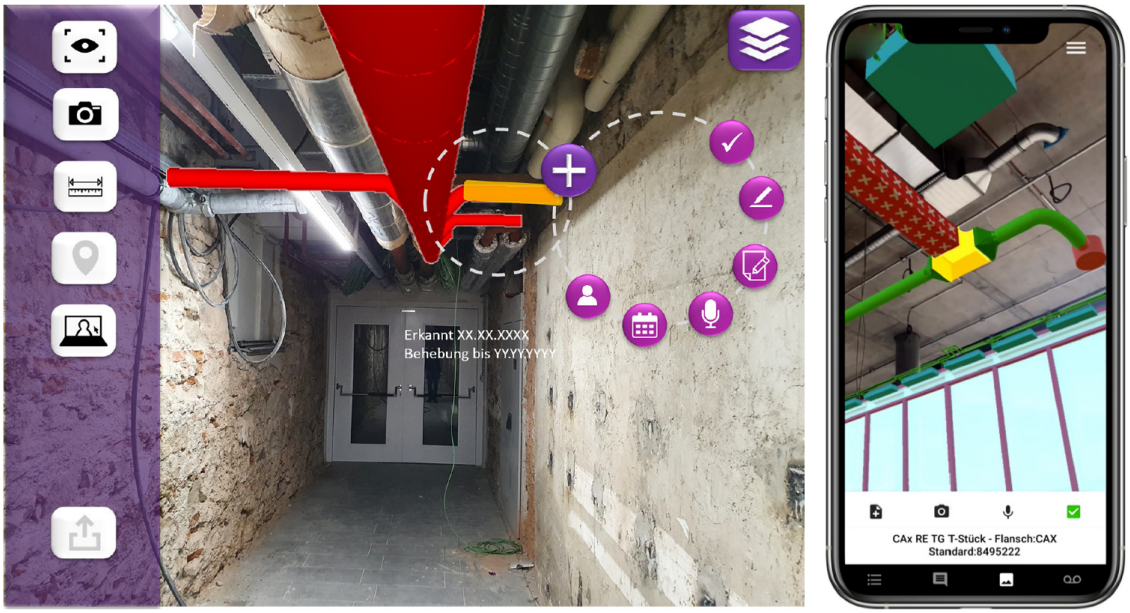


Abbildung 3.8: Head Mounted Display und Smartphone App von Gamma AR (SCHRANZ et al., 2020)

VR findet derzeit in der Planungsphase, in Schulungen und in der Immobilienbranche seine größte Anwendung. So kann es genutzt werden, um einen realen Eindruck des Modells zu gewinnen, Dimensionen besser einschätzen zu können und Blickwinkel nachzuvollziehen. Weiterhin ist es sehr nützlich in der Kollaboration. Projektbeteiligte können sich auf der virtuellen Baustelle treffen und die Planung besprechen. Unterschiedliche Anwendungen dienen dabei entsprechenden Zwecken. Enscape und ALLVR sind hierbei sehr gut für visuell ansprechende Darstellungen, da diese über eine Echtzeitrenderingfunktion verfügen und so durch reale Lichteinflüsse und Farben dem Projekt "Lebenëinflößen. Dies ist für Architekten und Immobilienmakler von Vorteil, da sie auf diese Art den Kunden, die zum Teil keine Fachkenntnisse besitzen, ihre Produkte angemessen vermarkten können. VRex ist visuell etwas simpler konzipiert und mehr auf einen effizienten Workflow und ein möglichst leicht anwendbares Konfliktmanagement konzentriert. Es bietet jedoch als die derzeit einzige VR-Plattform (Stand Juni 2022) eine Kollaboration mit BCF an, wodurch erstellte Konflikte direkt im Modell eingesehen und in einem Meeting gemeinsam besprochen werden können. Weiterhin findet VR in der Sicherheitsschulung Anwendung. Gefährliche Situationen können hierdurch auf eine reale Art und Weise simuliert und dadurch den Beteiligten die Risiken buchstäblich besser vor Augen geführt werden.

4. Verkehrssicherheit und BIM an der Landesbaudirektion Bayern

4.1 Geschichte der Verkehrssicherheit

Die Entwicklung der Verkehrssicherheit war ein stetiger Prozess, der sich als Konsequenz aus vielen tragischen Schicksalen, aber auch aus wirtschaftlichen Faktoren bildete. Er formte unser heutiges Straßenbild, die Fahrzeugausstattung, unsere verkehrspädagogischen Methoden und Gesetze. Durch die Verbreitung von motorisierten Kraftfahrzeugen wurden immer mehr Maßnahmen zur Sicherung des Verkehrs erhoben. Zuvor waren Straßen eine frei nutzbare Fläche, die von Passanten und Kutschen gleichermaßen genutzt wurden. Zwar gab es Verkehrsunfälle seit jeher, doch erst durch das deutliche Überschreiten der Geschwindigkeiten von Fußgängern und Pferden, erstmals durch die dampfbetriebene Lokomotive erreicht, wurde die Häufigkeit und Schwere zu einem wachsenden Problem. Im Folgenden wird auf die historisch relevantesten Punkte chronologisch eingegangen.

19. Jahrhundert

1869 explodiert in London die erste gasbetriebene Signalanlage kurz nach ihrer Installation (OSSENKOPP, 2018).

Ein Reichsgesetz schreibt 1877 deutschlandweit einheitliche Warntafeln an Bahnübergängen vor (FOCUS, 2020).

Der erste Führerschein wird 1888 an Carl Benz ausgestellt, dieser hatte jedoch nur die Funktion einer Fahrerlaubnis und bedurfte keiner bestandenen Fahreignungsprüfung (SPIEGEL, 2008).

1900-1910

Am 29. September 1903 wird in Preußen die erste Fahreignungsprüfung durchgeführt, welche angehende Kraftfahrzeugfahrer rudimentär prüft. Hier fragte der Dampfkesselüberwachungsverein, der Vorgänger des heutigen Technischen Überwachungsvereins (TÜV), die technischen Kenntnisse des Fahrers ab und testete die Fahrkünste in einer kurzen Übungsrunde im Hof (SPIEGEL, 2008).

1906 gibt es bereits rund 10.000 Autos und 16.000 Motorräder in Deutschland (STATISTA, 2000).

Im selben Jahr veröffentlicht das Amtsblatt der preußischen Provinz zu Cassel die ersten Verordnungen. Hierin werden unter anderem Vorgaben festgelegt, die die Anmeldung und Kennzeichnung des Fahrzeuges bestimmen, wo Kraftfahrzeuge fahren dürfen und

wo sie eine polizeiliche Erlaubnis benötigen. Ebenso werden Sonderregelungen für das Wettfahren und Mitführen von Anhängern aufgestellt (AMTSBLATT, 1906).

1909 werden durch das kaiserliche Staatsministerium des Innern die Geschwindigkeitsbeschränkungen für PKW, die innerhalb von Ortschaften auf maximal 15 km/h begrenzt wurden, aufgehoben und nur für Kraftfahrzeuge über 5.5 t beibehalten (AMTSBLATT, 1910).

1910-1920

1910 wird der offizielle Führerschein deutschlandweit eingeführt und existiert in dieser Form bis heute (SPIEGEL, 2008).

Im selben Jahr werden die ersten international anerkannten Verkehrszeichen im deutschen Reichsgesetzblatt veröffentlicht, auf die sich die Länder Bulgarien, Deutschland, Großbritannien, Monaco, Italien, Österreich-Ungarn und Spanien ein Jahr zuvor einigen (FOCUS, 2020).

In Cleveland wird 1914 die weltweit erste erfolgreiche elektrische Verkehrsampel aufgestellt (OSSENKOPP, 2018).

1919 werden in Deutschland vereinzelt die ersten weißen Straßenmarkierungen appliziert (DSGS, 2022).

1920-1930

1924 wird in Berlin die erste deutsche Ampelanlage aufgestellt (OSSENKOPP, 2018).

Im selben Jahr wird ebenso in Berlin die Deutsche Verkehrswacht (DVW) mit dem Ziel gegründet, die Verkehrssicherheit zu verbessern und sich für die Minderung der Verkehrsunfälle einzusetzen (DVW, 2022).

1930-1940

1930 gibt es bereits rund 297.000 PKW in Deutschland (STATISTA, 2000).

1934 werden in der Reichs-Straßenverkehrs-Ordnung rechteckige Prellsteine als Fahrbahnbegrenzungen aufgenommen, die Vorläufer der heutigen Fahrzeugrückhaltesysteme (REICHSGESETZBLATT, 1934). Zusätzlich wurden die Bankette für eine sichere Spurführung mit heller Schaumlava befestigt.

1935 werden Straßenmarkierungen zum festen Bestandteil auf Autobahnen (DSGS, 2022).

1934 wird Kraftfahrzeugen eine Vorrangstellung im Verkehr eingeräumt und alle Geschwindigkeitsbeschränkungen aufgehoben (REICHSGESETZBLATT, 1934).

1937 wird gesetzlich festgelegt, dass Fußgänger den Gehweg benutzen müssen und die Straße nur überqueren, sich also nicht mehr darauf aufhalten dürfen (REICHSGESETZBLATT, 1937).

1940-1960

1940 wird erstmals in der Straßenverkehrsordnung (StVO) bestimmt, dass Fahrräder mit roten Schlusslichtern und an den Pedalen mit gelben Rückstrahlern ausgestattet sein müssen (§25 Absatz 1-4 des StVO i.d.F.v. 24. April 1940).

Durch den Krieg entschleunigt sich das Fahrzeugwachstum und es gibt 1950 die relativ geringe Anzahl von 518.000 Autos (ZAHLENBILDER, 2020).

1951 wird die regelmäßige Hauptuntersuchung durch den TÜV eingeführt (SPIEGEL, 2001).

Im selben Jahr tauchen die ersten Zebrastreifen auf, waren aber bis 1964 nur Indikatoren für Fußgängerübergänge ohne den Vorrang für Fußgänger (SCHUBERT, 2022)

1959 patentiert der schwedische Ingenieur Nils Ivar Bohlin (BOHLIN, 1959) seinen Dreipunktgürtel, der umgehend in Schweden serienmäßig für alle Fahrzeuge eingeführt wird (POSMIK, 2010).

1960 - 1980

1960 gibt es bereits 4,8 Mio. Fahrzeuge auf deutschen Straßen, mit einem Wachstumstrend, der bis heute anhält (Stand 2020: 47,7 Mio.) (ZAHLENBILDER, 2020).

1969 wird der Deutsche Verkehrssicherheitsrat (DVR) gegründet, der motiviert von der stetig wachsenden Anzahl an Verkehrstoten, Politik und Industrie mittels den neuesten wissenschaftlichen Erkenntnissen berät und Trainings und Kampagnen entwickelt (DVR, 2022).

Deutschland folgt dem schwedischen Beispiel und führt 1974 serienmäßig den Dreipunktgurt ein und zwei Jahre später die Anschnallpflicht (POSMIK, 2010).

Ab 1980

1980 wird der erste verkehrsberuhigte Bereich eingeführt (§42 Absatz 4a des StVO i.d.F.v. 21. Juli 1980).

1985 wird der Sicherheitsgurt durch das Deutsche Patentamt als eine der acht Erfindungen gewählt, die dem Menschen in den letzten 100 Jahren am meisten brachten (MÖSER, 2002).

Seit 2007 verfolgt der DVR die Vision Zero, die als Ziel hat keinerlei Verkehrstote mehr beklagen zu müssen (DVR, 2022).

Beginn des Sicherheitsaudits

Die Geschichte hat gezeigt, wie sich die Verkehrsinfrastruktur von einem unregulierten zu einem ausgeklügelten System entwickelt hat. Hierbei wurden die Belange der Verkehrssicherheit in einer wachsenden Anzahl technischer Regelwerke über Planung, Bau und Betrieb von Straßen hinreichend integriert. Nichtsdestotrotz gibt es einzelne neu gebaute Straßen, die auffällig unfallbelastet sind. Um dieses Problem zu lösen, hat sich

das Verkehrssicherheitsaudit entwickelt. Seine Geschichte ist demnach relativ jung und beginnt 1988 mit der Einführung des Road Traffic Acts im Vereinigten Königreich (TULLO, 1988). Deutschland begann erst 2002 mittels der Empfehlungen für das Sicherheitsaudit von Straßen (ESAS) (FGSV, 2002) Empfehlungen aufzustellen, die der Verbesserung der Verkehrssicherheit dienen sollten und den Grundstein für die Zertifizierung des Sicherheitsauditors legten. 2008 entwickelte das Europäische Parlament mittels der Richtlinie 2008/96/EG ein Sicherheitsmanagement für die Straßeninfrastruktur (EU-PARLAMENT, 2008), die das transeuropäische Straßennetz verbessern sollten und erst 2019 wurde aus den Empfehlungen in der ESAS bindende Richtlinien in Form der Richtlinien für das Sicherheitsaudit von Straßen (RSAS) entwickelt (RSAS, 2019), die das Hauptleitwerk für das heutige Sicherheitsaudit bilden.

4.2 Verkehrssicherheitstheorie

"Die wichtigste Grundlage der Notfallversorgung von Patienten ist die Notfallvermeidung."(SCHNIEDER & SCHNIEDER, 2013, Steiger, 2012, S.5). Mit diesen Worten leitet der Gründungspräsident der Björn Steiger Stiftung Dr. med. h.c. Siegfried Steiger das Thema Verkehrssicherheit ein. Die Notfallversorgung ist nämlich in den letzten Jahren massiv verbessert worden, doch ist die Zahl der Verkehrsunfälle zum Zeitpunkt des Zitats nicht rückläufig. Teil der Lösung sind die pädagogischen Maßnahmen, die sich im Fahrschulunterricht und in der schulischen Verkehrserziehung äußern. Diese haben jedoch ihre Grenzen, weshalb umfassende wissenschaftliche Untersuchungen unternommen worden sind, um die eher abstrakte, nicht unmittelbar messbare Größe der Verkehrssicherheit zu quantifizieren und berechenbar zu machen und so die Sicherheit prognostizieren, bewerten und verbessern zu können. Das Sicherheitsaudit, um das es letztendlich in dieser Arbeit geht, ist die letzte Instanz, die prüft, ob die aus der Verkehrssicherheitstheorie gewonnenen Erkenntnisse bezüglich der Gestaltung der Verkehrswegeinfrastruktur in der Planung beziehungsweise im Bestand eingehalten werden. Das Feld der Verkehrssicherheitstheorie ist sehr umfangreich und erstreckt sich über konzeptionelle Grundlagen wie Formulierungen über Darstellungsweisen bezüglich Wahrnehmung, Statistik und Risikometrie über Modellierungsmethoden für psychische Prozesse und Systemdynamiken bis hin zu der Realisierung der Maßnahmen. Erschwerend kommt hinzu, dass die diversen Einzelaspekte dieser Bereiche aufgrund der noch bestehenden terminologischen Unschärfe nur in sehr komplexer Weise miteinander kombiniert werden können (SCHNIEDER & SCHNIEDER, 2013). Das Verkehrssystem interagiert wie in 4.1 dargestellt mit den unterschiedlichen Aspekten seines Umfelds, daher kann Verkehrssicherheit aus vielen verschiedenen Gesichtspunkten betrachtet werden. Drei wesentliche Betrachtungsgegenstände sind die Wirtschaft, die Psychologie und die Technik. In diesem Kapitel wird primär auf den zweiten Punkt eingegangen.

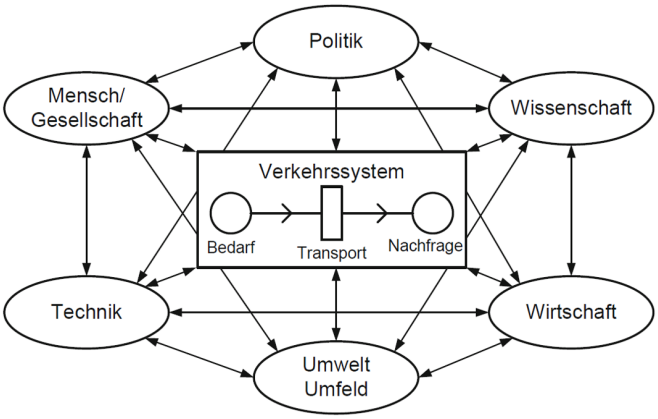


Abbildung 4.1: Aspekte und Umfeld des Verkehrs (SCHNIEDER & SCHNIEDER, 2013)

4.2.1 Verkehrspsychologie - Wahrnehmung

Verkehrssicherheit ist ein mentales Konstrukt und kann daher nicht als solches mittels den Sinnen wahrgenommen werden, sie ist nur aus beobachtbaren oder messbaren Sachverhalten ermittelbar. Sicherheit wird demnach empfunden, wenn das Risiko als gering eingeschätzt wird, weshalb auch von der Risikowahrnehmung gesprochen wird. Sicherheitsdefizite wie ünübersichtliches Radwegendeöder nicht ausreichende Anfahrsichtweiten fallen in den Bereich der Verkehrspsychologie und haben direkt mit der Wahrnehmung zu tun. Hierbei spielen drei der fünf Sinne eine Rolle: visuelle, akustische und haptische Wahrnehmung.

Visuelle Wahrnehmung

Der primäre Sinn im Straßenverkehr ist die visuelle Wahrnehmung. Diese wird in drei Ebenen aufgeteilt. Die physikalische, die physiologische und die psychologische Ebene. Die erste Ebene erstreckt sich von der Reizquelle des Lichts, also dem sichtbaren Objekt bis hin zum Sinnesorgan, also dem menschlichen Auge. Dieses kann bei ausreichender Beleuchtungsstärke mittels seinen Zapfen das reflektierte Licht wahrnehmen und so die Objekte erkennen. Mit abnehmender Helligkeit nimmt die Fähigkeit des Farbsehens ab. Die Gestaltung der Infrastruktur und Verkehrsmittel nutzt daher gezielt optische Reizquellen, die das menschliche Auge in seiner Wahrnehmung unterstützen. Die physiologische Ebene beschreibt den körperlichen Wahrnehmungsapparat von Netzhaut bis hin zu den neuronalen Prozessen in den visuellen Hirnarealen. Die letzte Ebene besteht aus dem psychologischen Aspekt, der sich mit der Entstehung des eigentlichen Wahrnehmungsbildes im menschlichen Gehirn befasst. Dieser wird durch den Konsum von Drogen, aber auch durch das Lebensalter beeinflusst.

Akustische Wahrnehmung

Die akustische Wahrnehmung ist im Verkehr ebenfalls sehr relevant. Sie dient einerseits der Orientierung und andererseits der Kommunikation. Die über die Luft übertragenen Schallwellen werden hierbei durch das Außenohr über das Mittelohr bis hin zum Innenohr

übertragen und gelangen über den Hörnerv zum Hirn. Diese Reize werden in den primären und sekundären Hirnarealen kognitiv verarbeitet und interpretiert.

Haptische Wahrnehmung

Die haptische Wahrnehmung spielt im Verkehr meistens durch Vibrationen eine Rolle. Straßenbeläge können so konstruiert sein, dass sie dem Fahrer durch Rütteln eine Warnung vermitteln. Ähnlich dazu gibt es Lenkräder, die durch leichten Gegendruck das Verlassen der Fahrspur signalisieren. Weiterhin können Sehbehinderte durch taktile Elemente im Verkehrsraum unterstützt werden, die ihnen helfen sich zu orientieren.

Risikowahrnehmung

Die Risikowahrnehmung wird in unterschiedliche Kategorien geteilt. Es gibt mehr als 300 verschiedene Klassen, die offensichtlich an dieser Stelle nicht alle diskutiert werden können, weshalb nur auf zwei sehr relevante Wahrnehmungspaare eingegangen wird.

Objektive vs. Subjektive Risikowahrnehmung

Die objektive Risikowahrnehmung ist als das Produkt aus Eintrittswahrscheinlichkeit und Schwere des Schadens pro Zeit definiert und findet daher meist in der Wirtschaft und Wissenschaft Anwendung. Hierbei werden induktive und deduktive Risikomodelle erstellt. Erstere schätzt mangels empirischer Daten eine Fehlerwahrscheinlichkeit der Einzelkomponenten ab und Letzteres extrapoliert beispielsweise statistische Daten von Verkehrsunfällen.

Die subjektive Wahrnehmung von Risiken hängt von mehr als 50 Faktoren ab. Hier ein paar der besonders relevanten Einflussparameter:

Kontrolliertbarkeit: Die meisten Menschen empfinden das Autofahren, als weniger riskant als in ein Flugzeug zu steigen, obwohl statistisch gesehen letzteres viel sicherer ist. Die Tatsache, dass sie das Ruder buchstäblich in der Hand haben, vermittelt ein größeres Gefühl der Sicherheit (SCHNIEDER & SCHNIEDER, 2013). Dies wurde ebenso durch eine Studie im Kontext der Risikowahrnehmung von Kurven belegt. Hierbei wurde festgestellt, dass Kurven, die weit einsehbar sind, dem Fahrer, aufgrund der vermeintlichen Kontrollierbarkeit, ein sicheres Gefühl vermitteln und zu einer ungünstigen Anpassung des Fahrverhaltens führen. Es hat sich im Zuge dessen sogar herausgestellt, dass Kurven, die dem Fahrer ein subjektiv unsicheres Gefühl vermitteln, beispielsweise durch gezielte Bepflanzungen, zu einem besseren Fahrverhalten und damit zu weniger Unfällen führen (SCHADE & ENGELN, 2008).

Katastrophenpotential: Ebenso werden Risiken, die mehrere Personen in einem einzelnen Ereignis treffen können, wie bei einem Flugzeugabsturz, als größer empfunden. Zusätzlich werden diese durch die mediale Berichterstattung emotional verstärkt.

Freiwilligkeit: Freiwillig eingegangene Risiken wie das Rauchen oder Fahrradfahren ohne Helm werden trotz ihrer statistisch gesehenen viel größeren Gefahr deutlich unterschätzt

im Verhältnis zu unfreiwillig eingegangen Risiken, wie etwa die Strahlungsgefahr von 5G Funktürmen.

Wissenschaftlicher Kenntnisstand: Das zuletzt genannte Beispiel der 5G-Funktürme trifft ebenso für diesen Faktor zu. Der Mangel an Wissen über die neue Technologie führt leichter zu einer Überschätzung der davon ausgehenden Gefahr.

Aktive vs. passive Risikowahrnehmung

Als aktiver Risikowahrnehmer wird der Beobachter erster Ordnung bezeichnet, also derjenige, der im Auto sitzt und fährt. Der Beobachter zweiter Ordnung ist der Beifahrer oder Passant, der das Geschehen passiv wahrnimmt. Umfragen haben gezeigt, dass Fahrer ihre eigenen Fähigkeiten tendenziell im Vergleich zu anderen Fahrern überschätzen und dass dieser Neigung Männer mehr unterliegen als Frauen 4.4.

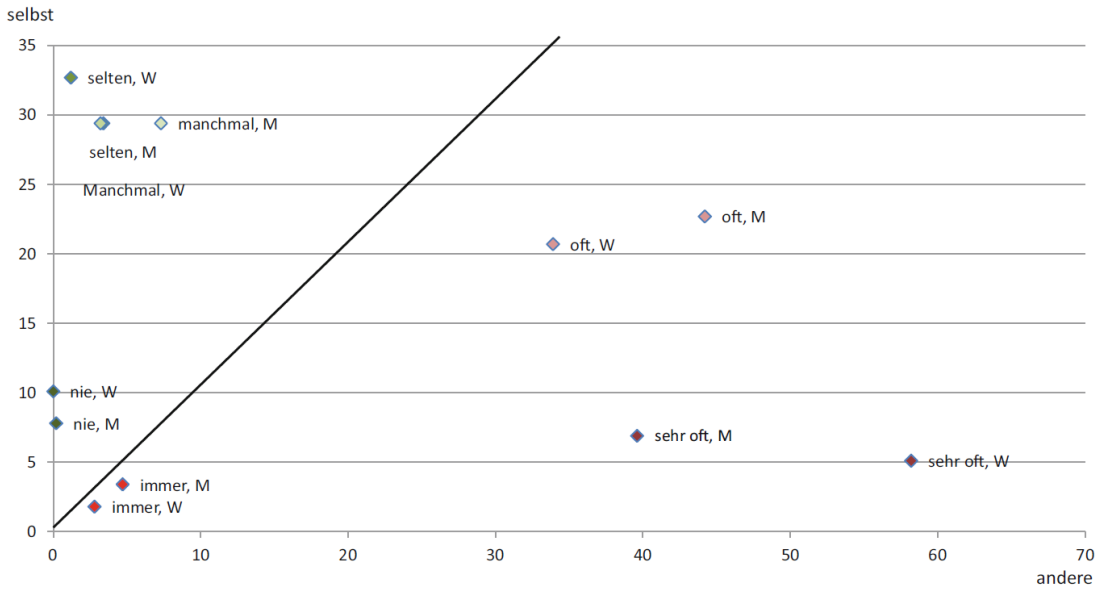


Abbildung 4.2: Fremd- und Selbstwahrnehmung (SCHNIEDER & SCHNIEDER, 2013)

Diese Erkenntnisse wurden zusätzlich durch Untersuchungen der Hirnaktivitäten mittels der funktionalen Magnetresonanztomografie (fMRT) bestätigt. Diese zeigten nämlich deutlich höhere Aktivitäten bei Beifahrern als bei den Fahrern selbst, da Beifahrer zusätzlich den Fahrer im Kontext des Fahrens miterleben, nicht dem Entscheidungsdruck ausgesetzt sind und nicht im gleichen Maß an den Vorteilen der Entscheidung teilhaben.

4.2.2 Modelle und Modellierung

Modellierung des menschlichen Verhaltens

Das menschliche Verhalten zu modellieren ist sehr schwierig und psychische Prozesse lassen sich zumeist nur indirekt, wie etwas über den Hautschweiß oder die Herzfrequenz, messen. Im Folgenden werden auf die Faktoren eingegangen, die im Menschen ein

Fehlverhalten erzeugen und das Leitermodell von Rasmussen erläutert, das die Entscheidungsfindung abbildet.

Menschliche Fehlhandlungen haben unterschiedlichste Gründe. Um sie konzeptionell besser verstehen zu können, werden diese wie folgt kategorisiert.

Rezeptionsstörung: Durch die Ermüdung der Augenmuskulatur werden Überwachungs- und Steuertätigkeiten beeinträchtigt.

Wahrnehmungsstörung: Der Verkehrsteilnehmer interpretiert einen visuellen Reiz falsch. Ein Beispiel wäre, dass ein Hinweis nicht richtig erkannt und somit die Vorfahrt genommen wird.

Koordinationsstörung: Durch langsamere Bewegungsabläufe leidet die motorische Präzision, besonders bei Positionierungsarbeiten.

Aufmerksamkeits- und Konzentrationsstörungen: Vor allem Arbeitsschritte, die kontinuierlich und repetitiv durchgeführt werden müssen, wie Steuer- und Kontrolltätigkeiten, werden mit verminderter Leistung ausgeführt.

Störungen des Denkens: Die Bildung von Assoziationen und Begriffen ist gestört und das Erinnerungsvermögen gemindert, was die Kontrolltätigkeit geordneter assoziativer Abläufe negativ beeinflusst.

Störung der sozialen Beziehungen: Durch aggressive oder egozentrische Tendenzen können affektive Reaktionen folgen, die die Funktionsfähigkeit in einem Team oder in der Kollaboration im Verkehr mindern (SCHNIEDER & SCHNIEDER, 2013).

Um die kognitiven Prozesse darzustellen, hat Rasmussen ein Modell dargestellt, das von Werther (WERTHER, 2006) als das bedeutendste Modell zur Handlungsregulierung im Bereich der Arbeitswissenschaft bezeichnet wurde. In diesem werden die Informationsverarbeitungsprozesse schematisch als Leitermodell dargestellt. Die sogenannte Rasmussenleiter besteht aus den vier Ebenen Strategisch, Dispositiv, Taktisch und Operativ, während der Prozess der Entscheidungsfindung als Petrinetz dargestellt wird. Hierbei verläuft die wissensbasierte Analyse aufsteigend und die wissensbasierte Planung absteigend und der Prozess beginnt mit der Aktivierung des Handlungsbedarfs. Darauf folgt die Informations- und Datenbeschaffung in Form des Beobachtens und die Identifikation der Gegebenheiten. Die Gewichtung zwischen den Instanzen Dispositiv und Strategisch hängt von der Eindeutigkeit des Zustands ab, also davon, ob die Situation mehr oder weniger bewusst bewertet werden muss. Zur Erreichung des erwünschten Zielzustands werden zuerst entsprechende Maßnahmen und dann Handlungsweisen festgelegt und letztendlich ausgeführt. Da sich die Fähigkeiten eines Menschen durch Erfahrungen und Erkenntnisse verändern, müssen im Rasmussenmodell auch Abkürzungen dargestellt werden. Hierfür werden regelbasierte Abkürzungen durch punktierte und assoziative Wissensverknüpfungen durch gestrichelte Linien dargestellt. Die Abkürzung zwischen Aktivierung und Ausführung stellt die sensomotorischen Fertigkeiten dar, also die Handlungen, über die sozusagen nicht mehr nachgedacht werden müssen.

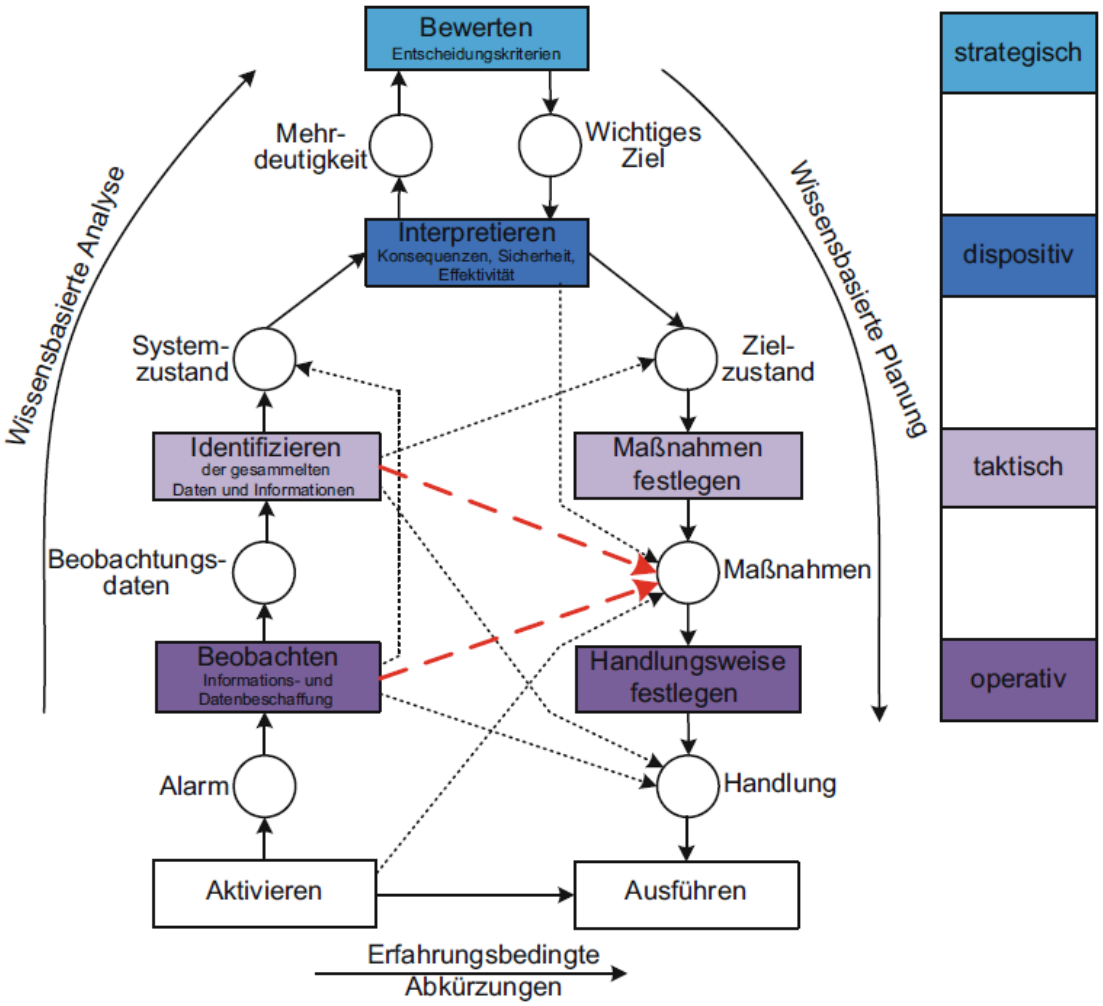


Abbildung 4.3: Rasmussenleiter und Petrinetzdarstellung der Entscheidungsfindung (SCHNIEDER & SCHNIEDER, 2013)

In der Verkehrssicherheitstheorie werden neben dem menschlichen Verhalten auch viele andere Modelle, wie etwa technisch-physikalische und systemdynamische Modelle entwickelt. Alle im Detail zu betrachten würde den Rahmen dieser Arbeit übersteigen, weshalb an dieser Stelle nur ein kurzer Einblick in die beiden genannten Modellarten gegeben wird.

Technisch-physikalisches Modell

Ein sehr intensiv genutztes technisch-physikalisches Modell im Ingenieurwesen ist die Finite Elemente Methode (FEM). Dieses eignet sich, um Karosserien und Materialeigenschaften zu simulieren, um bei gleichzeitiger Wirtschaftlichkeit die maximale Fahrgastsicherheit zu gewährleisten (SCHNIEDER & SCHNIEDER, 2013). Hierzu wird die Struktur durch unendlich kleine Elemente dargestellt, die die Materialeigenschaften wiedergeben und so für Berechnungen genutzt werden können.

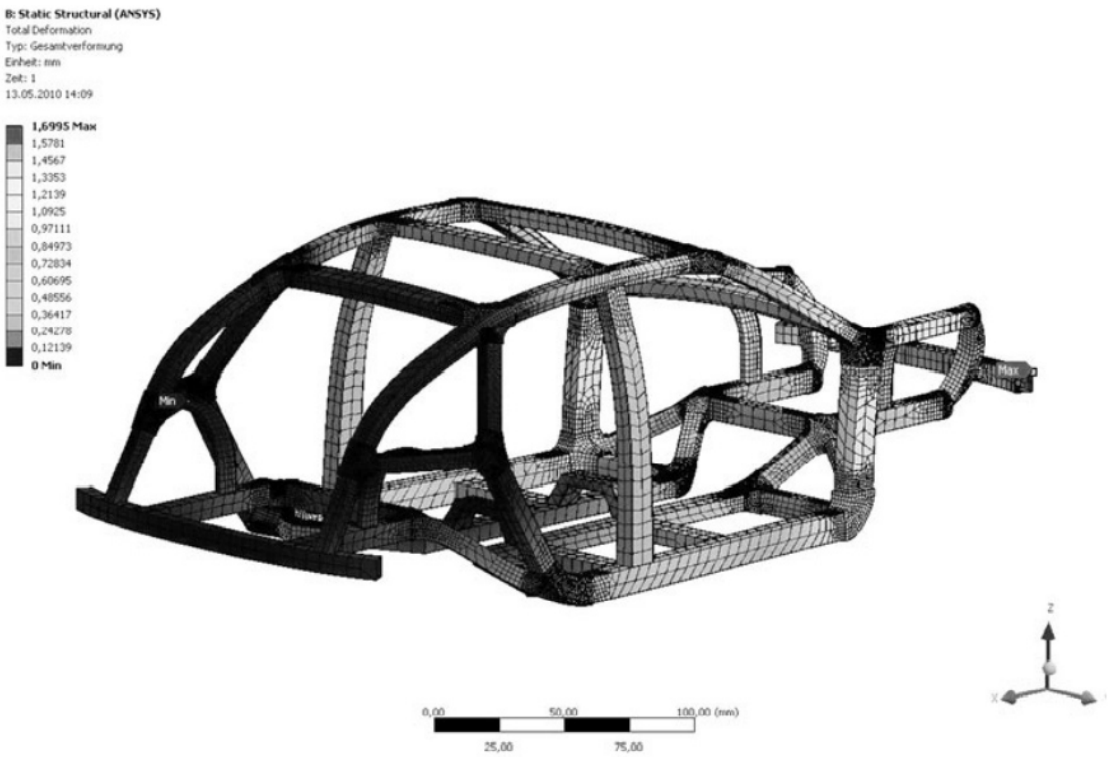


Abbildung 4.4: Fahrzeug als Finite Elemente Modell (SCHNIEDER & SCHNIEDER, 2013)

Systemdynamische Modelle

Da Verkehr kein statisches, sondern ein dynamisches System ist, wird, um kritische Aspekte für die Verkehrssicherheit zu erkennen, das Regelkreismodellkonzept verwendet. Die Erstellung solcher kybernetischer Strukturen setzt ein fundiertes Verständnis der relevanten Einflussfaktoren und der Wirkungszusammenhänge voraus. Um ein Höchstmaß an Vollständigkeit zu erreichen wird sich beim Entwurf an klare Strukturen, Prozessmodelle und Checklisten gehalten. Komponenten eines solchen Regelkreises sind im Kontext eines Autofahrers der Regler (Autofahrer), die Regelstrecke (realisierte Trajektorie des Fahrzeugs), das Messglied (Autofahrer in Interaktion mit den Instrumenten) und die Stellglieder (Bremse, Lenkrad etc.). Hierbei wäre der Seitenwind ein Beispiel für eine Störgröße. Die Illustration 4.5 zeigt ein Blockschaltbild, das die Querdynamik eines Fahrers unter Alkoholeinfluss darstellt. Dieses berücksichtigt eine verzögerte Wahrnehmung und Reaktion.

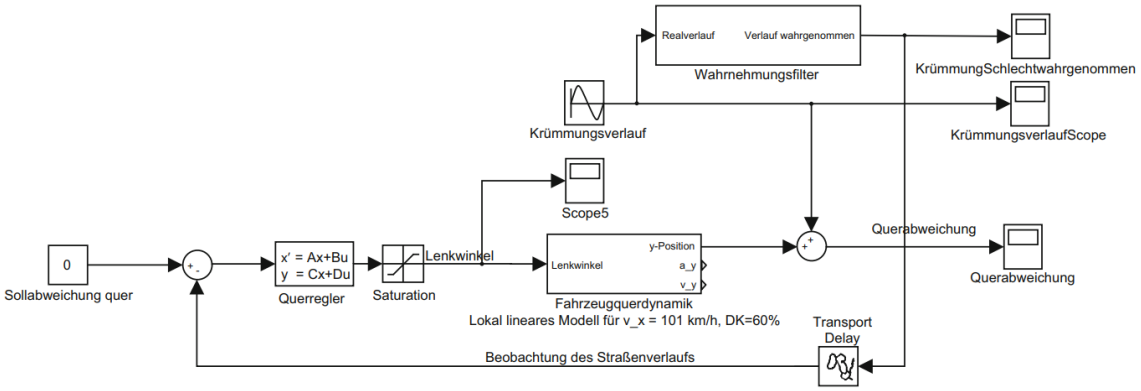


Abbildung 4.5: Blockschaltbild der Querregelung mit Wahrnehmungseffekten (SCHNIEDER & SCHNIEDER, 2013)

4.2.3 Präventive Maßnahmen

Um die wissenschaftlich Erkenntnisse in einen sicheren Verkehr umsetzen zu können, gibt es verschiedene Realisierungskonzepte. Eines der Implementierungsmethoden der Verkehrssicherheitskonzepte sind die Maßnahmen der Prävention. Hier gibt es neben der Gestaltung der Verkehrsmittel, der Vorgabe für Verkehrsobjekte und der Verkehrsorganisation die Gestaltung der Verkehrswegeinfrastruktur, ein Aspekt der durch die Auditoren geprüft wird. In Bestandsanlagen kann das Sicherheitsaudit zwar als eine reaktive Maßnahme auf eine nicht ausreichend sichere Verkehrsanlage angeordnet werden, im Kontext der Verkehrssicherheit gelten die aufgrund des Sicherheitsaudits angeordneten Maßnahmen jedoch immer als präventiv. Im Folgenden werden einige Beispiele genannt, um das Konzept zu veranschaulichen.

Im Bereich des Schienenverkehrs können sich diese Maßnahmen auf die automatischen Bahnsteigtüren mit den installierten Sensoren oder auf die Schutzräume neben den Gleisen beziehen. Der Straßenverkehr hat eine noch größere Vielfalt an Möglichkeiten, da hier jeder einzelne Fahrer betrachtet werden muss. Die technischen Regelwerke (siehe Kapitel 4.3) sind so ausgelegt, dass sie für eine möglichst sichere Anlage sorgen. Zusätzlich sind Defizitlisten aufgestellt worden, die zur Überprüfung der Planung dienen. Unter der Vielzahl der Kriterien spielen die Linienführung und die Kurvenradien eine zentrale Rolle, die so gestaltet sein müssen, dass einerseits die Fahrer selbst bei höheren Geschwindigkeiten die Spur halten können und andererseits die Monotonie einer Strecke durchbrochen wird, um durch bewusst gesetzte Stimuli der Müdigkeit entgegenzuwirken. Weiterhin müssen die Straßen in Bereichen von Kreuzungen und Einfahrten so konstruiert sein, dass für eine ausreichend große Haltesichtweite beziehungsweise Sichtweite gesorgt ist. Regelquerschnitte geben optimale Fahrstreifenbreiten vor, ebenso ist klar geregelt, wie nicht motorisierte Verkehrsteilnehmer durch Fuß- und Radwege vom KFZ-Verkehr getrennt werden.

Mittels den genannten Methoden und Modellen kann also ein wissenschaftlicher Ansatz erschaffen werden, der es ermöglicht, die notwendigen Maßnahmen rational zu begründen.

In der Verkehrssicherheitstheorie gibt es noch viele weitere Felder, die auf die Statistik, die Risikoabschätzung und die Techniken und Methoden eingehen. Die in diesem Kapitel genannten Aspekte, der Verkehrspsychologie, der Modellierung der Verkehrsdynamik über kybernetische Modelle und der präventiven Maßnahmen dienten der Veranschaulichung der Auswirkung dieser wissenschaftlichen Erkenntnisse auf die Leitkriterien des Sicherheitsaudits.

4.3 Richtlinien

Dieses Kapitel fasst die relevantesten Richtlinien zusammen, die darauffolgend im Kontext des Sicherheitsaudits der LBD praxisbezogen verwendet werden.

Richtlinien für die Anlage von Landstraßen (RSAS)

Die RSAS konzentrieren sich auf die Prozesse und Verfahren des Sicherheitsaudits. Grundsätzlich wird das Sicherheitsaudit in die Kategorien präventiv und reaktiv unterteilt. Alle Sicherheitsaudits in der Planung fallen unter erstere Kategorie, wohingegen das Sicherheitsaudit im Bestand meist unter letztere Kategorie fällt, da diese anlassbezogen durchgeführt werden. Handelt es sich jedoch um eine Änderung im Bestand, kann es ebenso als präventiv eingestuft werden. Das Ziel ist immer eine Minderung von Unfällen und das Vermeiden von Unfallfolgen. Es dient somit als eine Qualitätssicherung für eine sichere Anlage für alle Verkehrsbeteiligten. (RSAS, 2019).

Richtlinien für die Anlage von Landstraßen (RAL)

Die RAL sind technische Regelwerke und bilden das primäre Nachschlagewerk für den korrekten Entwurf von Landstraßen. Seine zentralen Ziele sind Verkehrssicherheit, Verkehrsqualität und Umweltverträglichkeit. Es beinhaltet Definitionen für Entwurfsklassen, Skizzen für Regelquerschnitte, Grundlagen für die Einhaltung der erforderlichen Sichtweiten, Knotenpunktgestaltungen und Ausstattungselemente wie Fahrzeug-Rückhaltesysteme (FGSV, 2012b).

Empfehlungen zum Schutz vor Unfällen mit Aufprall auf Bäume (ESAB)

Bei einem Unfall auf der Landstraße sterben die meisten Menschen statistisch gesehen durch einen Aufprall auf einen Baum. Dieser Fakt zusammen mit der Tatsache, dass Straßenbepflanzungen landschaftsprägende Elemente sind, die teilweise sogar historischen Wert besitzen, erzeugt, eine Verantwortung diese landschaftsgerecht wiederherzustellen oder neu zu errichten und gleichzeitig ein möglichst geringes Unfallrisiko zu erzeugen. Hierfür wurden die ESAB entworfen. Sie definieren die Verfahren, um auffällige Bereiche aufzufinden und gibt Maßnahmen vor, die Aufpralle nach Möglichkeit vermeiden und Unfallfolgen verringern sollen (FGSV, 2006).

Merkblatt zur Örtlichen Unfalluntersuchung in Unfallkommissionen (MUKO)

In diesem Merkblatt wird die Methodik der systematischen Unfallkommission beschrieben. Es wird erläutert, wie Unfalltypenkarten geführt und Unfallschwere und Unfallkatego-

rie definiert werden. Weiterhin wird gezeigt, wie Unfallhäufungen festgelegt und mittels Diagrammen und Ortsbesichtigungen analysiert werden. Danach werden die Schritte erläutert, mittels derer die Maßnahmen zur Vermeidung beziehungsweise Minderung der Unfälle durch die Unfallkommission bestimmt werden und die Rahmenbedingungen gegeben, die zur Prüfung der Umsetzungsmaßnahmen und Wirksamkeit notwendig sind (FGSV, 2012a).

Merkblatt für die Ausbildung und Zertifizierung der Sicherheitsauditoren von Straßen (MAZS)

Hier werden die Anforderungen an den Auditor bestimmt, die ein abgeschlossenes einschlägiges Hochschulstudium und eine mehrjährige Erfahrung auf dem Gebiet der Planung von Straßenverkehrsanlagen oder im Bereich straßenbezogener Sicherheitsuntersuchungen voraussetzt. Weiterhin wird die Ausbildung und die darin enthaltenen Module definiert, von denen mindestens vier abgeschlossen werden müssen. Ein Großteil der MAZS besteht aus Tabellen, die den Inhalt dieser Module bestimmen. Die Ausbildung wird mit einem Leistungsnachweis abgeschlossen. Das daraufhin erhaltene Zertifikat gilt für drei Jahre und kann unter bestimmten Rahmenbedingungen verlängert werden. Im letzten Kapitel wird auf die Website verwiesen, die die Ausbildungsstellen angeben und alle aktiven Auditoren auflistet (MAZS, 2019).

Richtlinien zum Planungsprozess und für die einheitliche Gestaltung von Entwurfsunterlagen im Straßenbau (RE)

Die RE2012 sind umfangreiche Richtlinien, die die Planungsprozesse für den Neu-, Um- und Ausbau von Bundesfernstraßen definieren und Anforderungen an Umfang und Inhalt bestimmen. Das Ziel ist es, die Prozesse zu beschleunigen, sie durch einheitliche Kommunikations- und Abstimmungsprozesse sicherer zu machen, die Verständlichkeit zu verbessern und den Qualitätsstandard zu sichern. Hierzu sind entsprechende Mustervorlagen und Planformate gegeben (BMVBS, 2012).

Empfehlungen für Radverkehrsanlagen (ERA)

Die ERA bilden die Grundlage für Planung, Entwurf und Betrieb von Radverkehrsanlagen und sind primär für den Neubau ausgelegt, werden aber auch für die Anwendung bei bestehenden Straßen empfohlen. Sie gelten innerhalb und außerhalb bebauter Gebiete. Neben der Querschnittgestaltung werden Führungsformen für unterschiedliche Bereiche, wie Hauptverkehrsstraßen und Knotepunkte oder Überquerungsanlagen bestimmt. Weiterhin werden bautechnische Aspekte, der Betrieb und die Methoden der Wirkungskontrolle und Qualitätssicherung definiert (FGSV, 2010).

Richtlinien für integrierte Netzgestaltung (RIN)

Die RIN sind Vorgaben für eine verkehrsübergreifende Betrachtung, die sich an den Regeln der Raumordnung und Landesplanung orientieren. Somit kann sich die Verkehrsnetzgestaltung entsprechend der Raumordnungsziele des jeweiligen Bereiches entwickeln. Daneben werden auch die Umwelt- und landschaftsbezogenen Planungsziele erwogen.

Die Richtlinien gelten neben dem KFZ-Verkehr, für den öffentlichen Personenverkehr und den Rad- und Fußverkehr (FGSV, 2008).

Handbuch für die Bemessung von Straßenverkehrsanlagen (HBS)

Die HBS gliedert sich in die drei Teile Autobahnen, Landstraßen und Stadtstraßen und enthält Beispielsammlungen für Berechnungen und Formblätter. Diese dienen der Ermittlung und Bewertung der Kapazität der Straßenverkehrsanlage und der Qualität des Verkehrsablaufs. Die unterschiedlichen Verkehrsanlagen werden durch entsprechende Qualitätskriterien bewertet. Beispiele für die Verfahren bei Landstraßen sind die Ermittlung der Fahrtgeschwindigkeit bei freien Strecken oder die Bewertung der Verkehrsqualität in Bezug auf die Kapazität der Linksabbieger bei bedingt verträglichem Abfluss bei Knotenpunkten mit Lichtsignalanlage (FGSV, 2015).

Straßenverkehrsordnung (StVO)

Jeder, der einmal eine Führerscheinprüfung absolviert hat, kennt die StVO. Sie ist die Rechtsverordnung, die für jeden Verkehrsteilnehmer gültig ist. Der erste Teil regelt das Verhalten im Straßenverkehr, der zweite Teil enthält die Bedeutung der Verkehrsschilder und der letzte definiert die Bußgelder.

4.4 Sicherheitsaudit der LBD

Hintergrund

Das in 1999 gegründete SAS (Sicherheitsaudits für Straßen) nutzte die im Ausland gesammelten Erfahrungen, um ein Auditverfahren zu entwickeln, welches in den Empfehlungen für das Sicherheitsaudit von Straßen im Mai 2002 veröffentlicht wurde (GERLACH, 2004). Am 14.08.2003 wurden die ESAS erstmalig in Bayern eingeführt. Diese bildeten die Grundlage für die heutigen Richtlinien für das Sicherheitsaudit von Straßen (BMVI, 2019b). Für alle Bundesfernstraßen ist es nach BMV ARS 4/19 (BMVI, 2019b) Vorgabe ein Sicherheitsaudit für die Vorplanung, die Entwurfs- und Genehmigungsplanung durchzuführen, außerdem wird dieses in der Richtlinie 2008/96/EG seit dem 19.11.2008 ebenso für Straßen des transeuropäischen Straßennetzes (TEN-T) in den Phasen des Ausführungsentwurfs, der Fertigstellung und der ersten Betriebsphase gefordert (EU-PARLAMENT, 2008). Neben den genannten Richtlinien spielen noch das Merkblatt für die Ausbildung und Zertifizierung der Sicherheitsauditoren von Straßen (MAZS), die Straßenverkehrsordnung (StVO) und die Richtlinien für die Anlage von Landstraßen eine zentrale Rolle.

Ablauf

Die Arbeit des Sicherheitsauditors kann mit der des Prüfstatikers verglichen werden und genauso wie dieser muss der Auditor vor Beginn seiner Tätigkeit geschult und in die notwendigen Richtlinien eingewiesen werden. Die Auditprozesse richten sich hierbei nach der RSAS und MAZS, wohingegen die geprüfte Planung unter anderem den Vorgaben der RAL entsprechen muss. Vor dem Sicherheitsaudit in der Planung gibt es entsprechend der

Richtlinie 2008/96/EG (EU-PARLAMENT, 2008) Sicherheitsabschätzungen nach den Richtlinien für Wirtschaftlichkeitsuntersuchungen an Straßen (RWS), dem Handbuch für die Bewertung der Verkehrssicherheit von Straßen (HVS) und dem Bundesverkehrswegeplan (BVWP).

Der Auftraggeber initiiert das Sicherheitsaudit durch Beauftragung eines unabhängigen Auditors, ein Kriterium, das erfüllt ist, wenn er keine Projektverantwortung trägt und nicht bei dem Entwurf beteiligt war. Die zentralen Fragen eines Sicherheitsaudits nach RSAS sind:

- Ist eine sichere Benutzung für alle Verkehrsteilnehmer möglich?
- Wird die Verkehrsanlage regelkonform genutzt?
- Ist im Hinblick auf die Verkehrssicherheit die optimale Gestaltung gewählt worden?
- Lassen die Erkenntnisse eine andere Entwurfsausbildung oder Verbesserung sinnvoll erscheinen?

Das Audit selbst unterteilt sich in folgende Phasen:

1. Vorplanung (VP)
2. Entwurfsplanung (EP)
3. Ausführungsplanung (AP)
4. Vor Verkehrsfreigabe (VF)
5. Nach Verkehrsfreigabe (NF) (ërste Betriebsphase")

Die Phasen 1 und 2 erfolgen vor der Genehmigung und 3 vor der Freigabe zur Bauausführung. In der Auditphase 4 wird insbesondere auf die Sichtverhältnisse, Markierungen und Beschilderung eingegangen und in der Phase 5 wird geprüft, ob die Verkehrsteilnehmer die Verkehrsanlage regelkonform nutzen. Weiterhin wird sichergestellt, dass durch die Bepflanzungen ein halbes Jahr nach Freigabe keine Sicherheitsmängel entstanden sind.

Der Arbeitsablauf kann entsprechend dem MAZS (MAZS, 2019) und der Schulungsmaterialien von Kuehn (KUEHN, 2019) wie folgt zusammengefasst werden:

1. Prüfung der Unterlagen auf Vollständigkeit
2. Auditierung der Planunterlagen durch:
   (a) Virtuelle Benutzung der Verkehrsanlage aus der Sicht aller Verkehrsteilnehmer
   (b) Überprüfung der sicheren Gestaltung wichtiger räumlich-funktionaler Situationen, wie etwa:
      - Ortseingangsbereiche

- Überquerungsstellen
- Haltestellenbereiche

(c) Überprüfung der Planung auf mögliche missbräuchliche Benutzung (Beispiel: Gehwegparken)

3. Selbstkontrolle mit Checklisten
4. Ortsbesichtigung, Prüfung der Planung vor Ort
5. Abfassung eines schriftlichen Auditberichts

Dieser wird an den Auftraggeber übergeben, der dann entscheidet, ob und inwieweit er auf die genannten Defizite eingeht. Klar zu differenzieren ist an dieser Stelle, dass es nicht die Aufgabe des Auditors ist die Arbeit des Planers zu übernehmen und er daher auch keine konkreten Änderungen des Entwurfs vornehmen darf.

Im Arbeitsablauf der MAZS fallen mehrere Punkte auf, die für das Thema dieser Arbeit unmittelbar relevant sind. Diese sind: "Virtuelle Benutzung", "räumlich-funktionale Situationünd "Kontrolle mit Checklisten". Letztere, auch Defizitlisten genannt, wurden zur Unterstützung des Auditors durch die Bundesanstalt für Straßenwesen (BASt) veröffentlicht und entsprechen der aktuellen Sicherheitsforschung (BMVI, 2019b). Sie enthalten eine umfangreiche Auflistung an Defiziten, die in der jeweiligen Auditphase zu beachten sind. Die Punkte, die am häufigsten im Zusammenhang mit schweren Unfallereignissen stehen, werden Kerndefizite genannt und da diese die primäre Grundlage für die in dieser Arbeit entwickelten ACCC bilden, wird im Folgenden eine Übersicht über die einzelnen Punkte gegeben (WÜST, 2019).

Unsicherer Seitenraum

Die RAL definiert einen sicheren beziehungsweise unsicheren Seitenraum durch drei Kriterien. Als sicher gilt eine Straße erstens, wenn dessen Verlauf durch geeignete Bepflanzung verdeutlicht wird, jedoch zweitens gleichzeitig unattraktiv für Wild ist. Drittens gilt es Hindernisse wie Bäume und Pfeiler zu vermeiden oder vor Anprall zu sichern. Hierbei gilt für Strauchpflanzen mit einer maximalen Stammdicke von 0,08 m ein Mindestabstand von 3,00 m vom befestigen Fahrbahnrand. Bäume, die neu gepflanzt werden, dürfen nur an von Fahrzeugen nicht erreichbaren Stellen platziert werden und selbst hinter Fahrzeugrückhaltesystemen ist ein Mindestabstand von 3,00 m geboten. Die Empfehlungen zum Schutz vor Unfällen mit Aufprall auf Bäume (ESAB) empfehlen sogar einen Abstand von mindestens 4,5 m, bei dem auf Fahrzeugrückhaltesysteme nur verzichtet werden darf, wenn der Streckenabschnitt nach dem Merkblatt zur örtlichen Unfalluntersuchung in Unfallkommissionen (MUko) nicht als Unfallhäufungslinie gilt. Weitere Ausnahmen bilden Gräben, die abkommende Fahrzeuge nicht überwinden können, Böschungen, die über 3,00 m über der Fahrbahn liegen oder Bäume, die aus anderen Gründen vorhanden sein müssen (Beispiel: Sicherung einer Hanglage).

Tabelle 4.1: Empfohlene Radien und Mindestlängen von Kreisbögen (FGSV, 2012b)

Entwurfsklasse	Radienbereiche R [m]	Mindestlängen von Kreisbögen min L [m]
EKL 1	>= 500	70
EKL 2	400 – 900	60
EKL 3	300-600	50
EKL 4	200-400	40

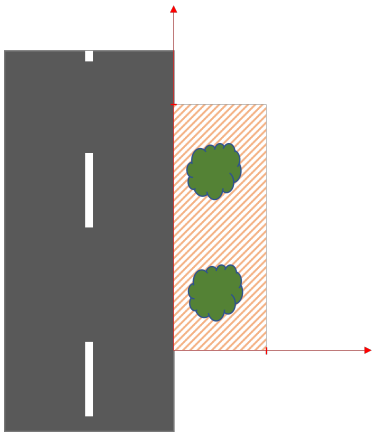


Abbildung 4.6: Unsicherer Seitenraum

Unstetige Radienfolge

Um eine adäquate Linienführung zu gewährleisten, ist zu prüfen, ob die "...Abfolge der räumlichen Elemente eine hinreichende Erkennbarkeit des Straßenverlaufs und einen optisch befriedigenden Gesamteindruck des Bauwerks Straße gewährleisten."(FGSV, 2012b, Blatt 19) Das bedeutet, dass die Straße sich einerseits an die topografischen Bedingungen anpassen muss, um in das Landschaftsbild zu passen und andererseits so zusammengesetzt sein muss, dass sie gut befahrbar ist. So ist eine gerade Landstraße entlang eines Kanals sinnvoll, sollte aber nicht die Länge von 1.500 m überschreiten, da dies zu schnellem Fahren verleitet und bei Nacht das Blenden des entgegenkommenden Verkehrs erhöht. Dementsprechend werden Landstraßen hauptsächlich in einer Abfolge von Kreisbögen geführt, dessen Radien, Übergangsbögen und Mindestlängen vorgegebenen Richtlinien folgen. So ist für die jeweiligen Entwurfsklassen eine Mindestlänge und ein entsprechender Radienbereich gefordert (siehe Tabelle 4.1). Die Entwurfsklasse entspricht im Regelfall der Straßenkategorie der Landstraße, welche in der RAL definiert sind und weicht nur in Einzelfällen bei über- oder unterdurchschnittlichem Verkehrsaufkommen ab. Für den Übergang von Gerade in Kreisbogen und von Kreisbogen in Kreisbogen gibt es vorgegebene Parameter, die über die Graphen (siehe Abb. 4.7) definiert sind. Als zentrales Nachschlagewerk dient hierbei die RAL und bei der Führung der Radwege werden die Empfehlungen der Radverkehrsanlagen (ERA) genutzt.

Um einen unstetigen Kurvenverlauf zu vermeiden, der für Fahrer gefährlich werden könnte werden zwischen den unterschiedlichen Kurven und zwischen den Geraden und Kurven

Übergangsbögen angeordnet. Diese werden als Klothoiden ausgebildet für die die Formel

$$A^2 = R \cdot L$$

gilt. Wobei A der Klothoidenparameter, R der Krümmungsradius und L die Länge des Kurvenbogens vom festgewählten Kurvenausgangspunkt zum betrachteten Kurvenpunkt ist. Hierbei soll die Klothoide im Bereich

$$\frac{R}{3} \leq A \leq R$$

liegen, für kleine Radien im oberen Bereich, für größere im Unteren und A soll möglichst nicht kleiner als 100 m sein. Auf die Klothoiden darf bei Radienübergängen verzichtet werden, die größer als 2000 m sind.

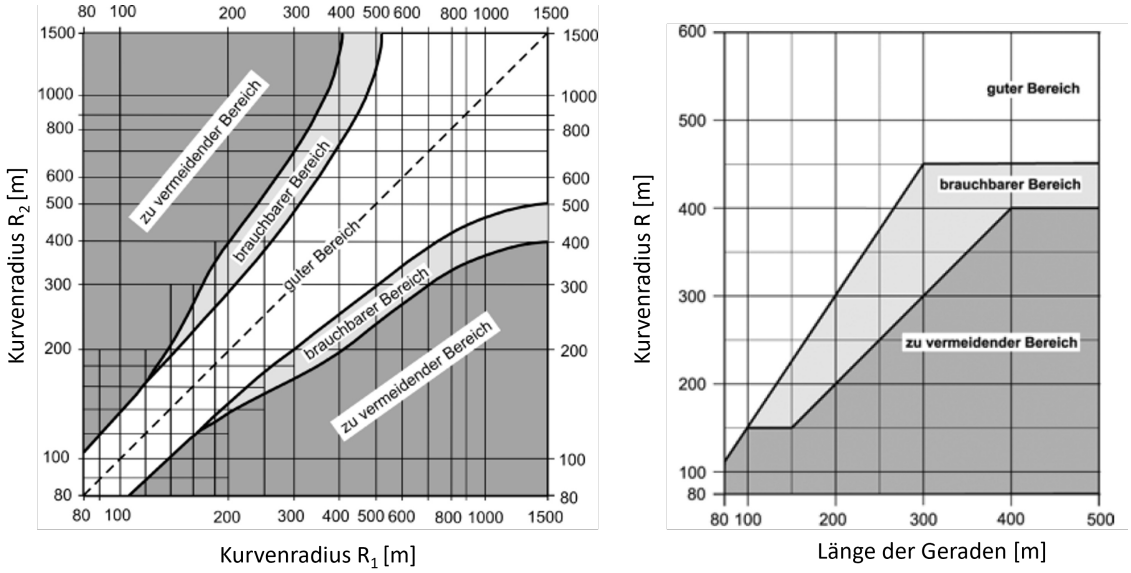


Abbildung 4.7: Verhältnis aufeinanderfolgender Radien [links] Übergang Gerade zu Kurve [rechts]

Eingeschränkte Anfahrsicht, unzureichende Haltesichtweite und Erkennbarkeit von Knoten

Ob ein Kraftfahrzeug an einer Kreuzung abbiegen, anhalten oder diese aus einer angemessenen Entfernung gut erkennen kann, hängt davon ab, ob die Linienführung dafür korrekt ausgelegt und das Sichtfeld nicht behindert ist (siehe Abb. 4.8). Nach der RAL gelten für PKWs eine anzusetzende Aughöhe von 1,00 m und für Lkws eine von 2,50 m. Die einzuhaltenden Sichtfelder richten sich hierbei nach der Entwurfsklasse (EKL) und der zulässigen Höchstgeschwindigkeit am Knoten.

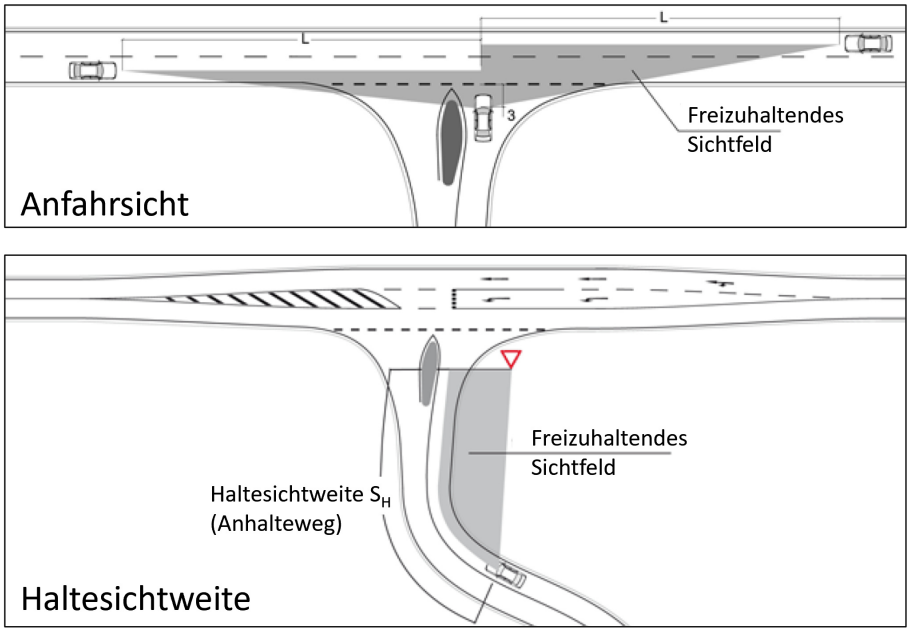


Abbildung 4.8: [oben] Anfahrsicht mit Schenkellänge L [unten] Haltesichtweite

Ebenso sind die minimalen Kuppen- und Wannenhalbmesser durch die EKL definiert. So hat ein Fehler in der räumlichen Linienführung, der zu große Sichtschattenbereiche durch Springen oder Tauchen verursacht (siehe Abb. 4.9) oder einen Kurvenbeginn verdeckt, einen hohen negativen Einfluss auf die Verkehrssicherheit. Ein Kurvenbeginn gilt als verdeckt, "[...] wenn aus einer Entfernung von 75 m vor dem Kurvenbeginn im Lageplan die vorausliegende Straßenoberfläche nicht mindestens bis zu dem Punkt einsehbar ist, an dem eine Richtungsänderung von 3,5 gon im Lageplan vorliegt."(FGSV, 2012b, Blatt 26) Eventuelle Defizite in der räumlichen Linienführung sind im Lage- und Höhenplan zu beseitigen, außer es kann mittels Persepektivbildern (siehe 4.9) nachgewiesen werden, dass die Sichtverhältnisse für den Kraftfahrer angemessen sind. Die erforderlichen Haltesichtweiten hängen von der Längsneigung und der EKL ab. Da der Fahrer genug Zeit zum Reagieren haben muss, sollten die Sichtweiten 30% über den erforderlichen Haltesichtweiten liegen. Können diese Grenzwerte nicht eingehalten werden, ist eventuell eine Beschränkung der Höchstgeschwindigkeit notwendig. Ebenso können Haltesichtweiten durch Böschungen, Lärmschutzwände oder andere Objekte im Sichtfeld beeinträchtigt werden. Sichthindernisse wie Böschungen oder schlecht platzierte Wegweiser können dafür sorgen, dass die Anfahrsicht beeinträchtigt und keine gefahrenfreie Einfahrt in eine bevorrechtigte Straße möglich ist. Ist die zulässige Höchstgeschwindigkeit der übergeordneten Straße nicht auf 70 km/h beschränkt, muss die Schenkellänge L mindestens 200 m betragen (siehe Abb. 4.8).

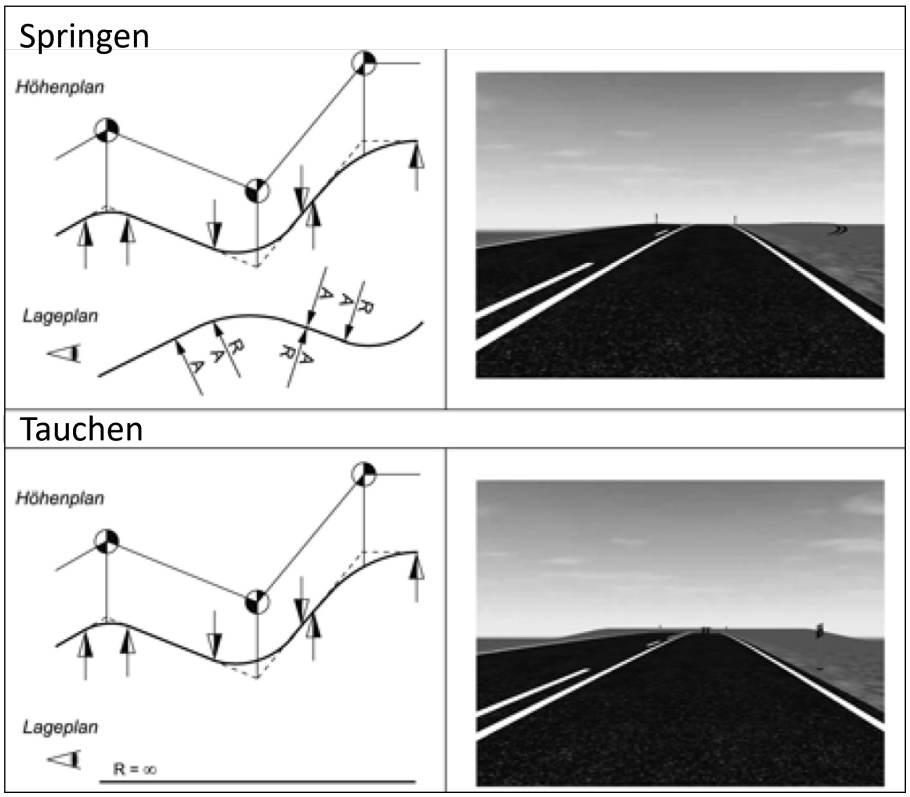


Abbildung 4.9: Perspektivbilder mit den Fällen Springen und Tauchen

Entwässerungsschwache Zonen, fahrdynamisch unzureichende Querneigung und mangelhafte Bankette

Aus ökologischen und wirtschaftlichen Gründen werden Landstraßen in der Regel offen entwässert. Ausnahmen können aus wasserwirtschaftlichen oder baulichen Gründen auftreten. Ist die Längsneigung, Querneigung oder Schrägverwindung zu gering beziehungsweise nicht vorhanden, kann das Wasser auf der Fahrbahn nicht abfließen, was Gefahren durch Spritzwasser und Aquaplaning verursachen kann. Auf langen Brücken und in Tunneln soll daher die Längsneigung mindestens 0,7% betragen und in Verwindungsbereichen mindestens 1,00% (besser 1,5%) und darf nur in begründeten Ausnahmefällen 0,7% betragen. Die Mindestquerneigung beträgt 2,5% und darf in Kurven sogar negativ sein, wenn der Radius über 3000 m beträgt, um eine Entwässerung zu garantieren. Ist der Radius kleiner, korreliert aus fahrdynamischen Gründen die Querneigung mit dem Kurvenradius. Hierfür bietet die RAL ein Diagramm zur Ermittlung der korrekten Querneigung (Abb. 4.10). Für ein gutes Spurhaltevermögen ist es wichtig, dass diese eingehalten werden. Darauf ist spezifisch in Übergangsbereichen von Ausbau- in Bestandsstrecken zu achten. Die maximale Querneigung sollte nicht mehr als 7,00% betragen. Um die Entwässerung auch in Verwindungsbereichen zu garantieren ist eine Mindestanrampungsneigung gefordert, die für die entsprechende EKL in der RAL definiert sind. Bankette und Seitenstreifen, über die entwässert wird, sollten 12% nach außen geneigt sein, ansonsten 6%. Außerdem ist vorgegeben, dass Bankette standfest ausgebildet werden und mindestens 1,5 m breit sind (in Einschnitten neben Mulden 1,00 m). Neben Geh- und Radwegen können sie auf 0,5 m reduziert werden.

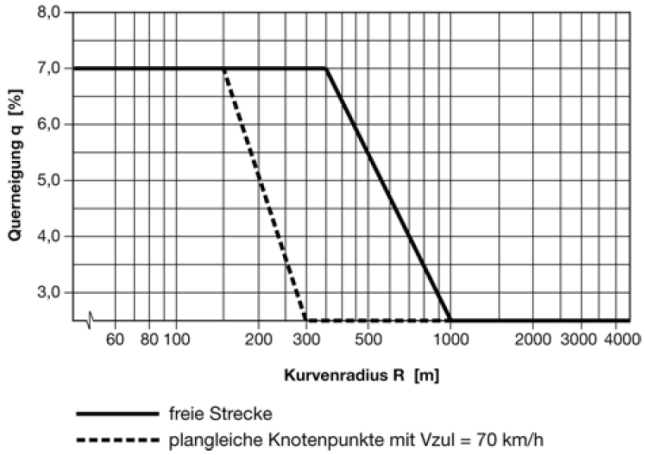


Abbildung 4.10: Querneigung in Abhängigkeit vom Kurvenradius

Unsicheres Radwegende und ungünstige Knotenpunktwahl

Schnittstellen zwischen Rad- und Kraftfahrzeugverkehr bilden Gefahrenquellen für Unfälle, vor allem wenn das Radwegende nicht sinnvoll geplant und der Übergang nicht klar zu erkennen ist. Nach der ERA müssen daher entsprechende Bordführungen oder Schutzinseln das Radwegende baulich deutlich hervorheben. Außerdem sollte ein Verpflechtungslänge von 10 bis 20 m eingehalten werden, die den Radweg in den Kraftfahrzeugverkehr überleitet.

Abbildung 4.11: Unsicheres Radwegende (KUEHN, 2019)

"Knotenpunkte sind so zu gestalten, dass eine sichere Führung der durchfahrenden sowie der ein-/abbiegenden und querenden Verkehrsströme gewährleistet ist."(FGSV, 2012b) Dies ist erfüllt, wenn der Knotenpunkt übersichtlich und verständlich aufgebaut, früh genug erkennbar und sicher befahrbar ist. Weiterhin sollten Knotenpunkte möglichst weit voneinander entfernt sein und die Distanz bei EKL 1 nicht weniger als drei Kilometer betragen und bei EKL 2 nicht weniger als zwei. Über die RAL ist definiert, welche Knotenpunktart

entsprechend der EKL zu wählen ist. Es werden planfreie, teilplanfrei, teilplangleiche und plangleiche Knotenpunkte unterschieden.

4.5 Der BIM-Standard in der LBD

In Bayern sollen ab dem Jahr 2025 alle neu beginnenden Neu-, Um- und Ausbauprojekte im Bereich der Bundesstraßen und Staatsstraßen die BIM-Methodik in der Planungs- und Bauphase anwenden. Dies entspricht den Vorgaben des BIM-Masterplans der BMVI (MEISTER et al., 2021), die vorhat, den BIM-Standard bundesweit bis 2027 im Regelbetrieb einzuführen. Um dieser Entwicklung eine klare Struktur zu geben, veröffentlichte die Landesbaudirektion den BIM-Leitfaden (KLEMT-ALBERT et al., 2021), der die stufenweise Einführung begleiten und orientieren soll. Die Grundidee war hierbei, die Beschäftigten zu informieren, sie durch ein Nachschlagewerk zu unterstützen und durch den realisierbaren Mehrwert zu motivieren. Im Folgenden wird auf dessen relevanteste Punkte eingegangen und der aktuelle Stand der BIM-Implementierung in der Landesbaudirektion erörtert.

4.5.1 Zielsetzung

Um allen Beteiligten eine klare Intention für die Etablierung der BIM-Methodik zu geben, legt die bayerische Staatsbauverwaltung vier grundlegende Ziele für die Digitalisierung des Straßen- und Brückenbaus fest. Diese sind über den gesamten Lebenszyklus des Projektes gültig. Als erstes Ziel wird die Effizienzsteigerung genannt, die durch eine erhöhte Transparenz und Konfliktreduktion erreicht wird und somit Genehmigung und Kommunikation beschleunigt. Das zweite Ziel ist die Förderung von projektorientiertem Denken, das interdisziplinäres Arbeiten favorisiert und den Projekterfolg priorisiert. Das dritte Ziel ist die Digitalisierung der Prozesse, um Arbeitsschritte zu automatisieren, Informationen leichter zugänglich zu machen und physische Dokumente einzusparen. Das letzte Ziel bezieht sich auf die Planungssicherheit. Hierbei ist die durch 4D- und 5D-BIM mögliche Verbesserung angepeilt, die für höhere Kosten- und Terminsicherheit sorgen soll und somit Nachträge reduziert.

4.5.2 Stufenplan

Geplant ist eine stufenweise Einführung durch Pilotprojekte, die auf didaktisch sinnvolle Weise es den Beteiligten ermöglichen soll, die BIM-Methodik kennenzulernen, zu verstehen und anzuwenden. Diese ist in drei Stufen eingeteilt (siehe Abb. 4.12).

Abbildung 4.12: Stufenplan des BIM-Leitfadens (KLEMT-ALBERT et al., 2021)

In der ersten Stufe, der Vorbereitungsphase, sollen durch erste Erfahrungen mittels Pilotprojekten in ausgewählten Bauämtern Grundlagen für die weitere BIM-Implementierung geschaffen werden. In der zweiten Stufe werden die Pilotprojekte auf alle Bauämter ausgeweitet und die ersten organisatorischen Strukturen in den operativen Einheiten etabliert. Hierbei wird die Leit- und Zentralstelle BIM (ZBIM) helfen, die Pilotprojekte einzurichten und durch Workshops das nötige Wissen an die jeweiligen BIM-Projektgruppen vermitteln. Die dritte Phase ist die Applikation. Ab 2025 soll BIM im vollen Umfang angewendet und die Beteiligten so weit geschult werden, dass sie in der Lage sind, ihre jeweiligen Positionen vollständig auszufüllen. So soll aus den BIM-Projektgruppen die BIM-Gruppe entstehen, die als Servicestelle des Amtes den Fachabteilungen bei BIM-spezifischen Leistungen zur Verfügung steht.

4.5.3 Datenmanagement der LBD

Für die Kollaboration wurde an der LBD der Open BIM-Ansatz gewählt, der als grundlegenden Datenstandard IFC und für die modellassoziierte Kommunikation BCF nutzt. Dadurch können die Bauämter softwareunabhängig agieren und haben gleichzeitig eine konsistente Datenlage. Um eine Single Source of Truth zu haben, wird mit einer CDE gearbeitet. Über diese können alle Projektbeteiligten jederzeit Zugang zum Modell und den projektrelevanten Daten erhalten, ohne das Risiko von Informationsverlust zu haben, die oft bei Planungsübergaben und Leistungsphasenübergängen vorkommen. Die Beteiligten tragen hierbei die Verantwortung, die Informationen regelmäßig abzurufen und zu aktualisieren. Durch den transparenten Verlauf ermöglicht die CDE eine eindeutige Kommunikation und einen qualitätsgesicherten Ablauf.

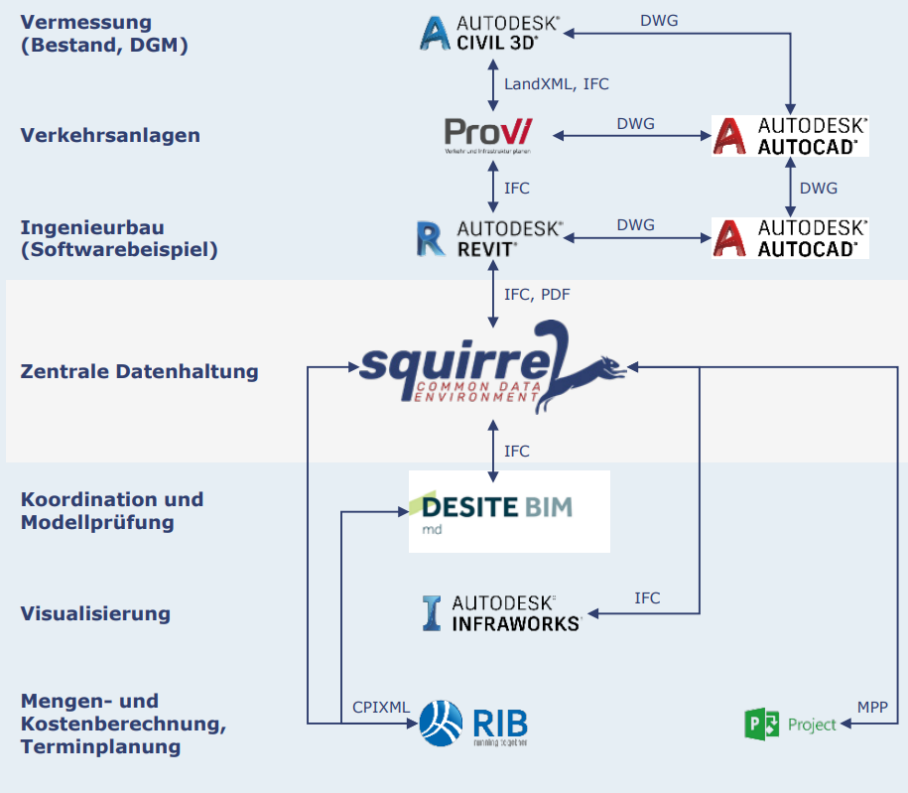


Abbildung 4.13: Verwendete Software der LBD

Das Pilotprojekt Ingolstadt verwendet die webbasierte CDE Squirrel (siehe Abb. 4.13). Die beauftragten Planer wurden hierfür in der Nutzung dieser Plattform geschult, um den korrekten und eigenständigen Umgang zu lernen und bei den entsprechenden Data Drops die jeweiligen Teilmodelle ihrer Disziplin hochladen zu können. Die Planer prüfen hierbei zusammen mit ihren Fachkoordinatoren die Modelle auf fachliche Richtigkeit und Plausibilität, während der BIM-Fachmanager die Übereinstimmung der Modellierung mit den Vorgaben der Auftraggeber-Informations-Anforderung (AIA) und BIM-Abwicklungsplan (BAP) verifiziert. Vor dem Abschluss einer Leistungsphase werden dann die Teilmodelle durch den BIM-Gesamtkoordinator als Koordinationsmodell im Big Data Drop, nach einer Qualitätsprüfung, die sowohl Modellierungs- als auch Planungsfehler umfasst, dem Auftraggeber zur Autorisierung vorgelegt.

Die Kommunikation erfolgt über das Modell mittels BCF. Hier können Probleme oder Informationen übermittelt und gleichzeitig dokumentiert werden, welche durch den Auftraggeber oder den BIM-Gesamtkoordinator zugewiesen werden. So werden Aufgaben erzeugt, die eine klare Zuständigkeit haben und ein projektorientiertes, interdisziplinäres Arbeiten gefördert. In regelmäßigen Abständen werden Besprechungen abgehalten, die dank der CDE modellbasiert sind. Alle Daten können hier entweder direkt über modellinhärente Informationen oder über Dokumente eingesehen werden, die über intelligente Links zugänglich sind.

4.5.4 Modellierung

Der Entwurf des Building Information Model folgt klaren Vorgaben. Die LBD hat sich hierfür ein eigenes LOD-Konzept definiert, das im Kern der in DIN-EN 17412 (DIN, 2020) definierten Level of Information Need (LOIN) entspricht, jedoch statt der Dokumentation als drittes Attribut den Zusatz Level of Accuracy (LOA) enthält. Diese wird im BIM-Leitfaden als eine Kombination aus der Lagegenauigkeit des Modells und der zulässigen Toleranz definiert. Somit besteht der LOD, hier austauschbar mit LOIN, aus den drei Instanzen LOG, LOI (siehe Kapitel 3.5) und LOA.

Um der jeweiligen Fachdisziplin oder Phase adäquat zu dienen wäre es unvorteilhaft immer ein Gesamtmodell zu nutzen. Dieses würde bei großen Modellen die Rechenleistung unnötig beanspruchen und das Projekt unübersichtlich machen. Daher hat die LBD ein Teilmodellkonzept definiert, das in verschiedene Gruppen eingeteilt wird. Unterschieden werden hier das Abschnittsmodell, das Koordinationsmodell, das Gesamtmodell und das Fachmodell. Ersteres enthält alle fachlichen Planungen innerhalb eines räumlichen Abschnitts. Das Koordinationsmodell besteht aus einer Auswahl an Teilmodellen, die spezifisch für einen Zweck wie eine modellbasierte Planungsbesprechung zusammengestellt werden und sich dabei auf konkrete Planungs- beziehungsweise Revisionsstände beziehen, die nicht zwingend die aktuellsten sein müssen. Das Gesamtmodell hingegen bezieht sich auf die Zusammenstellung aller Teilmodelle auf dem aktuellsten Stand. Diese werden für den Abschluss von Leistungsphasen genutzt und sollen das Projekt in seiner Gesamtheit festhalten. Für das Pilotprojekt des staatlichen Bauamts Landshut wurden die Fachmodelle der Kategorien Landschaftsplanung, Verkehrsanlagen, Vermessung, Ingenieurbau und Baugrund erstellt. Diese wurden wiederum entsprechend der Vorgaben der AIA und den Projektanforderungen durch den Auftragnehmer in einzelne Teilmodelle aufgeteilt (siehe Abb. 4.14).

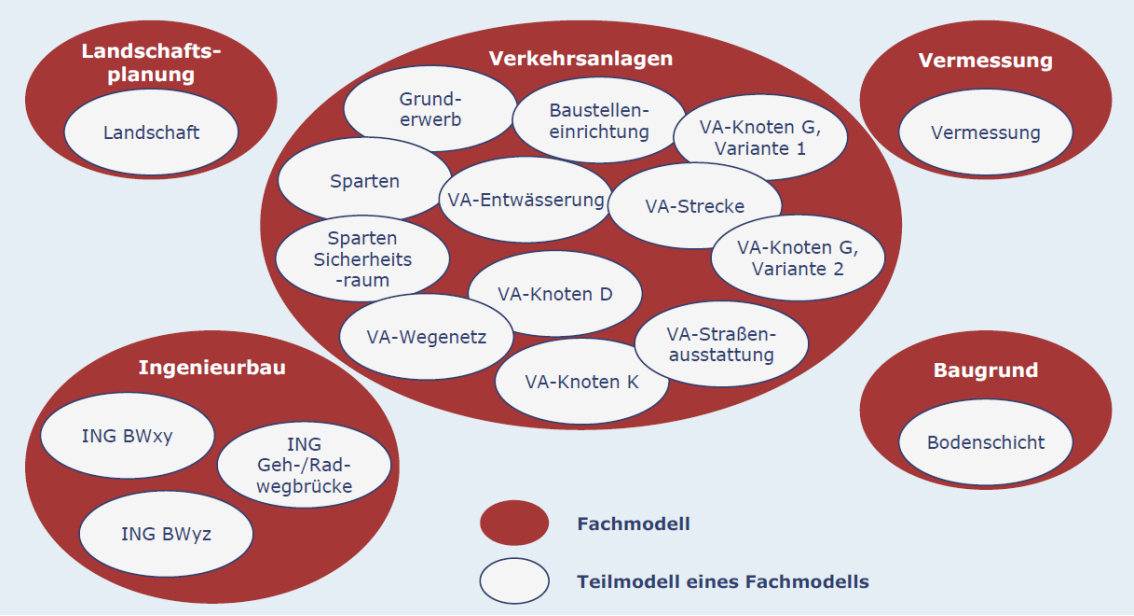


Abbildung 4.14: Teilmodellkonzept des Pilotprojekts Landshut

4.6 Qualitätsmanagement

In Kapitel 3.4 wird auf die Kriterien für die generelle Modellqualität eingegangen. Das Qualitätsmanagement der LBD hat daher viele Überschneidungspunkte, definiert die Kriterien aber aus einer unternehmerischen Perspektive. So hängt die Qualität des Modells auch hier im Wesentlichen von drei Kriterien ab. Das erste ist die Güte, mit der das Modell entworfen wurde. Kollisionen, also Objekte, die einander überschneiden, oder falsche semantische Informationen, beeinflussen diese zum Beispiel negativ. Das zweite Kriterium entspricht der fachlich-technischen Qualität der konventionellen Bauplanung, die durch die entsprechende AEC-Fachkraft, wie Statiker, HLSler oder Architekten bestimmt wird. Das letzte Kriterium wird durch die Datenqualität bestimmt, primäre Punkte hierbei sind wohlgeformte und vollständige Schemas und Aktualität der Daten. Der Auftragnehmer schuldet eine fehlerfreie, baubare Planung und Darstellung als Modell, weshalb sich der BIM-Fachkoordinator vergewissern muss, dass der Modellentwurf mit den Vorgaben der AIA und BAP übereinstimmt. Im Anschluss wird dies auftraggeberseitig verifiziert. Im Falle der einzelnen Fachmodelle geschieht das durch den BIM-Fachmanager, der BIM-Manager hingegen prüft dann, das daraus entstandene Koordinationsmodell. Die Modellprüfungen können dank BIM teilweise über Regelsätze erfolgen, doch ist hierfür eine hohe Modellqualität notwendig. Erst durch eine konsistente Datenqualität können die jeweiligen Leistungsphasen auf korrekte Ausführung geprüft werden. Weiterhin ist es notwendig, dass aus dem Modell 2D-Pläne abgeleitet werden können, da gewisse Arbeitsschritte, wie das Kontrollieren der Radienfolge, nach wie vor auf konventionelle Art effektiver durchgeführt werden können. Die zentrale Qualitätssicherung und die hierfür benötigten vorprogrammierten Regeln werden durch die ZBIM bereitgestellt.

5. Konzept für die Modell- und Programmentwicklung

Bisher wurde auf die allgemeine Theorie eingegangen, die relevant ist, um die weitere Vorgehensweise in der modellbasierten Unterstützung des Sicherheitsauditors verstehen zu können. Es wurde über die möglichen ACCC-Methoden gesprochen, eine Einsicht in IFC gegeben und ein Überblick über die Geschichte und Theorie der Verkehrssicherheit sowie der BIM-Anwendung und dem Sichereitsaudit der LBD gegeben. In den nächsten Abschnitten wird auf die Methodik eingegangen und die einzelnen Arbeitsschritte erläutert, die ein Modell so vorbereiten, dass es den Auditor letztlich bei seinem Sichereitsaudit durch eine Repräsentation in der virtuellen Realität möglichst effektiv unterstützen kann.

Klar zu unterscheiden sind bei der Untersuchung zwei Vorgehensweisen. Die erste verfolgt eine Lösung über die BCF-Issues, mit denen es möglich ist in VR zu interagieren. Die zweite besteht daraus, eine IFC-Datei so zu verändern, dass sie im Anschluss ein "physischesÖbjekt enthält, das als visuelle Unterstützung zur Erkennung von Defiziten dienen soll. Unabhängig von der Variante stehen immer die Kerndefizite im Fokus.

5.1 Geometrische Informationsbedarfstiefe

Durch die Anwendung von VR hat die Qualität der geometrischen Informationsebene einen besonders hohen Stellenwert. Der Bereich Geometrie und Semantik, überschneidet sich in manchen Fällen, doch ist ersteres primär durch seinen visuellen Aspekt gekennzeichnet. Zwar kann die Auswertung der semantischen Informationen durch Programme den Auditor unterstützen, doch soll die Untersuchung mittels VR einen realen Eindruck der Planung vermitteln und helfen, bessere Einschätzungen treffen zu können. So können mittels VR beispielsweise Haltesichtweiten geprüft werden, weshalb es notwendig ist, dass das Gelände um die Straße ausreichend detailliert aufgenommen wurde, um in Bereichen wie etwa von einer Kurve nachvollziehen zu können, ob das Sichtfeld frei ist. Dasselbe gilt für Einfahrtbereiche, die auf einer Mittelinsel Hinweisschilder enthalten. Diese dürfen nicht nur stilisiert als Objekte existieren, sondern müssen den realen Bedingungen entsprechen, damit nachempfunden werden kann, ob sie dem Anfahrenden ein Sichthindernis bilden oder nicht (siehe Abb. 5.1). In vielen Planungen wird die Relevanz des Bewuchses in den frühen Phasen vernachlässigt, da das Ausmaß oft nicht absehbar ist, daher wird das Projekt in einem Zeitraum von einem halben Jahr nach Projektabschluss noch einmal untersucht, um den Einfluss dieser Aspekte auf die Sicherheit untersuchen zu können. Eine solche Eventualität könnte demnach auch visuell hervorgehoben werden.

Abbildung 5.1: 1. Leitplanke versperrt eventuell Sicht. 2. Wird die Sicht von einfahrenden Pkws sowie Lkws von dem Schild behindert? 3. Senke im Boden korrekt entwässert? 4. Bewuchs in Kurve geplant? Wenn ja, versperrt er die Sicht?

Im gegebenen Modell sind die Bankette und Mulden nicht als explizite Objekte modelliert, heißt, dass diese nur visuelle Bestandteile der Böschung sind und daher semantisch nicht abgefragt werden können. Die Bankette sind teilweise über das Bankettmaterial ableitbar, welches jedoch nicht überall vorhanden ist (siehe Abb. 5.2), aber selbst durch Auslesen des Objektes für das Bankettmaterial lässt sich nicht die korrekte Kronenbreite ablesen, da wie im Bild ersichtlich die Breite des Elements durch einen abgeneigten Teil vergrößert wird. Ebenso verhält es sich mit den Mulden, welche auch nicht modelliert sind. Visuell macht es keinen Unterschied, doch kann auf diese Weise nicht abgefragt werden, ob zwischen der Mulde und der Böschung die Beziehung IfcRelConnects existiert, über die ermittelt werden könnte, ob die notwendige Mulde bei einer Einschnittböschung vorhanden ist. Dieser Punkt gehört zwar in die semantische Kategorie, aber es sei an dieser Stelle jedoch erwähnt, dass die Modelle der Pilotprojekte leider sowieso keine Beziehungen enthielten, bis auf eine Zugehörigkeit zum Default Building Storey. Dieser Umstand ist der Tatsache geschuldet, dass viele Anwendungen die Beziehung zweier Objekte, wie etwa den Kontakt, nicht in das IFC-Schema übersetzen.

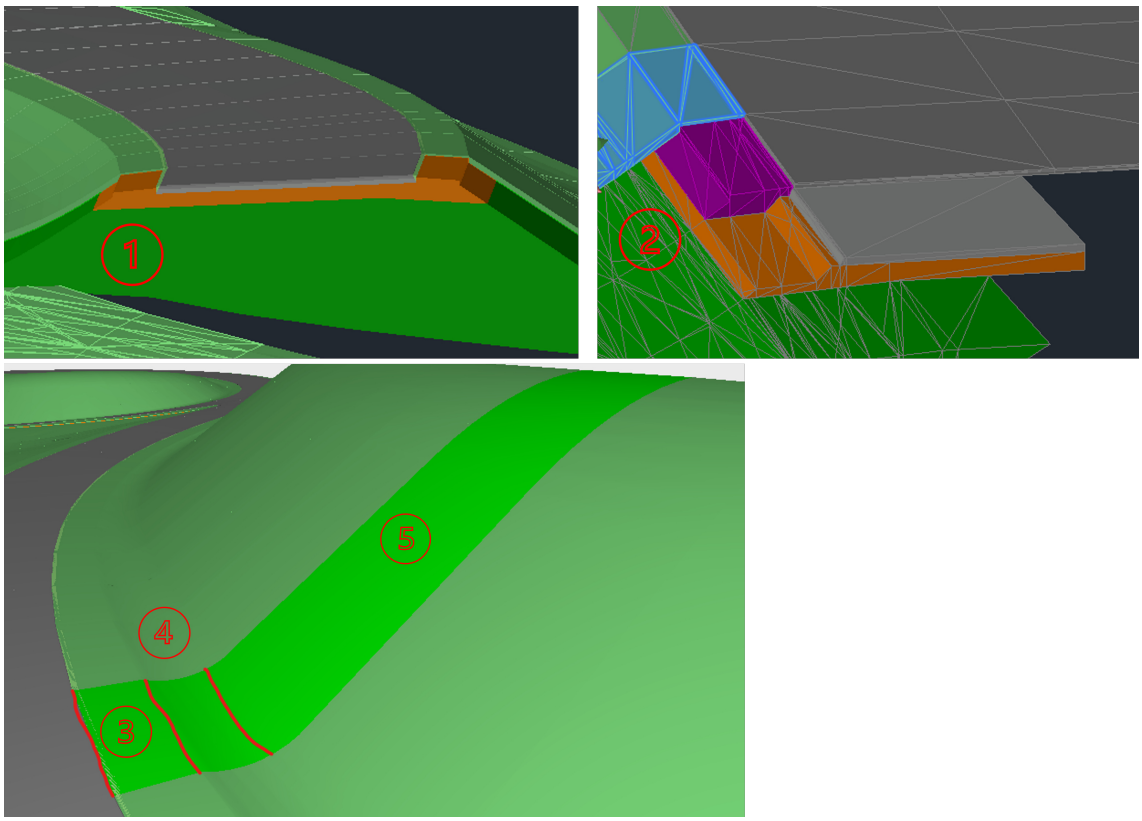


Abbildung 5.2: [oben] Querschnitt ohne (1) und mit Bankettmaterial (2) [unten] (3) Bankett, (4) Mulde, (5) Böschung

Der nächste Aspekt ist, dass im Modell die Straße alle 5 m in Querschnitte eingeteilt und meist über die gesamte Breite verläuft (siehe Abb. 5.3). Um eine korrekte Analyse der Querneigung über eine geometrische Abfrage machen zu können, muss die Fahrbahn jedoch in ihre einzelnen Fahrstreifen segmentiert werden, da die Neigung nicht immer konstant über den gesamten Querschnitt verläuft, sondern über ein Dach oder eine Verwindungskante getrennt sein kann. Auf diese Weise können die Eckpunkte der Segmente, die eine konstante Querneigung haben, abgefragt und so diese ermittelt werden. Noch besser wäre es jedoch, wenn die Information der Eckpunktkoordinaten oder die Querneigung an sich für das entsprechende Objekt in einem Eigenschaftssatz übermittelt wird, da die geometrische Abfrage rechenaufwendig ist. Am besten wäre es jedoch, wenn die Straßenelemente über IfcLinearPlacement platziert wären, wodurch mittels der Eckpunkte und der Orthogonalen zu der die Strecke repräsentierenden Kurve direkt eine Querneigung ermittelt werden könnte. Diese Option ist aber erst mit IFC 4.3 möglich, wie bereits im Kapitel 2.3 erläutert.

Ein weiterer Punkt ist, dass die rechteckigen Segmente nicht nur aus vier Eckpunkten, sondern aus einer alternierenden Anzahl an Außenpunkten jeweils für die obere- und untere Ebene des Objekts getrennt bestehen (siehe Abb. 5.3). Es ist möglich diese Punkte abzufragen, die obere Ebene zu ermitteln und die Koordinaten der äußersten und obersten zwei Eckpunkte in einer Querschnittsebene zu ermitteln, um hiermit eine Querneigung zu berechnen, wie später gezeigt wird. Die Fehleranfälligkeit ist hierbei jedoch sehr hoch und das Ergebnis für einige Segmente nicht korrekt, da diese nicht rechteckig sind, weshalb,

wie bereits erwähnt, eine alphanumerische Angabe der Querneigung oder ein repräsentativer Vektor von Vorteil wäre. Das entwickelte Programm für die Querneigungsprüfung ist ein Beispiel für eine Überschneidung der geometrischen und semantischen Information, der sich nicht ausschließlich auf die Identifikation des Elements bezieht, sondern durch die Abhängigkeit der Querneigung vom Radius definiert wird. So enthält das Modell für jedes Segment die Information des Radius in einem Eigenschaftssatz, weshalb eine Querneigungsanalyse entwickelt werden konnte, die geometrisch die Neigung ermittelt und mit dem alphanumerisch gegebenen Radius abgleicht.

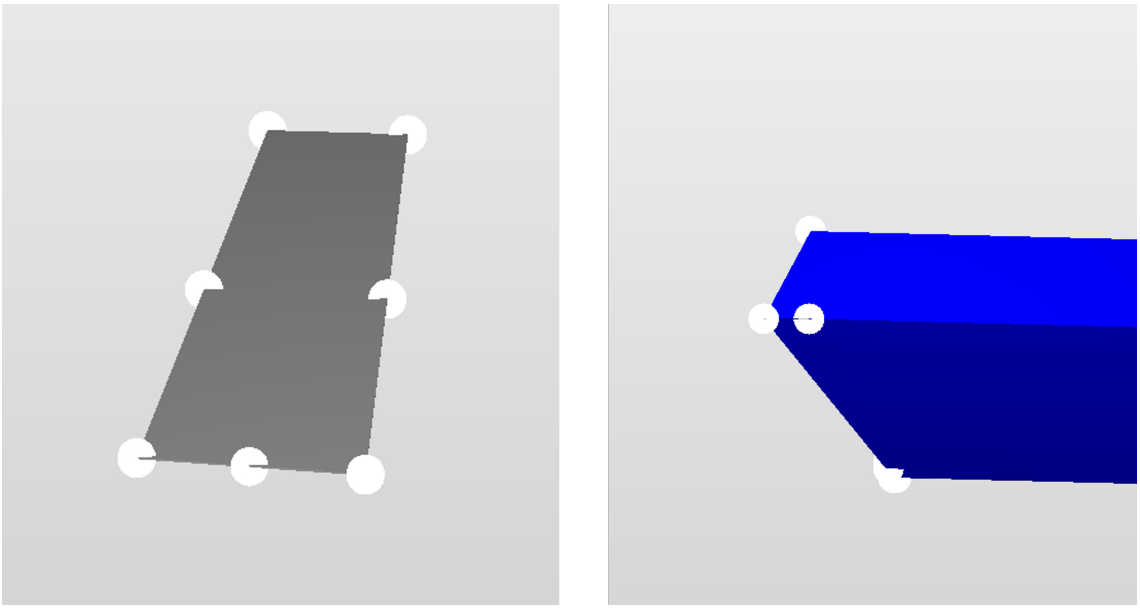


Abbildung 5.3: Die Polygonpunkte der Ashpaltdeckschicht

Die genannten Punkte sind ein Ansatz für die Entwicklung eines geometrischen Modellstandards. So kann die LBD diese nutzen, um zu definieren, wie das Modell für welche Leistungsphase zu entwerfen ist. Zusammengefasst können also diese Objekte genannt werden, die explizit modelliert werden müssen:

- Bankettkrone getrennt von Bankett oder Angabe von Kronenbreite
- Mulde explizit
- Bankett explizit
- Böschung explizit
- oder Böschung besteht aus:
    - Hang
    - Mulde
    - Bankett

Gleichzeitig dient diese Liste zusammen mit der später aufgeführten Tabelle für die alphanumerische Informationsbedarfstiefe als eine Checkliste, um das gegebene Modell

auf seine Qualität bewerten zu können. Mit der wachsenden Erfahrung in den weiteren Projekten muss diese weiter ausgearbeitet und festgehalten werden, auf welche Objekte in einer visuellen Darstellung nicht verzichtet werden kann und welche eventuell durch eine semantische Information ersetzt werden können.

5.2 Alphanumerische Informationsbedarfstiefe

Um den Auditor effektiv mittels ACCC unterstützen zu können ist ein klar strukturierter alphanumerischer Informationsgehalt notwendig. Große Firmen etablieren daher eigene Standards, um ihre Prozesse vereinheitlichen und somit beschleunigen zu können. Die buildingSmart Plattform bietet zwar eine umfangreiche Palette an Entitäten und Typen an, mit denen ein Modell bereits sehr gut definiert werden kann, doch ist nicht alles abgedeckt. Wie bereits im Kapitel 2.3 erörtert, ist der Bereich Infrastruktur noch in der Entwicklung, weshalb für viele Bestandteile vorerst eigene Definitionen festgelegt werden müssen. Abgesehen davon ist es in vielen Fällen sinnvoll, zusätzliche firmeneigene Attributslisten zu entwickeln, um sich an interne Standards anzupassen. Modellieren die Planer entsprechend solcher Vorgaben, ist die Auswertung aufgrund festgelegter Begriffe um ein vielfaches erleichtert. Für die Definition eines adäquaten Begriffskonzeptes fließt das Verständnis des IFC-Schemas mit den Informationsanforderungen eines Sicherheitsaudits zusammen. Da es im BIM-Leitfaden der LBD noch keine Vorgaben bezüglich eines Modellierungsstandards gibt, wird sich an den Defizitlisten und den Vorgaben der RAL orientiert. Tabelle 5.1 ist das Ergebnis der Anpassung der Modelle der Pilotprojekte.

Tabelle 5.1: Alphanumerische Modellinhalte

IFC Entität und zugehörige Eigenschaftssätze												
IfcRoadPart	x	x									x	x
IfcKerb	x		x									
IfcReinforcedSoil	x			x								
IfcRailing	x				x							
IfcRoadPartTypeEnum.ROUNDABOUT	x					x						
IfcPavement	x						x					
IfcRoadPartTypeEnum.SHOULDER	x							x				
IfcRailingTypeEnum.GUARDRAIL	x				x					x		
IfcRoadPartTypeEnum.TRAFFICISLAND	x											
IfcRoadPartTypeEnum.ROADMARKING	x								x			
IfcEarthworksFillTypeEnum.EMBANKMENT	x											
IfcFooting	x											
Eigenschaft und zugehöriger Datentyp	LBD_IfcIdentifier	LBD_RoadProperties	Pset_KerbCommon	LBD_BöschungsAttribute	Pset_RailingCommon	LBD_RoundaboutProperties	Pset_PavementCommon	LBD_BankettAttribute	Pset_MarkingLinesCommon	LBD_FahrzeugrückhaltesystemeAttribute	Pset_RoadSymbolsCommon	Pset_RoadDesignCriteriaCommon
IfcEntität (IfcLabel)	x											
Referenz (IfcLabel)	x											
Längsneigung (IfcRatioMeasure)		x						x				
Böschungsneigung (IfcRatioMeasure)				x								
IstDamm (IfcBoolean)				x								
IstEinschnitt (IfcBoolean)				x								
HatMulde (IfcBoolean)				x				x				
HasDrainage (IfcBoolean)				x								
Entwurfsklasse (IfcLabel)								x				
Radius (IfcPositiveRatioMeasure)		x										
HatVerwindungskante (IfcBoolean)		x										
Verkehrsführung (Einfädelung/Ausfädelung)		x										
Verkehrswegbezeichnung (IfcLabel)		x						x	x			
Fahrbahnbelag (IfcLabel)		x										
Crossfall (IfcRatioMeasure)								x				x
Neigungsrichtung (IfcLabel)								x				
DesignSpeed (IfcLinearVelocityMeasure)												x
DesignTrafficVolume (IfcCountMeasure)												x
DesignVehicleClass (IfcLabel)												x
LaneWidth (IfcPositiveLengthMeasure)												x
NumberOfThroughLanes (IfcCountMeasure)												x
RoadDesignClass (IfcLabel)												x
VerkehrsraumHöheKleiner45 (IfcBoolean)		x										

Seitenstreifenbreite (IfcPositiveLengthMeasure)		x										
Von_Bau_km (IfcLabel)	x											
Bis_Bau_km (IfcLabel)	x											
Reference (IfcLabel)					x		x			x		
Height (IfcPositiveLengthMeasure)					x	x				x		
Upstand (IfcPositiveLengthMeasure)			x									
CombinedKerbGutter (Boolean)			x									
StructuralSlope (IfcRatioMeasure)							x					
NominalWidth (IfcPositiveLengthMeasure)							x					
Kronenbreite (IfcPositiveLengthMeasure)								x				
DashedLine (IfcBoolean)									x			
Regelquerschnitt (IfcLabel)		x										
Entlang_Verkehrsweg (IfcLabel)				x								
Text (IfcText)											x	
TypeDesignation (IfcLabel)											x	

5.3 Modellentwurf

5.3.1 Testentwurf einer Trasse

Um eine realistische Bewertung der Tests und des Arbeitsablaufs zu ermöglichen, wird ein idealisiertes Modell entworfen, das als Vergleichsmaßstab für die Pilotprojekte der LBD dient. Hierfür wurde Autodesk AutoCAD Civil 3D verwendet, ein mächtiges Tool für den Modellentwurf im Bereich Tiefbau und Infrastruktur, das einen IFC- Im- und Export ermöglicht. Um nachvollziehen zu können, wie die geometrischen und alphanumerischen Informationen eingepflegt werden und wie diese letztendlich nach dem Export im IFC-Schema vorhanden sind, wurde das Teilmodell "VK_300_MOD_08_220211_B_Knoten_G"des Pilotprojektes "B299, dreistufiger Ausbau Geisenhausen"(Im Folgenden nur noch B299) so modifiziert, dass es alle nötigen Informationen enthält, um ein für das Sicherheitsaudit idealisierten Standard zu repräsentieren. Zusätzlich wurden Ausstattungselemente eingefügt, die als Testgegenstände dienen.

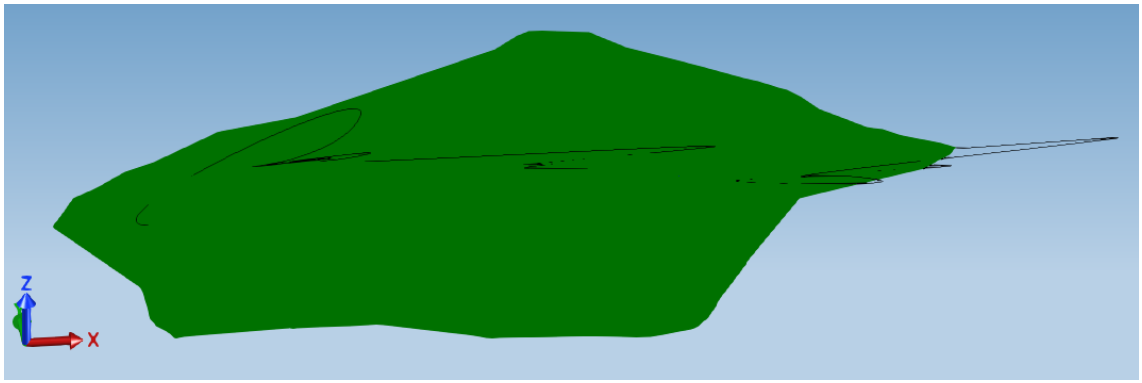


Abbildung 5.4: Inkorrekte Darstellung von IfcAlignment in Open IFC Viewer nach Export aus Civil3D

Vor dem Entwurf des idealisierten Teilmodells wurde jedoch zuerst ein eigenständiges Modell einer Landstraße entworfen, um ein generelles Verständnis für die Modellentwicklung zu erhalten und nachvollziehen zu können, welche Daten bei dem Export in IFC erhalten beziehungsweise verloren gehen. Besonders interessant war es zu untersuchen, ob und wie der Export von IfcAlignment funktioniert, da diese Klasse in IFC 4.1 eingeführt wurde und Informationen enthält, die die Straße in einen gesamtheitlichen Kontext setzt. Die Modelle der Pilotprojekte wurden nämlich in iTwo Civil modelliert und in IFC 4.0 übergeben, welches die Straße nur in ihren Subentitäten, die über IfcLocalPlacement platziert wurden, übermittelt und daher keine Ausrichtungsinformationen beinhaltet.

Für den Entwurf wurden über Triangulationspunkte eines Beispielprojektes eine Oberfläche generiert, in die eine Ausrichtungslinie (Alignment) gezeichnet wurde. Basierend auf dieser Linie wurde unter Beachtung der Grenzwerte für Kuppenhalb- und Wannenhalbmesser ein Höhenprofil erzeugt. Daraufhin wurde ein zweistreifiger Regelquerschnitt 11,5+ für die EKL 2 modelliert und auf die Ausrichtungslinie gelegt. Civil3D 2023 unterstützt IFC 4.1 und kann daher IfcAlignment exportieren. Bei Nutzung der Version 2018 oder 19 von Civil3D

kann auch das Plug-in genutzt werden, das von Felix Rampf in seiner Bachelorthesis entwickelt wurde (RAMPF, 2017). Obwohl die Ausrichtungsentität nach dem Export im IFC-Schema vorhanden ist, lässt sich diese nicht in Solibri nutzen und wird im Open IFC Viewer nicht korrekt dargestellt (siehe Abb. 5.4). Dementsprechend war es nicht möglich zu untersuchen, wie mittels der Solibri-API auf die Ausrichtungsinformation zugegriffen werden kann, um es für eine weitere Anwendung zu nutzen.

5.3.2 Behebung von Ex- und Importkonflikten

Generell erzeugte der Ex- und Import von IFC-Schemas Konflikte zwischen den unterschiedlichen Softwareanwendungen, deren Ursachen meist nicht direkt nachvollziehbar waren. So war das Modell des Pilotprojektes B299 nach dem Import in Civil3D nicht sichtbar. Dieses Problem wurde durch einen Umweg über Revit gelöst. Auch hier war das importierte Modell im IFC 4 Schema anfangs nicht sichtbar, jedoch konnte dies durch eine Verknüpfung zur Ursprungsdatei gelöst werden (siehe Abb. 5.5). Um die Dateigröße der IFC-Schemas zu verringern und die Leistung zu steigern, wurde zuvor der Solibri IFC Optimizer genutzt. Im Falle des Teilmodells der B299 konnte so die Dateigröße um 32% reduziert werden. Nachdem das Modell in Revit mit seinem IFC-Schema verknüpft und somit sichtbar wurde, musste die Verknüpfung an das Modell gebunden werden (siehe Abbildung 5.5). Um ein möglichst schlankes Modell zu erhalten wurden zusätzlich alle für die Prüfung nicht relevanten Elemente, wie abzutragende Erde, aus dem Modell gelöscht. Daraufhin kann der Export aus Revit erfolgen. Untersucht man jedoch danach die exportierte Datei fällt auf, dass alle ursprünglichen Eigenschaftssätze verschwunden sind und somit jede Möglichkeit, die einzelnen Elemente auf semantischer Ebene voneinander zu unterscheiden. Daher muss an dieser Stelle auf die korrekten Exporteinstellungen geachtet werden muss.

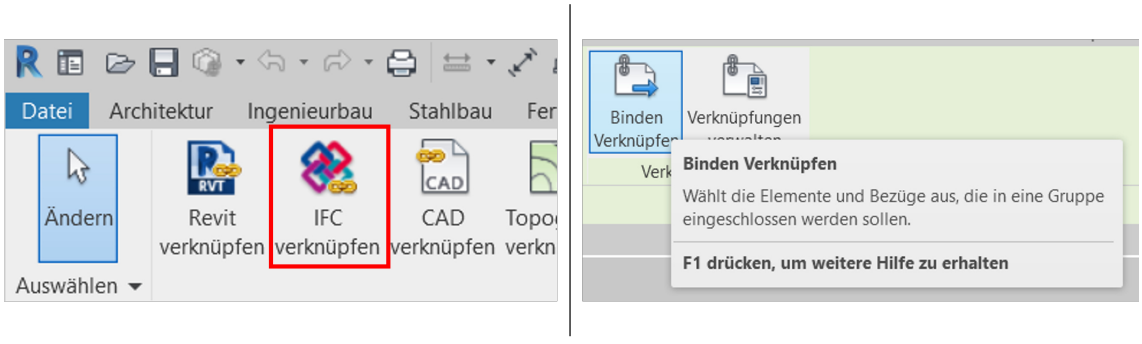


Abbildung 5.5: [links] Verknüpfungen, [rechts] Binden Verknüpfen

Für den IFC-Export bietet Revit zwei Methoden an die Eigenschaftssätze in IFC zu übermitteln. Die erste ist über die Erstellung einer Bauteilliste. Hierbei können Kategorien wie Wände, Achsen, Bewährung und dergleichen mit den neuen Eigenschaftssätzen belegt werden (siehe 5.6). So kann beispielsweise eine Bauteilliste unter Ansicht/Erstellen mit dem Namen Älle_Datenërstellt und dieser alle relevanten Informationen zugeordnet werden. Daraufhin muss unter den Exporteinstellungen "Bauteillisten als Eigenschaftssätze

exportierenängeklickt und IFC 2x3 Reference View 2.0 ausgewählt werden und der Export kann gestartet werden.

Die zweite Methode ist über eine Mapping-Tabelle. Hier werden in einer Textdatei 5.6, dessen Pfad in den IFC-Exporteinstellungen angegeben werden kann, Eigenschaftssätze individuell definiert und den entsprechenden IFC-Entitäten zugeordnet. Dieses Verfahren wird als Mapping bezeichnet und ist ausschließlich auf die IFC-Klassen beschränkt. Da das Modell nur IfcBuildingElementProxy Klassen enthält, ist das Mapping in diesem Fall relativ simpel, da nur ein Eigenschaftssatz für die besagte Klasse definiert werden muss. Zusätzlich ist die zweite Methode sehr effizient, die eine präzise Attribuierung ermöglicht und redundante Informationen vermeidet, weshalb diese Variante gewählt wurde. Es sei angemerkt, dass an dieser Stelle aufgefallen ist, dass die Teilmodelle unterschiedliche IFC-Schemas aufwiesen, die zwischen 2x3 und 4 variierten. Von einer solchen Inkonsistenz wird abgeraten.

Hier stellt sich eventuell die Frage, warum überhaupt in Civil3D statt in Revit weitergearbeitet wird. Die Antwort liegt in den unterschiedlichen Bearbeitungsmöglichkeiten. Wie bereits erklärt bietet Revit zwar eine Möglichkeit der Attribuierung an, doch ist diese auf tabellarisch eingeteilte reviteigene Elemente beziehungsweise IFC-Klassen beschränkt, auf die ausschließlich semantisch verwiesen werden kann, wohingegen Civil3D mit dem kommerziellen Plugin PSED documentation von Camilion eine flexible interaktive Elementauswahl ermöglicht. So können die unterschiedlichen Elemente einzeln angeklickt und über eine Toolbar einer Gruppe von Eigenschaftssätzen zugeordnet werden. Abgesehen davon beinhaltet Civil3D Werkzeuge, die spezifisch für den Infrastrukturbau ausgelegt sind, was in späteren Schritten nützlich wird.

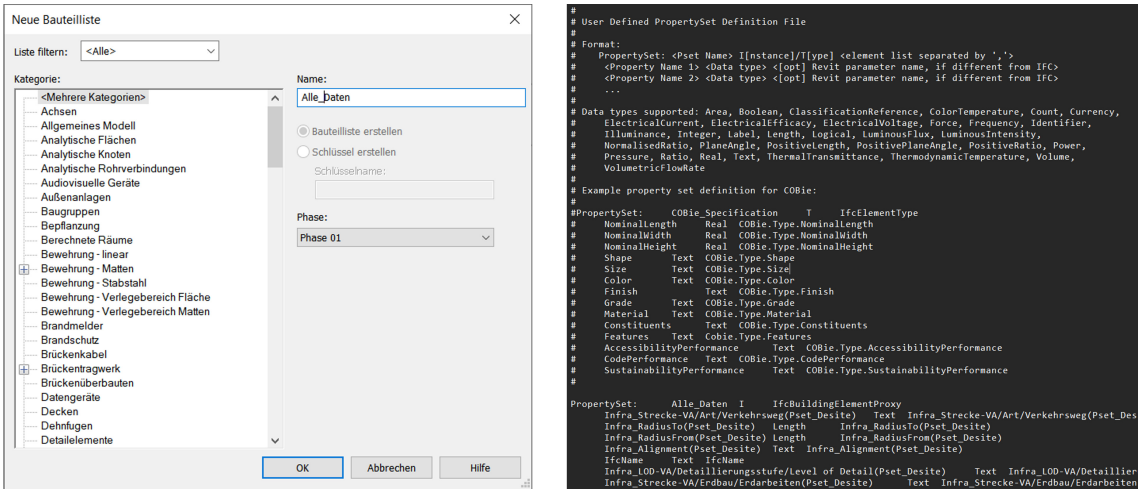


Abbildung 5.6: [links] Erstellen einer Bauteilliste, [rechts] Mapping-Tabelle im Editor

Nachdem das Modell in Civil3D importiert wurde, kann es mit den Informationen entsprechend der Tabelle 5.1 angereichert werden. Zusätzlich werden Bäume und Schilder eingesetzt und einige Defizite eingearbeitet, um als eine Referenz für die Checks zu dienen.

Wird das Modell nach dem Export aus Civil3D und Abschluss der Bearbeitung in Solibri geöffnet, fallen neben den visuellen Fehlern, auf die im nächsten Abschnitt eingegangen wird, zusätzlich auf, dass zwar die Daten der in Civil3D definierten Eigenschaftssätze übertragen wurden, jedoch nicht die mittels der Mapping-Tabelle aus Revit übertragenen unter dem Namen "Daten_Aus_RevitExport". Wird das IFC-Schema untersucht, lässt sich feststellen, dass durch den Export, die Daten aller IfcPropertySingleValues verschwunden sind, mit Ausnahme von (1) Ïnfra_LOD-VA_Detaillierungsstufe_Level of Detail(Pset_Desite)ünd (2) Ïnfra_RadiusTo(Pset_Desite)ünd Ïnfra_RadiusFrom(Pset_Desite)". Bei diesen sind zwar die Werte enthalten, jedoch wurde bei (1) der Nennwert (Nominal Value) von IfcText in IfcLengthMeasure verändert und in (2) von IfcLengthMeasure in IfcAmountOfSubstanceMeasure, weshalb bei ersterem die Leistungsphase als 300 m angegeben wird und bei letzterem der Datensatz überhaupt nicht zu sehen ist. Warum Civil3D diese Veränderungen vornimmt ist nicht ersichtlich, doch da der Radius für die spätere Anwendung benötigt wird, lässt sich das Problem mit dem Windows Editor durch Suchen und Ersetzen der fehlerhaften Begriffe leicht beheben.

Das Modell für das Projekt St2101 Ertüchtigung Antoniberg-Tunnel bestand aus vielen Teilmodellen, die erst zusammengefügt ein für das Sicherheitsaudit sinnvolles Gesamtmodell erzeugten. Erneut musste eines der Teilmodelle durch einen Revitexport im Mappingverfahren in das Schema IFC 2x3 verändert werden, um mit den anderen kompatibel zu sein. Daraufhin wurde SimpleBIM verwendet, um die Modelle zu einem Gesamtmodell zu mergen. Die Teilmodelle waren jedoch für die Bearbeitung zu groß und ermöglichten keine flüssige Arbeitsmethode, weshalb die Untersuchung auf das Modell des B299-Projekts beschränkt wurde.

5.4 Behebung visueller Darstellungsfehler

Wird das bearbeitete Modell in Solibri geöffnet, fallen als erstes einige unregelmäßig schwarz gefärbte Objekte auf (siehe Abb. 5.7). Verglichen zur ursprünglichen aus Squirrel heruntergeladenen Datei sind mehrere Stellen der Böschung statt grün in Schwarz dargestellt. Wird das zugrundeliegende IFC-Schema nach den entsprechenden GUIDs durchsucht, werden mehrere Objekte unter jeweils einer ID gefunden. Beim Export aus Revit sind die ursprünglichen IDs nicht übernommen und die Objekte mit neuen belegt worden, die sich nach einer gewissen Anzahl vergebener GUIDs wiederholen. Weiterhin ist die Anzahl an IfcBuildingElementProxy-Elementen mehr als doppelt so groß wie zuvor und viele der Elemente sind im Modell nicht auffindbar. Obwohl das Vorhandensein mehrerer Objekte unter der gleichen GUID nicht sinnvoll ist und es auch keine Objekte geben sollte, die im Modell nicht auffindbar sind, war dies nicht das Problem hinter der fehlerhaften Darstellung der Farben, da dies erst nach dem Export aus Civil3D eintritt, die genannten Fehler im IFC-Schema, aber bereits nach dem Export aus Revit vorhanden sind.

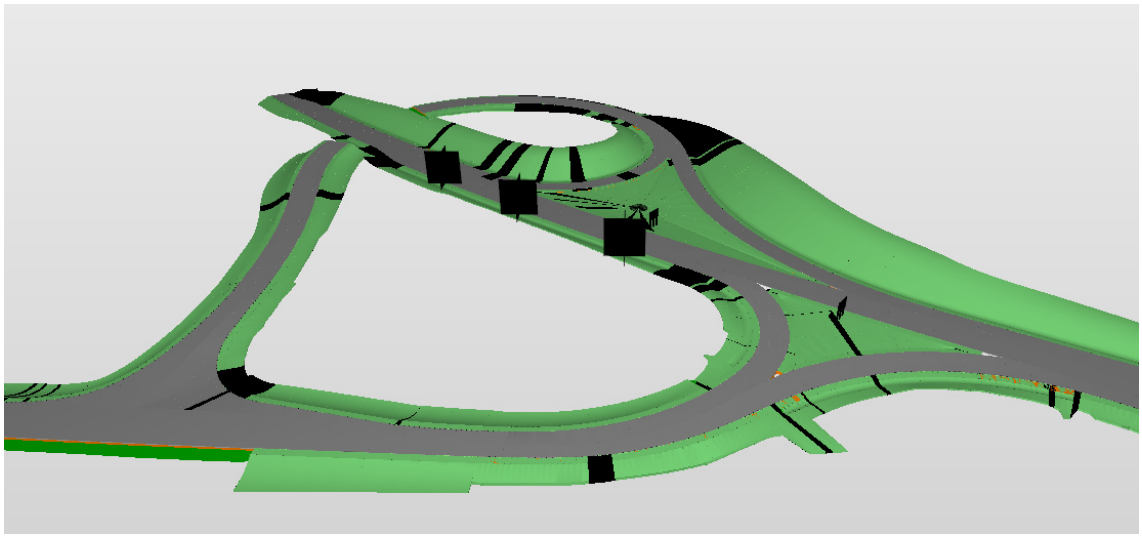


Abbildung 5.7: Modell mit Darstellungsfehlern

In den Attributen von IfcShapeRepresentation ist das Schlagwort "MappedRepresentation"(siehe Abb. 5.8) gegeben, das angibt, dass die Darstellung des Elements über eine Representation-Map gesteuert wird. In IfcStyledItem wird dementsprechend auf die Zeile 45 verwiesen, in der das Objekt einem Surface Style zugeordnet wird, das auf eine Farbdefinition im RGB-Spektrum verweist. Vorteil dieser Methode ist, dass beliebig viele Objekte einer Darstellungsweise zugeordnet werden können. Zusätzlich sind alle Objekte einem Layer in IfcPresentationLayerWithStyle zugeordnet. Dieser kommt jedoch nicht zur Anwendung, da IfcSurfaceStyle einen höheren Rang in der Darstellungsdefinition hat und somit diesen überschreibt. Das gegebene Beispiel verweist auf einen RGB-Raum, der die Böschung in Grün darstellt. Die Elemente, die in Schwarz dargestellt sind, verweisen auf einen RGB-Raum, der mit IFCCOLOURRGB($,0.0,0.0,0.0) definiert ist.

Werden die IFC-Schemas vor und nach dem Export aus Civil3D verglichen, zeigt sich eine deutliche Veränderung. So wird in der ursprünglichen Datei in IfcShapeRepresentation nicht auf das Schlagwort "MappedRepresentation", sondern auf SSurfaceModel"verwiesen. Weiterhin wird in IfcSurfaceStyle die Seite für die Darstellung des Stils als "BOTH", also für beide Seiten definiert, statt nur für die äußere Seite, wie in der Post-Export-Datei mit "POSITIVE"festgelegt ist. Außerdem wird in der aktuelleren Datei von IfcSurfaceStyle auf IfcSurfaceStyleShading verwiesen, das eine viel geringere Informationstiefe ermöglicht als IfcSurfaceStyleRendering, in dem Werte wie Reflektionsfaktor und Lichtstreuung bestimmt werden können. Letztendlich wird selbst nach der Untersuchung der Dateien nicht eindeutig klar, warum der Export die IFC-Schemas auf diese Weise verändert. Es ist zu vermuten, dass es einerseits an der notwendigen Änderung des IFC-Schemas von IFC 4 in 2x3 und an den unterschiedlichen Exportverfahren zwischen Revit und Civil3D liegt. Mit den gewonnenen Erkenntissen, war es jedoch möglich, die schwarzen Flecken zu entfernen, indem die fehlerhaften Objekte dem gleichen RGB-Wert wie die übrigen Elemente zugewiesen wurden.

Abbildung 5.8: Beziehung eines Elements zu seiner visuellen Darstellung

5.5 Solibri API

Unabhängig davon, ob der Konflikt mittels BCF oder über ein erzeugtes Hinweisobjekt in VR übermittelt wird, wird für die Untersuchung der Modelle Solibri verwendet. Über seine Schnittstelle zu Java können zusätzlich zu den vorhandenen Regeln weitere entwickelt werden, die das Modell auf entsprechende Konflikte untersuchen. Die Synchronisation mit einem BCF-Server wie BIMCollab ermöglicht es so die Konflikte zu teilen und in anderen Anwendungen (Beispiel: VRex) einzusehen. Über die API ist weiterhin auch die Entwicklung von Ansichten möglich. So werden die Fenster und die darin enthaltenen Interfaces bezeichnet, mit denen unterschiedliche Funktionen wie Filtern, Klassifizieren oder auswerten genutzt werden können. Wird die Tatsache vernachlässigt, dass die Identifizierung eines Objektes immer über eine semantische Abfrage geschieht, können die Konformitätsprüfungen in drei Kategorien eingeteilt werden.

1. Prüfung des Modells über geometrische Aspekte
2. Prüfung des Modells über semantische Aspekte
3. Eine Kombination aus beidem

Die Überprüfung des Seitenraums mittels des Abstands zwischen der Straße und den angrenzenden Objekten fällt demnach unter die erste Kategorie, ebenso die Regeln zur Überprüfung der Bankettbreite und dem Radwegende. Die zweite Kategorie ist die Abfrage nach bestimmten Werten. Diese hängt ausschließlich von der alphanumerischen Datenqualität ab und bezieht sich nicht auf das geometrische Modell. Unter diese Variante fallen die Regeln zur Überprüfung der Bankettentwässerung, der Böschungsneigung und die semantische Querneigungsprüfung. Unter den letzten Punkt fällt die geometrische

Querneigung, da hier die Querneigung geometrisch ermittelt, der Radius jedoch semantisch abgefragt wird. Im Folgenden werden ausgewählte Programme vorgestellt, um das Prinzip darzustellen und ein Verständnis für ihre Funktionsweise zu vermitteln. Angemerkt sei, dass der Fokus in diesem Kapitel auf dem Ablauf des Programms liegt. Die praktische Durchführung der Überprüfungen wird im darauffolgenden Kapitel 6 erläutert.

5.5.1 Regeln mit Konfliktrückgabe in Solibri

Programm: Unsicherer Seitenraum

Eine anschauliche Regel für die geometrische Abfrage des Modells ist die Untersuchung des Fahrbahnseitenraums. Hierfür werden in Solibri über zwei separate Filter die Objekte definiert. Der erste enthält Komponenten, die auf ihren Abstand geprüft werden sollen und der zweite die Straße, auf dessen Seitenrand sich der Abstand bezieht. Ersterer wird Komponente für Komponente durchlaufen (component) und jeweils der Abstand zwischen den beiden geprüft, falls vorhanden. Ist der Wert geringer als der angegebene Grenzwert, wird das Element zurückgegeben und dem Konflikt ein Name und eine Beschreibung zugeordnet (siehe Anhang A, Code A.1).

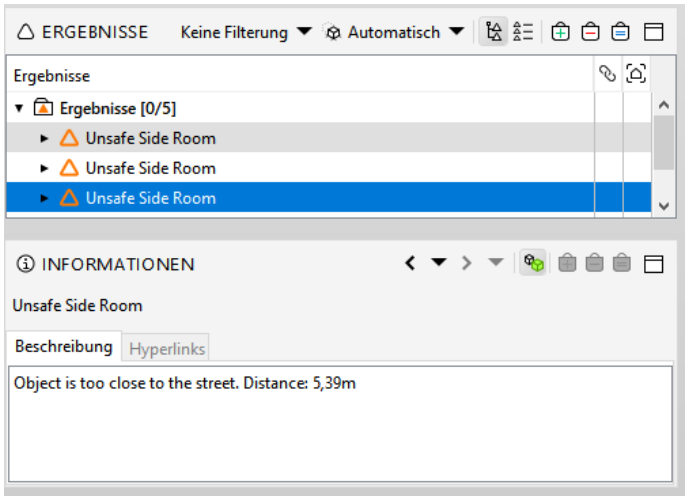


Abbildung 5.9: In Solibri wiedergegebene Informationen

In Solibri wird nun durch Klicken auf den Konflikt das Objekt farbig hervorgehoben und in einem weiteren Fenster Informationen zum Sicherheitsdefizit angezeigt, die zuvor in der Beschreibung definiert wurden (Abb. 5.9).

Programm: Geometrische Querneigung

Die Überprüfung der Querneigung in Relation zum gegebenen Radius bildet, wie bereits erläutert, eine Regel, die eine geometrische Abfrage mit einer semantischen kombiniert. Hierbei werden die Polygonpunkte der obersten Straßenschicht ausgelesen und so gefiltert, dass die zwei äußersten und obersten Punkte genutzt werden können, um über sie

eine Querneigung zu berechnen. Im Folgenden werden die Kernaspekte des Programms erläutert.

Im ersten Schritt werden die Triangulationspunkte der eingelesenen Komponente ermittelt. Diese bilden alle Koordinatenpunkte des Volumenkörpers. Teilweise kommen diese mehrfach vor, weshalb sie in ein Set geladen werden, das die redundanten Elemente entfernt.

```
40        TriangleMesh asphalt = component.getTriangleMesh();
          Collection<Vector3d> asphaltVertices = asphalt.toPointCollection();
42        Set<Vector3d> asphaltVerticesSet = new HashSet<>(asphaltVertices);
          Collection<Vector3d> asphaltVerticesNoDuplicates = new HashSet<>(
    asphaltVerticesSet);
```

Danach werden alle Punkte durchlaufen und gruppiert. Hierfür werden diese, die sich zueinander in einem Abstand von weniger als 0,10 m befinden, in ein Set geladen (siehe Abb. 5.3). Da die Punktgruppen jeweils in unterschiedlicher Reihenfolge erstellt werden und Vektoren sich nicht auf simple Weise sortieren lassen, kann nicht über eine Collection.contains()-Abfrage geprüft werden, ob die jeweilige Punktgruppe bereits enthalten ist, weshalb die Überprüfung auf Duplikate über eine doppelte Collection.containsAll()-Abfrage erfolgen muss.

```
    List<Collection<Vector3d>> pointsList = new ArrayList<>();
46      boolean same = false;
        for (Vector3d pointSource : asphaltVerticesNoDuplicates) {
48          Set<Vector3d> points = new HashSet<>();
            for (Vector3d pointTarget : asphaltVerticesNoDuplicates) {
50              same = false;
                if (pointSource.distance(pointTarget) < 0.10)
52                  points.add(pointTarget);
            }
54          for (Collection<Vector3d> vector3ds : pointsList) {
                if (vector3ds.containsAll(points) && points.containsAll(
    vector3ds)) {
56                  same = true;
                    break;
58              }
            }
60          if(!same)pointsList.add(points);
        }
```

Sind alle Punktgruppen erstellt und nur einmal vorhanden, wird innerhalb dieser Gruppen untersucht, welcher der Punkte den größten Z-Wert besitzt, um so das Polygon der obersten Ebene des Straßenelements auszufiltern. Zu beachten ist, dass einige Punkte nur Millimeter nebeneinander liegen und diese daher erstens in der Visualisierung (siehe Kapitel 5.1, Abb. 5.3) nicht zu erkennen sind und zweitens darauf geachtet werden muss, dass der Z-Wert über größer-gleich gesucht wird, da diese Punkte die gleiche Höhe haben.

Ist der höchste Punkt gefunden, beziehungsweise einer der höchsten Punkte, wird mit dem break-Kommando aus der Schleife gesprungen und direkt die nächste Punktgruppe untersucht.

```
        Collection<Vector3d> pointZ = new ArrayList<>();
64      for (Collection<Vector3d> pointsCollect : pointsList){
            OUTER: for (Vector3d pointSource : pointsCollect){
66              int biggerCount = 0;
                 for (Vector3d pointTarget : pointsCollect) {
68                  if (!pointSource.equals(pointTarget) && pointSource.getZ()
   >= pointTarget.getZ())
                        biggerCount++;
70
                    if (pointsCollect.size() == 1)
72                      biggerCount++;

74                  if (biggerCount == pointsCollect.size() - 1){
                        pointZ.add(pointSource);
76                      break OUTER;
                    }
78              }
            }
80      }
```

Um die äußeren Punkte in eine umlaufende Reihenfolge zu ordnen, wird die ConvexHull-Funktion verwendet, die Teil der Solibribibliothek ist. Diese findet die kleinste Anzahl an Punkten, die nötig ist, um alle Punkte zu umschließen. Dies ist nützlich, da so eventuell leicht nach innen versetzte Punkte ignoriert werden. Für die Anwendung des Algorithmus müssen die dreidimensionalen Punkte in zweidimensionale umgewandelt werden. Der ConvexHull-Algorithmus findet nur Anwendung, falls es mehr als zwei Polygonpunkte gibt. Ist dies nicht der Fall, wurden die Eckpunkte des Straßensegments falsch aufgenommen und das Ergebnis wird leer zurückgegeben. Um die Eckpunkte und die Referenzpunkte für die Querneigung, sowie die Neigung selbst zu ermitteln wurden drei Funktionen geschrieben, die im nächsten Abschnitt erläutert werden.

```
82      MPolygon2d polygon2d;
        List<Vector2d> polygonVec = new ArrayList<>();
84
        for(Vector3d point3d : pointZ) polygonVec.add(point3d.to2dVector());
86
        polygon2d = MPolygon2d.create(polygonVec);
88      Collection<Vector3d>cornerPoints;

90      if (pointZ.size() > 2) {
            MPolygon2d convexHull = ConvexHull.of(polygon2d);
92          List<Vector2d> convexList = convexHull.getVertices();

94                  cornerPoints = getCornerPoints(convexList,pointZ);
        }
96      else cornerPoints = pointZ;
```

```
98          if (cornerPoints.size() < 2)
                return results;
100
            List<Vector3d> twoReferencePoints = getTwoReferenceCorners(cornerPoints
    );
```

Die erste Funktion (siehe Anhang A, Code A.2) dient der Erkennung von Eckpunkten (getCornerPoints()) und liest die in einer umlaufenden Reihenfolge sortierten Polygonpunkte eines Rechtecks ein und überprüft die Vektorenrichtung der Verbingungsstrecken zwischen den Punkten. Genauer beschrieben, prüft es in einer Schleife, ob der kleinste Winkel (siehe Abb. 5.10) zwischen den beiden zu prüfenden Vektoren größer als 15 Grad ist. Ist dies der Fall, so ist der erste der zwei Koordinaten, die den aktuellen Richtungsvektor bilden, ein Eckpunkt und wird der Liste der Eckpunkte hinzugefügt. Sind alle Punkte durchlaufen, vergleicht das Programm die x- und y-Werte der zweidimensionalen Punkte mit denen der dreidimensionalen Punkte, die die oberste Ebene der Oberfläche bilden und ordnet sie diesen zu, um die entsprechenden 3D-Punkte auszugeben.

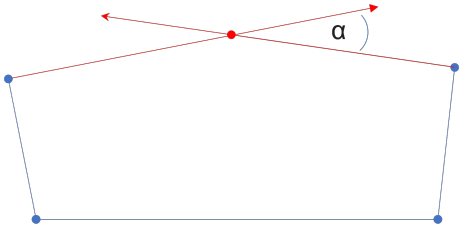


Abbildung 5.10: Polygonpunkte eines Rechtecks und der kleinste Winkel Alpha

Die zweite Funktion getTwoReferenceCorners() (siehe Code 5.1) ermittelt aus den Eckpunkten zwei Referenzpunkte, über die die Querneigung berechnet werden kann. Hierzu wird davon ausgegangen, dass die Rechtecksegmente mit ihrer langen Seite immer quer zur Straße liegen. Ist die längste Strecke ermittelt, werden die beiden Endpunkte in eine Liste geladen und zurückgegeben. An dieser Stelle ist es interessant zu erwähnen, dass die Ermittlung der Referenzlinie für die Querneigung, deutlich simpler wäre, wenn die IfcAlignment-Entität vorhanden wäre, da so diese senkrecht zur Laufrichtung dieser Linie verlaufen müsste.

```
224     List<Vector3d> getTwoReferenceCorners(Collection<Vector3d>cornerPoints){

226         List<Vector3d>cornerPointsList = new ArrayList<>(cornerPoints);
            cornerPointsList.add(cornerPointsList.get(0));
228
            List<Vector3d>twoReferenceCorners = new ArrayList<>();
230
            double distance = 0.0;
232         for (int i=0; i < cornerPointsList.size() - 1; i++){
                if (cornerPointsList.get(i+1).distance(cornerPointsList.get(i)) >
    distance){
```

```
234                 twoReferenceCorners.clear();
                    twoReferenceCorners.add(cornerPointsList.get(i));
236                 twoReferenceCorners.add(cornerPointsList.get(i+1));
                    distance = cornerPointsList.get(i+1).distance(cornerPointsList.
    get(i));
238             }
            }
240         return twoReferenceCorners;
        }
```

Algorithmus 5.1: getTwoReferenceCorners()

Die letzte Funktion (Code: 5.2) berechnet den Winkel der zuvor ermittelten Strecke bezogen auf die x-y-Ebene und gibt diesen im Bogenmaß zurück.

```
    // Returns the angle in radians
244 double querneigung(List<Vector3d>twoReferenceCorners){

246     double querneigung;

248     Vector3d richtung = twoReferenceCorners.get(1).subtract(
    twoReferenceCorners.get(0));
        Vector3d xyEbene = Vector3d.create(richtung.getX(),richtung.getY(),
    0.0);
250     querneigung = richtung.angle(xyEbene);

252     return querneigung;
    }
```

Algorithmus 5.2: querneigung()

An dieser Stelle sei angemerkt, dass offensichtlich ein direkt angegebener Wert für die Querneigung oder ein entsprechender Vektor die einfachste Variante für die Ermittlung der Querneigung wäre, dass aber die nächstbeste Option das Vorhandensein der Klasse IfcAlignment wäre. Diese würde es ermöglichen über die enthaltene Richtungsangabe, die dazu senkrechte Strecke und damit den Vektor, der die Querneigung repräsentiert, leichter zu ermitteln.

Programm: Böschungsprüfung

Die Böschungsprüfung ist eine Regel, die ausschließlich alphanumerische Daten ausließt. Diese Form der Regeln hängt vollständig von einer korrekten Informationsführung ab. Bei diesem Check wird untersucht, ob die Böschung als Einschnitt oder Damm klassifiziert ist und ob im Falle eines Einschnitts die vorgegebene Mulde vorhanden ist (siehe Anhang: A, Code: A.3)

5.5.2 Ansichten für das Erstellen von Hinweisobjekten in Konfliktbereichen

Die Manipulation des Modells für die Konflikterkennung ist im Prinzip der Konfliktrückgabe in Solibri sehr ähnlich und unterscheidet sich von dieser dadurch, dass hierfür, die in Solibri bezeichneten Ansichten entwickelt wurden. Hierdurch ist es möglich, interaktiv die IFC-Datei auszuwählen, die bearbeitet werden soll. Für die Erkennung von Konflikten wird wie bei den Solibri-Rulesets ein Programm durchlaufen, das diesen semantisch sehr ähnelt, sich jedoch dadurch unterscheidet, dass die Komponenten über den Auswahlkorb gefiltert und nicht einzeln Komponente für Komponente, sondern gesamt als Kollektion in das Programm geladen werden. Die Algorithmen unterscheiden sich daher primär in der Rückgabeform des Konflikts, der hier in Form eines großen Ausrufezeichens über der Konfliktstelle indiziert wird (Das Einsetzen des Ausrufezeichens wird im kommenden Abschnitt 5.6 erklärt). Es gibt jedoch auch Fälle, wie bei der Untersuchung des Straßenseitenraums, die etwas anders aufgestellt werden müssen. So besitzt die solibrieigene View-Funktion (Funktion, in der die Ansicht definiert wird), nicht die Möglichkeit einen Filter einzubauen wie bei den Rulesets. Wie bereits erwähnt wird daher der Auswahlkorb benutzt, was jedoch zur Folge hat, dass im Gegensatz zu den Rulesets, statt zwei nur eine Möglichkeit der Filterung besteht und somit die Straße nicht gesondert von den zu überprüfenden Objekten gefiltert werden kann. Daher wird in diesem Fall alles in einem Filter definiert und die Objekte über die Propertysets differenziert.

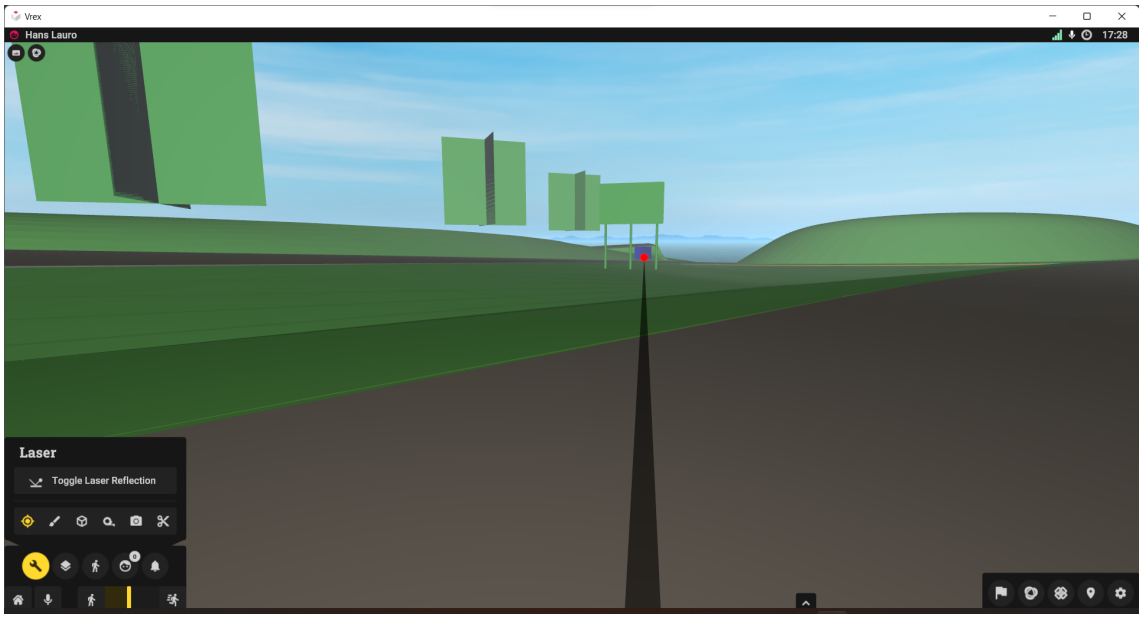


Abbildung 5.11: Sichtprüfung mit Fahrzeugdummy als Referenz

Eines der entwickelten Ansichten "Fahrzeugdummy einfügenërmöglicht es dem Anwender an ausgewählten Stellen ein Objekt platzieren zu können, um so ein Fahrzeug zu simulieren. Dies kann helfen, Blickwinkel in VR besser nachvollziehen zu können und Sichthindernisse zu identifizieren (siehe Abb. 5.11). Hierzu muss er die entsprechende Komponente, auf die das Fahrzeug platziert werden soll, filtern und den Speicherort angeben, an dem das IFC-Schema des Modells zu finden ist. Nachdem er den Button

zur Ausführung des Befehls klickt und das Programm abgeschlossen ist, kann er das neu erzeugte Modell mit den hinzugefügten Objekten öffnen und für die weitere Arbeit nutzen. In VRex kann die Aughöhe so eingestellt werden, dass Fahrer von KFZ oder LKW nachempfunden werden können. Ebenso wäre es möglich, das gleiche Modell über weitere Ansichten zu bearbeiten, und es so mit den entsprechenden Hinweisobjekten zu versehen. Im Folgenden wird anhand des Beispiels für den Fahrzeugdummy das Programm erklärt.

Der Entwurf des Ansicht-Layouts folgt stets denselben Prinzipien. Zuerst wird für das gegebene Panel eine Layoutbasis festgelegt. In diesem Fall ist es eine Gitterstruktur (GridBagLayout), die eine Zuordnung der entsprechenden Felder wie bei einem Schachbrett erlaubt 5.3. Im nächsten Schritt wird einmal definiert, dass sich die Gitter in horizontaler Richtung an das Fenster anpassen und somit von links nach rechts durchlaufen und dann der vertikale Abstand der Felder auf null gesetzt (shouldFill und shouldWeightX sind globale Variablen des Typ Boolean, die außerhalb der Funktion definiert wurden).

```
    public void initializePanel(JPanel panel) {
26
        //to structure the panel the GridBagLayout is used; it divides the
    panel into a grid
28      panel.setLayout(new GridBagLayout());
        GridBagConstraints gridBag = new GridBagConstraints();
30
        if (shouldFill)
32          gridBag.fill = GridBagConstraints.HORIZONTAL;//each grid is
    supposed to be completely filled horizontally

34      if (shouldWeightX)
            gridBag.weightx = 0.0;//The distance between the grids can be
    adjusted
```

Algorithmus 5.3: Gridbag Layout

Danach können die entsprechenden Felder mit den notwendigen Informationen, Buttons und Eingabefeldern über die x- und y-Koordinaten des Gitters (gridx und gridy) befüllt werden. Für das Einsetzen des Fahrzeugdummies sind hierbei nur der Dateipfad des originalen IFC-Schemas, die über einen Button aufgerufen werden können und die Schaltfläche zum Starten des Programms nötig (siehe Code 5.4).

```
        gridBag.gridwidth = 2;
38      JLabel jLabel = new JLabel("The selected path: ");
        gridBag.gridx = 0;//With the gridx and gridy the respective fields are
    selected
40      gridBag.gridy = 1;
        panel.add(jLabel, gridBag);
42
        JLabel jLabelPath = new JLabel("");
44      gridBag.gridx = 2;
        gridBag.gridy = 1;
46      panel.add(jLabelPath, gridBag);

48      gridBag.gridwidth = 4;
```

```
        JButton sourceButton = new JButton("Source IFC");
        gridBag.gridx = 0;
        gridBag.gridy = 2;
        panel.add(sourceButton, gridBag);

        JButton visualizeButton = new JButton("Visualize car");
        gridBag.gridx = 0;
        gridBag.gridy = 3;
        panel.add(visualizeButton, gridBag);

        gridBag.gridx = 0;
        gridBag.gridy = 4;
        panel.add(jlabelStatus, gridBag);
```

Algorithmus 5.4: Inhalt des Fensters

Es sei angemerkt, dass die offizielle Dokumentation von JPanel auf den üblichen Frame ausgelegt ist, also auf ein Fenster, wie es durch eine Windowsanwendung verwendet wird. Da das Panel hier jedoch für einen Solibri-Frame generiert wurde, verhielt sich das Layout in manchen Fällen nicht wie erwartet. So hat die Angabe, dass gridBag.fill als GridBagConstraints.HORIZONTAL definiert ist, nicht den erwünschten Effekt, dass die entsprechenden Felder das Fenster vollständig in horizontalen Richtung ausfüllen. Für die Datenverarbeitung hatte dies keinerlei Bedeutung, jedoch musste bei anderen Ansichten oft experimentiert werden, bis das erwünschte Layout zustande kam.

Danach werden die Buttons mit ihren entsprechenden Funktionen versehen 5.5. Da mit dem ersten der Dateipfad der IFC-Datei definiert werden kann, wird mit der Klasse JFileChooser eine Variable für die Dateipfadeingabe initialisiert und über einen Filter festgelegt, dass nur Dateien des Typs IFC angegeben werden dürfen. Ist ein korrekter Datentyp gewählt worden, wird der Pfad auf dem Interface wiedergegeben.

```
        //the source button invokes the ActionPerformedMethod from the
    ActionListener
        sourceButton.addActionListener(a-> {

            JFileChooser newChooser = new JFileChooser();
            FileNameExtensionFilter filter = new FileNameExtensionFilter("IFC
    Files", "ifc");//only files of ifc format are filtered
            newChooser.setFileFilter(filter);

            int rueckgabeWert = newChooser.showDialog(null, "Auswählen");//on
    the button to choose the file is written "Auswählen"
            path = newChooser.getSelectedFile().getAbsolutePath();

            if (rueckgabeWert == JFileChooser.APPROVE_OPTION) {
                jLabelPath.setText(path);//the chosen file path is now visible
    in the interface
                jlabelStatus.setText("");
            }
        });
```

Algorithmus 5.5: sourceButton

Der nächste Button (siehe Code 5.6) startet das Programm und enthält daher die Hauptfunktion ifcFileManipulation(), auf welche im Folgenden eingegangen wird.

```
            visualizeButton.addActionListener(b -> {
80          try {
                ifcFileManipulation();
82          } catch (IOException e) {
                e.printStackTrace();
84          }
        });
86  }
```

Algorithmus 5.6: visualizeButton

Der erste Schritt besteht daraus, der neuen Datei eine entsprechende Endung zu geben (Zeile 90, 5.7). Existiert diese bereits mit entsprechendem Namen, wird eine Zahl angehängt, die fortlaufend größer wird.

```
88  void ifcFileManipulation() throws IOException {

90      String newFile = path.replace(".ifc", "_WithCarDummy.ifc");//the
    created file copies the name from the original file but changes the end of
    the name

92      String newFileName = getNewFileName(newFile);//if the file already
    exists an increasing number is added

94      File from = new File(path);
        File to = new File(newFileName);
```

Algorithmus 5.7: ifcFileManipulation()(1)

Danach wird der Inhalt des originalen IFC-Schemas kopiert und in die neue Datei mit der zuvor definierten Bezeichnung eingefügt. Dann wird diese in die objectImplementationFunction() geladen. Bei korrekter Durchführung des Einsetzungsprozesses erscheint auf dem Ansichtsfenster ein Text in Grün zur Bestätigung der erfolgreichen Dateiänderung.

```
        copy(from, to);
98

        objectImplementationFunction(newFileName);
100

        jlabelStatus.setText("Successful Ifc-manipulation!");
102     jlabelStatus.setForeground(Color.GREEN);

104 }
```

Algorithmus 5.8: ifcFileManipulation()(2)

Die objectImplementationFunction() (Code 5.9) holt sich die Elemente aus dem Auswahlkorb und übergibt die Koordinaten des Mittelpunkts des Objekts zusammen mit der Höhe, an der das Objekt platziert werden soll, und der zu beschreibenden Datei an die processbuilderFunction().

```
        public static void objectImplementationFunction(String file){
146
        Set<Component> model = SMC.getSelectionBasket().get();
148
        for(Component object : model)
150         processBuilderFunction(object.getBoundingBox().getCentroid().getX()
    ,object.getBoundingBox().getCentroid().getY(),object.getBoundingBox().
    getCentroid().getZ() + carDummyHeight, file);
    }
```

Algorithmus 5.9: objectImplementationFunction()

Diese konvertiert die Koordinaten in Strings, um sie in den ProcessBuilderConstructor (siehe Code 5.10) einlesen zu können, welcher aus der Art des Prozesses, der erstellt wird (Python), der Datei, die den Prozess steuert, den Koordinaten und dem Dateinamen besteht. Theoretisch lassen sich beliebig viele Eingabeparameter erstellen.

```
    static void processBuilderFunction(double x, double y, double z, String
    newFileName) {
154
        String str_x = Double.toString(x);
156     String str_y = Double.toString(y);
        String str_z = Double.toString(z);
158
        try{
160         ProcessBuilder builder = new ProcessBuilder("python", "C:\\
    Masterarbeit Schnittger\\PythonScript\\CarPlacement_Colour.py",str_x,str_y,
    str_z,newFileName);
```

Algorithmus 5.10: processBuilderFunction()(1)

Danach wird der Prozess gestartet (Code 5.11) und die Datei entsprechend des angegebenen Pythonscripts bearbeitet. Um eventuelle Fehler in einer Konsole ausgeben zu können, wird in einer while-Schleife geprüft, ob eine Fehlermeldung ausgegeben wird. Auf die Funktionsweise des Pythonscripts wird im folgenden Kapitel detailliert eingegangen.

```
            Process process = builder.start();
162
            BufferedReader reader = new BufferedReader(new InputStreamReader(
    process.getInputStream()));
164         BufferedReader readers = new BufferedReader(new InputStreamReader(
    process.getErrorStream()));

166         String lines;
            while((lines=reader.readLine())!=null) System.out.println(lines);
168         while((lines=readers.readLine())!=null) System.out.println("Error
    lines"+lines);

170     } catch (Exception e) {
            e.printStackTrace();
172     }
```

```
174         }
    }
```

Algorithmus 5.11: processBuilderFunction()(2)

5.6 IFC-file Manipulation mittels Python

Um das Datenschema IFC zu erklären, wird in vielen Texten die Analogie zu einer PDF gemacht. Obwohl das grundsätzlich nicht falsch ist, vermittelt es den Eindruck als sei es nicht möglich diese zu verändern. IFC ist ein Datenschema, welches mittels eines Texteditors geöffnet und editiert werden kann. Ist die Syntax dahinter verstanden, lässt sich das Modell rein theoretisch auf beliebige Art und Weise verändern. IFC ist jedoch sehr komplex und hat eine vernetzte Struktur, sodass eine Änderung an einer Stelle, viele weitere Änderungen notwendig machen, die daraus resultieren. Um diese Arbeit zu erleichtern, wurden die bereits erläuterten IFC-authoring tools entwickelt. Für diese Arbeit wurde IfcOpenShell gewählt, da es hiermit keine Kompatibilitätsprobleme mit der Solibri-API gab.

IfcOpenShell kann über Python oder C++ genutzt werden. Daher greift die Solibri-API, die auf Java basiert, mittels eines Processbuilder-Constructors auf ein externes Pythonscript zu, welches wiederum die IfcOpenShell Bibliothek nutzt, um das IFC-Schema zu manipulieren. Auf diese Weise wird das Kompatibilitätsproblem umgangen, das bei einem direkten Zugriff des Javaquellcodes auf eine externe Bibliothek eintritt. Ein weiterer Vorteil ist, dass das Pythonscript beliebig verändert werden kann, ohne dass Solibri für die Aktualisierung neu gestartet werden muss, was nach einer Änderung im Javaquellcode notwendig wäre. Das Debugging wird hierdurch deutlich beschleunigt. Zusammenfassend lässt sich festhalten, dass das "Warum"über den Javaquellcode ausgeführt wird, und das "Was"über ein externes Pythonscript.

Das Prinzip ist für alle IFC-Manipulationen gleich. Über die API von Solibri werden, die entsprechenden Komponenten eingelesen, die zuvor durch den Anwender per Filter vorausgewählt wurden. So kann er exemplarisch bestimmen, dass die Straße durch das im Eigenschaftssatz "LBD_IfcIdentifierënthaltene Wort Äsphaltdeckschichtünter der Eigenschaft "Referenz"gefiltert werden soll und die auf Abstand zu prüfenden Objekte aus allen anderen im Auswahlkorb befindlichen Elementen bestehen. Diese werden nun im Javaquellcode eine nach dem anderen auf die gegebene Mindestdistanz geprüft. Ist diese kleiner als der angegebene Grenzwert, werden die Koordinaten des Mittelpunkts der Axis Aligned Bounding Box (AABB) der Komponente ausgelesen und zusammen mit dem Speicherort des Pythonscripts, der zu interpretierenden Sprache, dem Name und der Beschreibung der Fehlermeldung, der Distanz zwischen den beiden Objekten und deren GUID dem Processbuilder-Constructor übergeben. Danach wird der Prozess gestartet (siehe Code 5.12).

```
336 ProcessBuilder builder = new ProcessBuilder("python", "C:\\Masterarbeit
        Schnittger\\PythonScript\\Create_Hint_Colour_IFC_Hierarchie_general.py",
```

```
    str_x,str_y,str_z,newFileName,"Incorrect Crossfall","The crossfall does not
     correspond to the curve radius.",querneigung,fail.getStreetElement().
    getGUID(),"no other", "IfcRatioMeasure","Querneigung");
            Process process = builder.start();
```

Algorithmus 5.12: processbuilder-Constructor

Um beim Debugging nachvollziehen zu können, ob eventuelle Fehlermeldungen beim Durchlaufen des Pythonscripts vorkommen, wurde eine While-Schleife eingebaut, die die fehlerhaften Prozesse mittels getErrorStream() (Code 5.13, Zeile 340) ausliest und der Konsole übergibt (siehe Abb. 5.12), was in der Programmentwicklung sehr hilfreich war, da der Prozess ansonsten eine Black-Box gebildet hätte.

```
            BufferedReader reader = new BufferedReader(new InputStreamReader(
    process.getInputStream()));
340         BufferedReader readers = new BufferedReader(new InputStreamReader(
    process.getErrorStream()));

342         String lines;
            while((lines=reader.readLine())!=null) System.out.println(lines);
344
            while((lines=readers.readLine())!=null) System.out.println("Error
    lines"+lines);
346
        } catch (Exception e) {
348         e.printStackTrace();
        }
350 }
```

Algorithmus 5.13: getErrorStream()

```
while((lines=readers.readLine())!=null) System.out.println("Error lines"+lines);

lines: "  File "C:\Masterarbeit Schnittger\PythonScript\Create_Hint_Colour.py", line 13, in <module>"
```

Abbildung 5.12: Errorstream des Pythonscripts angezeigt in IntelliJ

Innerhalb von Python werden im ersten Schritt die übergebenen Werte in entsprechende Variablen geladen, um so eine klare Übersicht zu erhalten (Code 5.14). Diese umfassen die Koordinaten, an dem der Hinweis platziert werden soll, den Pfad der zu verändernden IFC-Datei, die Informationen, die im Eigenschaftssatz des Hinweises den Konflikt näher beschreiben soll und der Abstand und die GUIDs der beiden Objekte. Außerdem wird ein Ursprung bei den übergebenen Koordinaten und die Richtung der x, y und z-Achse definiert. Wegen eines doppelten Bezugs auf besagtes Koordinatensystem in den Funktionen (1) create_ifcaxis2placement() und (2) create_ifclocalplacement() vervierfachen sich die Werte des Ortsvektors, des Hinweises. Diese wurde durch eine Teilung der Koordinatenwerte durch vier gelöst, da dies simpler war als die Funktionen anzupassen, die an anderer Stelle auf diese Weise gebraucht wurden.

```
6 # Variablen definieren
```

```
num1 = float(sys.argv[1])/4.
num2 = float(sys.argv[2])/4.
num3 = float(sys.argv[3])/4.
pathName = str(sys.argv[4])
pset = sys.argv[5]
description = sys.argv[6]
distance = float(sys.argv[7])
objectGuid = sys.argv[8]
streetPartGuid = sys.argv[9]

# Ursprung definieren
O = num1, num2, num3
X = 1., 0., 0.
Y = 0., 1., 0.
Z = 0., 0., 1.
```

Algorithmus 5.14: Variablen definieren

(1) Daraufhin wird als Erstes eine Funktion definiert, mittels derer der Ursprung und die Orientierung des Koordinatensystems festgelegt werden kann, die den Raum aufspannt, in dem das Objekt platziert werden soll und die Entität IfcAxis2Placement3D zurückgibt.

```
# 1. Creates an IfcAxis2Placement3D from Location, Axis and RefDirection
    specified as Python tuples
def create_ifcaxis2placement(ifcfile, point=O, dir1=Z, dir2=X):
    point = ifcfile.createIfcCartesianPoint(point)
    dir1 = ifcfile.createIfcDirection(dir1)
    dir2 = ifcfile.createIfcDirection(dir2)
    axis2placement = ifcfile.createIfcAxis2Placement3D(point, dir1, dir2)
    return axis2placement
```

Algorithmus 5.15: create_ifcaxis2placement() CHEN, 2015

(2) Darauf aufbauend definiert nun die nächste Funktion eine Entität der Klasse IfcLocalPlacement, um ein Produkt (IfcProduct) oder alles innerhalb dessen Vererbungshierarchie relativ zu dem zuvor bestimmten Koordinatensystem zu platzieren.

```
# 2. Creates an IfcLocalPlacement from Location, Axis and RefDirection,
    specified as Python tuples, and relative placement
def create_ifclocalplacement(ifcfile, point=O, dir1=Z, dir2=X, relative_to=None
    ):
    axis2placement = create_ifcaxis2placement(ifcfile,point,dir1,dir2)
    ifclocalplacement2 = ifcfile.createIfcLocalPlacement(relative_to,
    axis2placement)
    return ifclocalplacement2
```

Algorithmus 5.16: create_ifclocalplacement() CHEN, 2015

(3) Mittels der nächsten Funktion kann eine Polylinie erzeugt werden, die die Grundfläche des einzusetzenden Objekts bestimmt.

```
# 3. Creates an IfcPolyLine from a list of points, specified as Python tuples
def create_ifcpolyline(ifcfile, point_list):
```

```
    ifcpts = []
42  for point in point_list:
        point = ifcfile.createIfcCartesianPoint(point)
44      ifcpts.append(point)
    polyline = ifcfile.createIfcPolyLine(ifcpts)
46  return polyline
```

Algorithmus 5.17: create_ifcpolyline() CHEN, 2015

(4) Dies wird in der letzten Funktion genutzt, um somit die Polylinie zu einem Volumenkörper, einem sogenannten Festkörper aus extrudierter Fläche (IfcExtrudedAreaSolid), zu extrudieren. Übergeben werden hierbei die Punkte der Polylinie, der Ort im relativen Koordinatensystem und die Richtung und Länge der Extrusion.

```
48  # 4. Creates an IfcExtrudedAreaSolid from a list of points, specified as Python
        tuples
    def create_ifcextrudedareasolid(ifcfile, point_list, ifcaxis2placement,
        extrude_dir, extrusion):
50      polyline = create_ifcpolyline(ifcfile, point_list)
        ifcclosedprofile = ifcfile.createIfcArbitraryClosedProfileDef("AREA", None,
         polyline)
52      ifcdir = ifcfile.createIfcDirection(extrude_dir)
        ifcextrudedareasolid = ifcfile.createIfcExtrudedAreaSolid(ifcclosedprofile,
         ifcaxis2placement, ifcdir, extrusion)
54      return ifcextrudedareasolid
```

Algorithmus 5.18: create_ifcextrudedareasolid() CHEN, 2015

Nachdem in create_guid eine Lamdafunktion definiert wird (Code 5.19), die später für die Generierung immer neuer GUIDs verwendet wird, wird die IFC-Datei geöffnet und in der Variable ifcfile gespeichert, um die Funktionen der IfcOpenShell-Bibliothek nutzen zu können. Danach werden weitere Variablen definiert, die im späteren Verlauf immer wieder Verwendung finden.

```
56  create_guid = lambda: ifcopenshell.guid.compress(uuid.uuid1().hex)

58  ifcfile = ifcopenshell.open(pathName)
    owner_history = ifcfile.by_type("IfcOwnerHistory")[0]
60  project = ifcfile.by_type("IfcProject")[0]
    context = ifcfile.by_type("IfcGeometricRepresentationContext")[0]
```

Algorithmus 5.19: Lambdafunktion

Da sich in der IFC-Struktur, per Konvention, nur IfcSite auf das globale Koordinatensystem bezieht und alle in der Raumstruktur-Hierarchie darunterliegenden Objekte sich mittels IfcLocalPlacement auf die jeweils höhere Instanz beziehen, muss für das eingesetzte Objekt diese Hierarchie beachtet werden. Wie bereits in Kapitel 2.3 erläutert, gibt es hier je nach Projektart Unterschiede. Alle Projekte beginnen mit IfcProject -> IfcSite doch gliedern sich Hochbauprojekte danach in IfcBuilding -> IfcBuildingStorey -> IfcSpace wohingegen Infrastrukturprojekte in IfcFacility -> IfcFacilityPart -> IfcSpace geordnet sein sollten. Um dieser Syntax treu zu werden, die Abhängigkeiten entsprechend definiert, sodass am

Ende ein IfcLocalplacement bestimmt ist, das seine Platzierung relativ zum Gebäude setzt (Code 5.20).

```
# IFC hierarchy creation
site_placement = create_ifclocalplacement(ifcfile)
building_placement = create_ifclocalplacement(ifcfile, relative_to=
    site_placement)
storey_placement = create_ifclocalplacement(ifcfile, relative_to=
    building_placement)
```

Algorithmus 5.20: IFC Hierarchie

Unabhängig von der Platzierung muss das Objekt jedoch auch in der Projektstruktur korrekt eingegliedert sein. An dieser Stelle ist aufgefallen, dass der Export die Hierarchiegliederung beschädigt hat. So beziehen sich nach dem Export die Objekte in IfcRelContainedInSpatialStructure nicht mehr auf ein Stockwerk, dargestellt in IfcBuildingStorey, sondern direkt auf IfcBuilding. Um diese korrekt zu erfassen, liest das Programm alle Hierarchieebenen aus und fügt sie einer Liste hinzu, ignoriert den Befehl jedoch, wenn diese Instanz nicht existiert (Code 5.21, Zeile 75).

```
buildingsList = []
infrastructure = []

try :
  site = ifcfile.by_type("IfcSite")[0]
  buildingsList.append(site)
  infrastructure.append(site)
except IndexError:
  pass

# From here it can be IfcBuilding or IfcFacility. Both are verified.
# IfcBuilding:
try :
  building = ifcfile.by_type("IfcBuilding")[0]
  buildingsList.append(building)
except IndexError:
  pass

try :
  building_storey = ifcfile.by_type("IfcBuildingStorey")[0]
  buildingsList.append(building_storey)
except IndexError:
  pass
```

Algorithmus 5.21: buildingsList

Auf diese Weise kann einerseits ein IndexError vermieden werden, aber auch eine Abfrage erstellt werden, die zwischen einem Projektaufbau für Gebäude und Infrastruktur unterscheidet. Hierbei ist aufgefallen, dass IfcOpenShell den Typ IfcFacility und IfcFacilityPart nicht kennt und einen "Type not defined Error ausgibt, welcher daher an dieser Stelle mit except gefangen wird (Code 5.22). Adaptiert IfcOpenshell sich jedoch an die aktuellen IFC-Schemas, kann das Programm die Struktur korrekt auslesen. Die erstellte Liste dient

hierbei später als Bezugselement, da das zuletzt hinzugefügt Objekt die räumliche Struktur bildet, in die das Objekt eingefügt wird.

```
92 # IfcFacility:
   try :
94   facility = ifcfile.by_type("IfcFacility")[0]
     infrastructure.append(facility)
96 except :
     pass
98
   try :
100  facilityPart = ifcfile.by_type("IfcFacilityPart")[0]
     infrastructure.append(facilityPart)
102 except :
     pass
```

Algorithmus 5.22: infrastructure

Im Folgenden wird die Form des eingesetzten Objekts bestimmt. Da es sich hier um ein Ausrufezeichen als Hinweis handelt, sind es zwei Körper, die durch zwei getrennte Extrusionen erstellt und dann zu einem Gesamtkörper in IfcShapeRepresentation vereinigt werden. Durch die Bestimmung einer Grundfläche (Code 5.23, Zeile 113), der Richtung und Position der Extrusion (Zeile 111-112) und die Möglichkeit, die Körper zusammenzufügen (Zeile 114-116) kann in der Form variiert werden. Danach wird product_shape die beiden IfcShapeRepresentation axis_representation und body_representation zusammengefügt und das Hinweisobjekt mit dieser Form als IfcBuildingElementProxy instanziiert (Zeile 120).

```
106 # Exclamation mark creation: Define the Exclamation mark shape as a polyline
        axis and an extruded area solid
    hint_placement = create_ifclocalplacement(ifcfile, relative_to=storey_placement
        )
108 polyline = create_ifcpolyline(ifcfile, [(0.0, 0.0, 0.0), (5.0, 0.0, 0.0)])
    axis_representation = ifcfile.createIfcShapeRepresentation(context, "Axis", "
        Curve2D", [polyline])
110
    extrusion_placement = create_ifcaxis2placement(ifcfile, (0.0, 0.0, 0.0), (0.0,
        0.0, 1.0), (1.0, 0.0, 0.0))
112 extrusion_placement_2 = create_ifcaxis2placement(ifcfile, (0.0, 0.0, 4.0),
        (0.0, 0.0, 1.0), (1.0, 0.0, 0.0))
    point_list_extrusion_area = [(0.0, -0.5, 0.0), (1.0, -0.5, 0.0), (1.0, 0.5,
        0.0), (0.0, 0.5, 0.0), (0.0, -0.5, 0.0)]
114 solid = create_ifcextrudedareasolid(ifcfile, point_list_extrusion_area,
        extrusion_placement, (0.0, 0.0, 1.0), 1.0)
    solid_2 = create_ifcextrudedareasolid(ifcfile, point_list_extrusion_area,
        extrusion_placement_2, (0.0, 0.0, 1.0), 11.0)
116 body_representation = ifcfile.createIfcShapeRepresentation(context, "Body", "
        SweptSolid", [solid, solid_2])

118 product_shape = ifcfile.createIfcProductDefinitionShape(None, None, [
        axis_representation, body_representation])
```

```
120 hint = ifcfile.createIfcBuildingElementProxy(create_guid(), owner_history, "
    Hint", "A hint to a conflict", None, hint_placement, product_shape, None)
```

Algorithmus 5.23: Entwurf des Ausrufezeichens

Nachdem die Entität kreiert worden ist, wird die Farbgebung definiert (Code 5.24). Hierfür wurde mit IfcSurfaceStyle gearbeitet. Dieses enthält, mittels IfcSurfaceStyleRendering übergeben, neben der Farbe, die in RGB gegeben ist, Werte die diffuse oder die regelmäßige Reflexion definieren. Zusätzlich kann bestimmt werden, auf welcher Seite die Darstellung angelegt werden soll, also auf der Außen- oder Innenseite des Objekts. Über "BOTH"(Zeile 128) wurden beide Seiten gewählt und mittels IfcPresentationStyleAssignment (Zeile 129) wird die zuvor definierte Darstellung den jeweiligen IfcExtrudedAreaSolid-Objekten (Punkt und Strich des Ausrufezeichens) zugeordnet.

```
122 # Farbgebung des Objekts
colour1 = ifcfile.createIfcColourRgb(None,0.99,0.0,0.0)
124 colour2 = ifcfile.createIfcColourRgb(None,0.99,0.0,0.0)
normalised = ifcfile.createIfcNormalisedRatioMeasure(0.0)
126 specular = ifcfile.createIfcSpecularRoughness(0.4)
surface_style_rendering = ifcfile.createIfcSurfaceStyleRendering(colour1, 0.0,
    colour2, None, None, None, normalised, specular,"FLAT")
128 surface_style = ifcfile.createIfcSurfaceStyle("Material","BOTH",[
    surface_style_rendering])
presentationstyle_assignment = ifcfile.createIfcPresentationStyleAssignment([
    surface_style])
130 styled_item = ifcfile.createIfcStyledItem(solid,[presentationstyle_assignment],
    "Material")
styled_item_2 = ifcfile.createIfcStyledItem(solid_2,[
    presentationstyle_assignment],"Material")
```

Algorithmus 5.24: Farbgebung des Objekts

Danach werden neben dem Material und den Mengenangaben noch Eigenschaftssätze definiert. Da es sich um ein Hinweisobjekt handelt, sind die ersten beiden Punkte irrelevant, müssen aber für eine korrekte Syntax erstellt werden (siehe A, Code A.4). Über die Eigenschaftssätze werden dem Anwender notwendige Informationen vermittelt (Code 5.25). Für diesen Test wurde die Bezeichnung und die Beschreibung des Konflikts, der Abstand zwischen den beiden Objekten und deren GUIDs eingefügt.

```
150 # Create and assign property set
property_values = [
152       ifcfile.createIfcPropertySingleValue("Conflict", "Reference", ifcfile.
    create_entity("IfcText", description), None),
      ifcfile.createIfcPropertySingleValue("Distance", "Distance", ifcfile.
    create_entity("IfcPositiveLengthMeasure", distance), None),
154   ifcfile.createIfcPropertySingleValue("Object that is too close", "Object that
     is too close", ifcfile.create_entity("IfcText", objectGuid), None),
  ifcfile.createIfcPropertySingleValue("Referred to street part", "Referred to
    street part", ifcfile.create_entity("IfcText", streetPartGuid), None)
156 ]
```

```
158 property_set = ifcfile.createIfcPropertySet(create_guid(), owner_history, pset,
        None, property_values)
    ifcfile.createIfcRelDefinesByProperties(create_guid(), owner_history, None,
        None, [hint], property_set)
```

Algorithmus 5.25: Eigenschaftssatz des Objekts

Im letzten Schritt wird verifiziert, ob die Projekthierarchie für Gebäude oder für Infrastruktur angelegt ist und dann die kleinste Ebene als Bezugsraum für das Objekt gewählt. Danach werden die erzeugten Veränderungen in die IFC-Datei geschrieben.

```
162 ## Relate the exclamation mark to the respective IFC Hierarchie (Decision
        between building or infrastructure hierarchie)
    if 'building' in locals() or 'building_storey' in locals():
164   ifcfile.createIfcRelContainedInSpatialStructure(create_guid(), owner_history,
        "Spatial Structure Container Building", None, [hint], buildingsList[-1])

166 else :
      ifcfile.createIfcRelContainedInSpatialStructure(create_guid(), owner_history,
        "Spatial Structure Container Facility", None, [hint], infrastructure[-1])
168
    # Write the contents of the file to disk
170 ifcfile.write(pathName)
```

Algorithmus 5.26: Einfügen des Hinweisobjekts

6. Fallstudie

Im vorherigen Kapitel wurden die Methoden erläutert, mittels denen die Modelle vorbereitet und analysiert wurden. Es wurden die entwickelten Programme erklärt und wie Im- und Exporte richtig konfiguriert werden. Hiermit wurde ein Einblick in die Datenerzeugung gegeben, die Grundlage für die modellbasierte Unterstützung des Auditors bildet und eine Aussage über die Datenqualität ermöglicht. In diesem Kapitel wird auf die praxisorientierte Umsetzung eingegangen und untersucht, wie sich die entwickelten Tools in der Anwendung bewährten, welche Vorteile sie brachten und welche Limitationen sich ergaben.

6.1 Virtuelle Begehung der B299

Um eine fachliche Resonanz zu den entwickelten Tools in Kombination mit VR zu erhalten, wurde mit den Auditoren des Landshuter Projekts ein virtuelles Audit durchgeführt. Hierzu wurde vor Ort in einer Präsentation die Durchführung des Audits mit den entwickelten Programmen am Building Information Model der B299 erläutert und dann die Beurteilung der Auditoren nach einer eigenständigen Anwendung ausgewertet.

6.2 Konzeptanwendung Konfliktrückgabe in Solibri

Wie bereits im vorherigen Kapitel erläutert, werden zwei Möglichkeiten der Defiziterkennung genutzt. In diesem Abschnitt wird die Variante erläutert, die Tests über SMC-Rulesets durchführt und die erkannten Konflikte über BIMcollab an die VR-Umgebung übermittelt. Als Erstes wird hierzu die vorbereitete IFC-Datei in Solibri geladen und die nötigen Rulesets im Ruleset Manager ausgewählt (siehe Abb. 6.1). Daraufhin sind die Rulesets im Modellinterface verfügbar und können für eine Prüfung ausgewählt werden. In den nächsten Abschnitten wird auf eine Auswahl an Prüfungen eingegangen und ihre Durchführung erläutert.

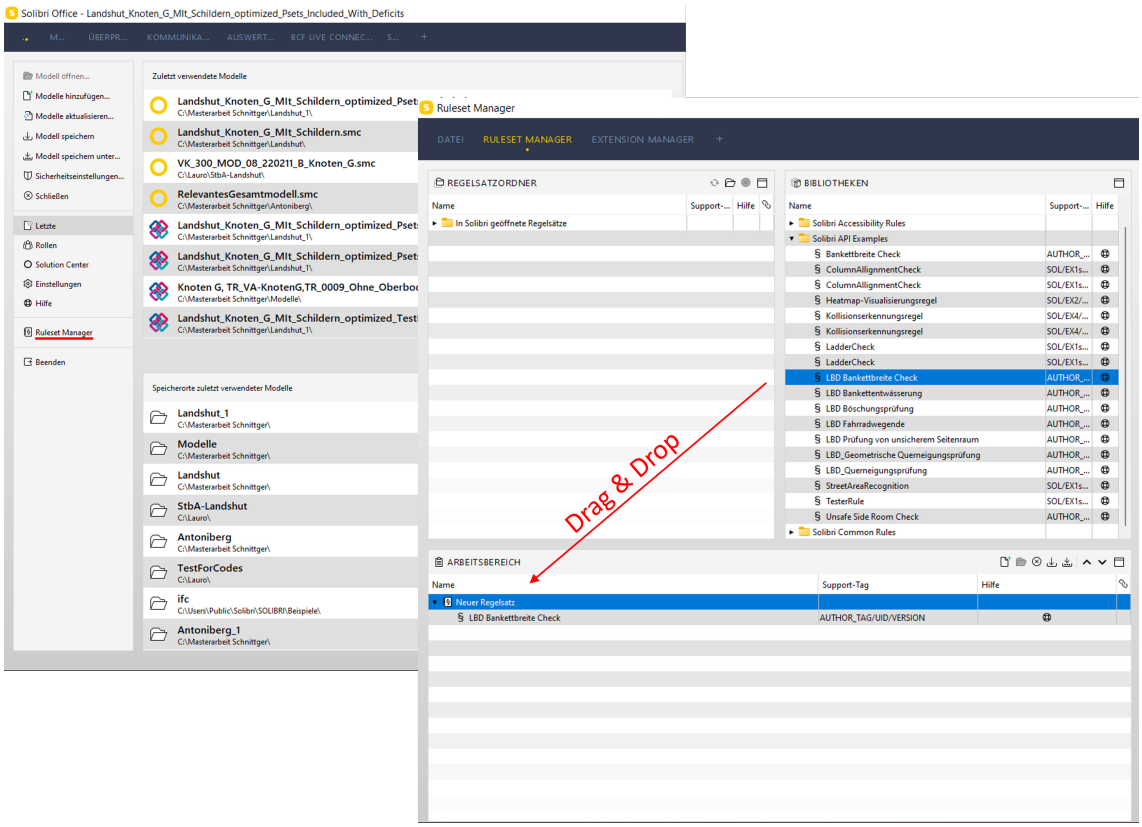


Abbildung 6.1: Ruleset Manager

6.2.1 Seitenraumprüfung

Der erste Check dient der Überprüfung der Seitenräume mit der Regel LBD Unsafe Side Room Check. Hierzu muss der Mindestabstand angegeben werden, den ein Objekt, gemessen vom nächsten Punkt der Straße, haben muss. Im Filter des Checks wird hierzu alles ausgewählt, das verifiziert werden soll (siehe Abb. 6.2). Das heißt, es werden alle Objekte einbezogen bis auf Böschungsentitäten, die Straße selbst und die Drainage. Wichtig bei dieser Regel ist die Nutzung des Auswahlkorbs, da dieser die Straßenkomponenten definiert. Hierfür wird am besten der Filter in den Ansichten genutzt. Auf diese Weise kann die Straßenoberfläche als Referenz genutzt werden (Asphaltdeckschicht). Die Korbsymbole rechts oben im Fenster des Filters geben die Möglichkeit, die Objekte dem Auswahlkorb hinzuzufügen, sie diesem zu entnehmen oder die gefilterten Objekte komplett dem Auswahlkorb zuzuordnen. Die Zahl rechts unten im Fenster (siehe Abb. 6.2) zeigt die mit dem Filter ausgewählten Elemente an, die sich im Auswahlkorb befinden. Nun da definiert ist, welche Objekte auf Abstand geprüft werden sollen, welche die Straße bilden und die Parameter gesetzt sind, kann der Check gestartet werden.

Die im Filter ausgewählten Komponenten werden jetzt eine nach der anderen in das Programm geladen und geprüft. In der Schleife werden alle Elemente durchlaufen, die im Auswahlkorb definiert sind, also 742 Durchläufe. Es wird zuerst abgefragt, ob es einen Abstand zwischen den beiden Objekten gibt und falls ja, wird geprüft, ob dieser kleiner als der angegebene Sicherheitsabstand ist. Ist dies der Fall, wird die Komponente mit

der angehängten Beschreibung zurückgegeben, ansonsten bleibt der Rückgabewert leer. Nach in etwa 20 s werden für das gegebene Beispiel fünf fehlerhafte Objekte gefunden.

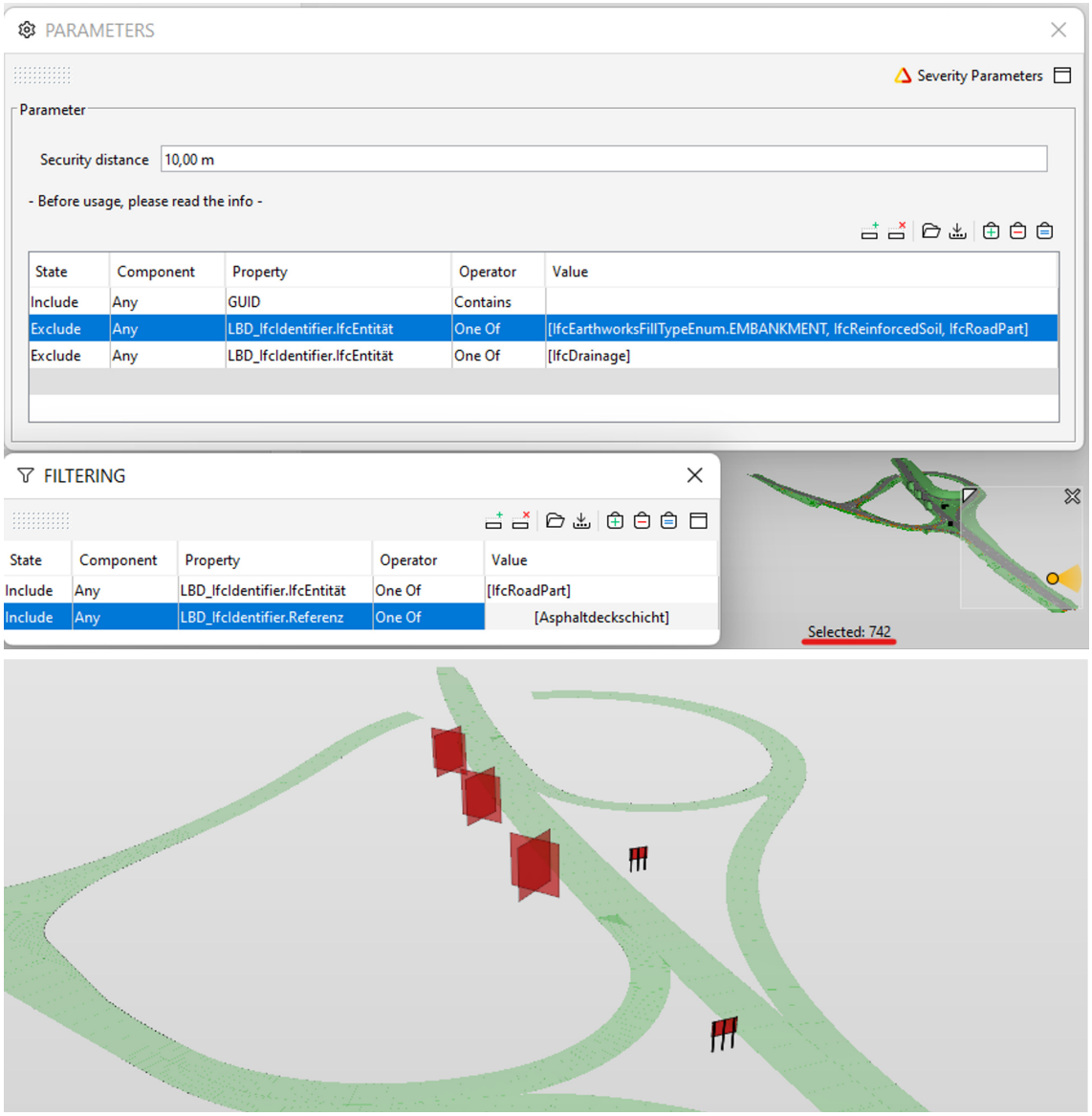


Abbildung 6.2: [oben]Parameter für Seitenraumcheck und Filter aus Ansicht, [unten] Ergebnis

6.2.2 Querneigungsprüfung

Als Nächstes wird geprüft, ob die Querneigung relativ zum Radius korrekt ist. Hierzu wird wie zuvor die Regel im Ruleset Manager per Drag & Drop in den aktuellen Regelkatalog gezogen. Anders als bei der Seitenraumüberprüfung benötigt dieser Check Informationen, die in den Eigenschaftssätzen der Komponenten enthalten sind, weshalb nicht nur der Filter für die Erkennung der Straßenoberfläche definiert werden muss, sondern auch der Name des Eigenschaftssatzes, in dem der Wert des Kurvenradius enthalten ist. Im gegebenen Beispiel wird der Begriff "Daten_Aus_RevitExport"für den Eigenschaftssatz eingegeben und für den Kurvenradius muss zwischen Înfra_RadiusFrom(Pset_Desite)ünd

Înfra_RadiusTo(Pset_Desite)ëntschieden werden. Diese Werte können unterschiedlich sein, da das Streckenelement zur einen Seite sich einer Geraden und zur anderen einer Kurve zuordnen kann (siehe Abb. 6.3). Leider war es nicht möglich, die Seite, auf der die Querneigung gemessen wird, durch einen Parameter zu definieren, da die Straße keine Variable für eine Richtung enthält, weshalb der Kurvenradius in manchen Fällen nicht der korrekten Querneigung zugeordnet wird. Hierbei wird also vorausgesetzt, dass der Auditor weiß, welche Seite, durch "Fromünd "To"definiert ist und im Falle einer Diskrepanz über die visualisierten Messpunkte prüfen kann, ob die Querneigungsseite mit dem Radius übereinstimmt. Falls, dem nicht so ist, kann er über das benachbarte Element die Neigung ermitteln, vorausgesetzt das Programm hat dort die Querneigung an der korrekten Stelle gemessen.

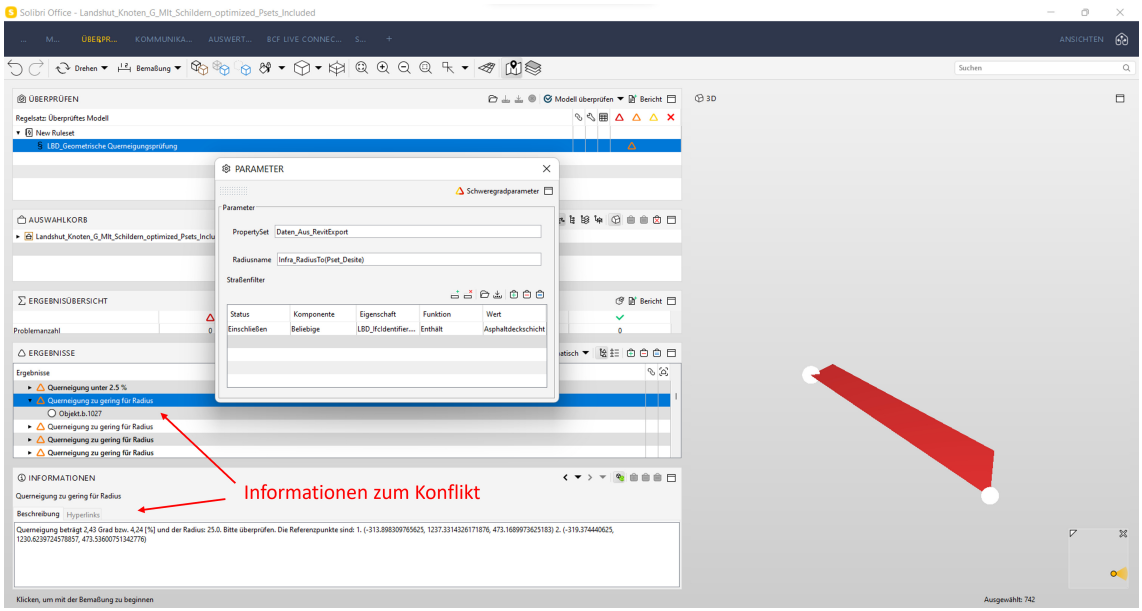


Abbildung 6.3: Querneigungsprüfung

6.2.3 Anzeigen von Radwegenden

Um sich anzeigen zu lassen, wo der Radweg in eine Kraftfahrzeugstraße endet, kann die Regel LBD_Fahrradwegende verwendet werden. Hierfür wird im Regelfilter der Radweg mittels des im Eigenschaftssatz LBD_RoadProperties enthaltenen Begriffs für die Verkehrswegbezeichnung (Radweg) und im Auswahlkorb die Kraftfahrzeugstraße (Nebenfahrbahn und B299) definiert. Berührt eines der Elemente die Straße mit einer Toleranz von 30 cm wird es als Radwegende definiert und visuell hervorgehoben.

6.2.4 Prüfen der Bankettbreite

Um die Bankettbreite zu prüfen, müssen diese im Filter definiert werden. Da sie im gegebenen Modell nicht explizit modelliert sind, wird stattdessen das Bankettmaterial ausgewählt. Dieses ist jedoch nicht wie ein Quader, sondern unregelmäßig geformt und passt sich an die entsprechende Umgebung an, weshalb die ausgelesene Breite nicht

immer mit der eigentlichen Kronenbreite übereinstimmt und daher die Ergebnisse teilweise Bankette nicht enthalten, obwohl sie schmaler als die angegebene Mindestbreite sind. Wie im Kapitel 5 erläutert, kann dieses Problem durch die Information der Kronenbreite oder besser eine explizite Modellierung gelöst werden.

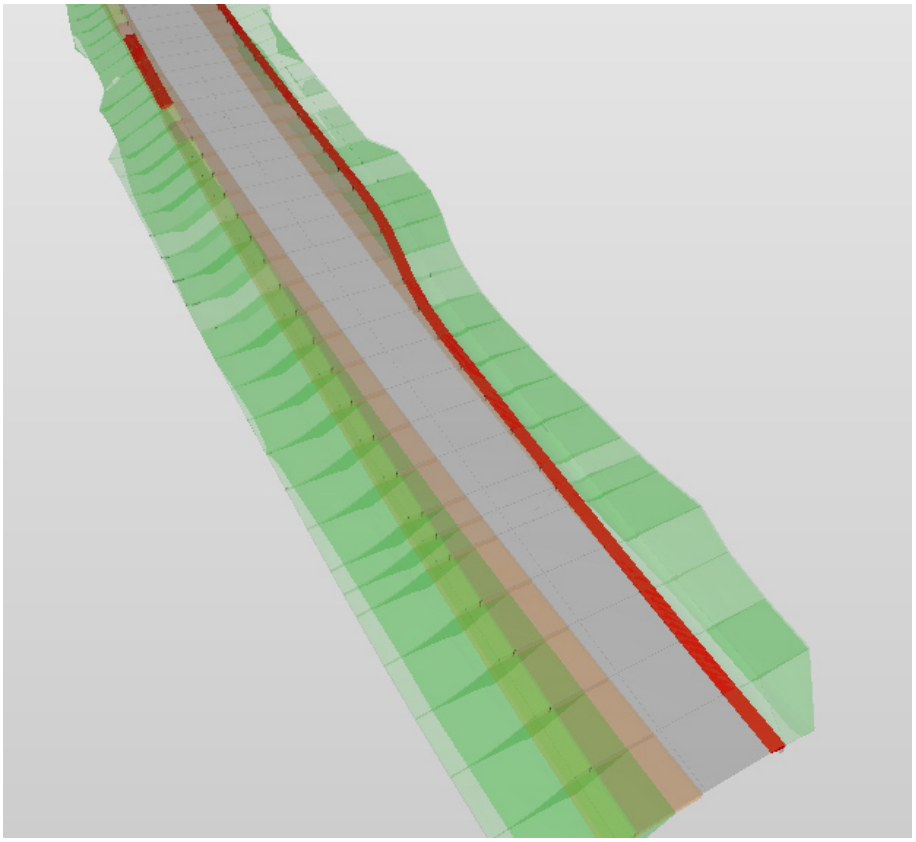


Abbildung 6.4: Bankettprüfung

6.2.5 Prüfen der Böschung

Die Prüfung der Böschung ist ein gutes Exempel für eine Abfrage, die rein auf der Basis der semantischen Informationen stattfindet. Ziel der Prüfung ist zu verifizieren, ob die Böschung einen Einschnitt bildet und wenn ja, ob sie an eine notwendige Mulde angrenzt. Hierfür müssen nur die entsprechenden Begriffe und im Filter die Klasse für die Böschung eingegeben werden. Im vorbereiteten Modell wurde ein Böschungselement mit den falschen Informationen ausgefüllt, was zur Folge hat, dass dieses als Konflikt zurückgegeben wird (Abb. 6.5).

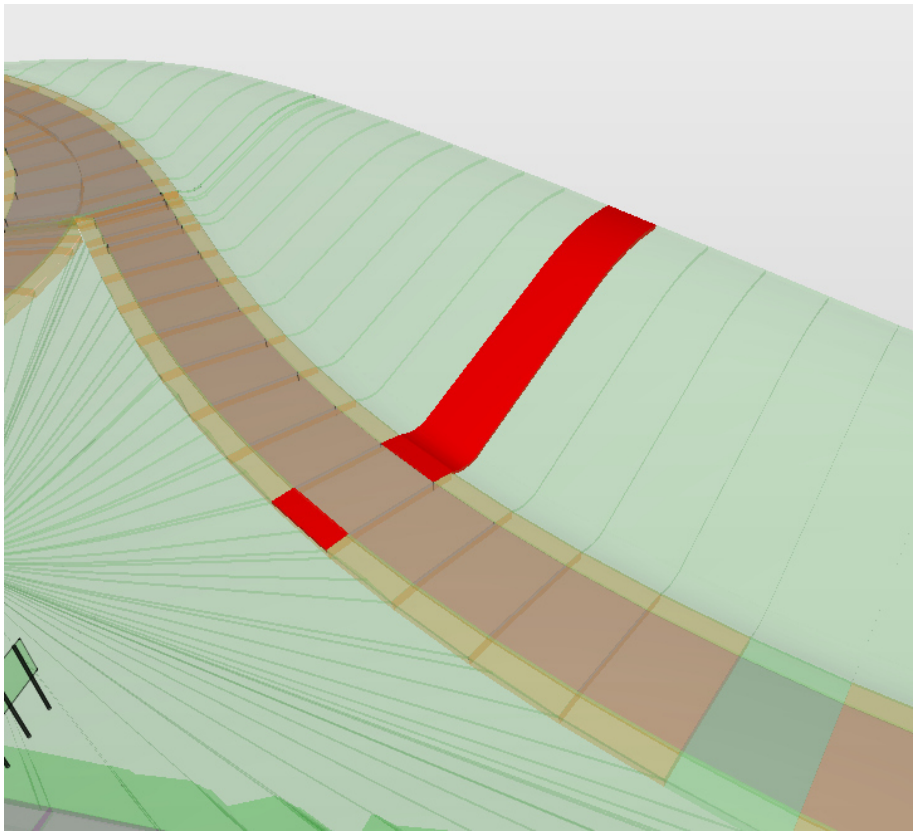


Abbildung 6.5: Böschungsprüfung

Die Voraussetzung, dass für jedes Böschungselement eine Information eingepflegt wird, die Aussage über eine benachbarte Mulde enthält, ist unpraktikabel, dient aber der Illustration des Nutzens einer hochwertigen IFC-Datenqualität, dessen Semantik eine solche Aussage ohne zusätzliche Arbeit ermöglichen würde. So enthalten vollkommen integre Modelle eine Information über die Beziehung der Objekte zueinander. Diese können unterschiedlicher Natur sein und enthalten unter anderem die Möglichkeit, dass ein Objekt ein anderes berührt (IfcRelConnectsElements) oder ein übergeordnetes Element aus untergeordneten Elementen besteht (IfcRelDecomposes). Diese Information wiederum ermöglicht es, eine Aussage darüberzumachen, ob eine Böschung eine Mulde besitzt.

Die restlichen Prüfungen folgen demselben Prinzip und werden an dieser Stelle nicht erläutert. Das Ergebnis der Checks wird in Solibri in der Ansicht Ergebnisse angezeigt und besteht aus den ermittelten Komponenten und den generierten Kommentaren, die beim Klicken auf den Konflikt angezeigt werden. Der Auditor kann diese Objekte nun einer genaueren Untersuchung unterziehen und gegebenenfalls einen Konflikt kreieren. Unter dem Reiter BCF Live Connector kann er sich über den Button mit dem BIMcollab-Server verbinden (siehe Abb. 6.6) und bereits erstellte Konflikte herunter- beziehungsweise neue hochladen. Eine sehr praktische Funktion an dieser Stelle ist "Probleme Importieren", das rechts neben dem Button zur Verbindung mit dem BIMcollab-Server auswählbar ist. Hier können die Konflikte importiert werden, die entweder bereits in einer Präsentation auf die wichtigsten Kernaspekte reduziert wurden oder direkt ausgewählt werden. Vorteil dieses

Verfahrens ist, dass die Konfliktbeschreibungen dadurch direkt in die Problemdetails überführt werden.

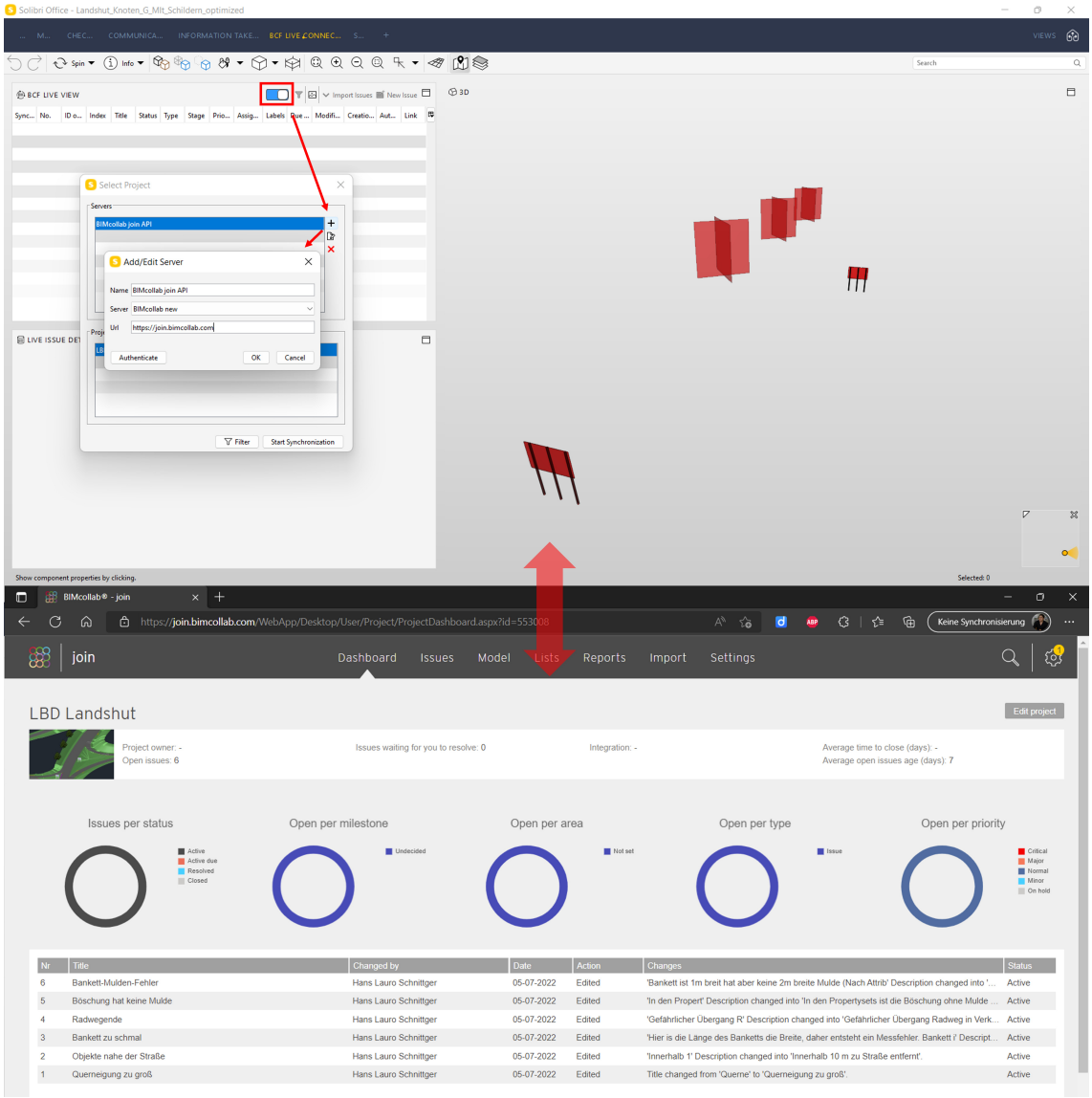


Abbildung 6.6: Verbindung zum BIMcollab-Server

6.2.6 Arbeiten in VR

Alle BCF-kompatiblen Anwendungen können die erzeugten Konflikte in ihrer Umgebung betrachten und gegebenenfalls bearbeiten. Hier kommt VRex ins Spiel, das einzige auf dem Markt verfügbare VR-Tool, das einen Zugriff auf BCF ermöglicht und somit die Konflikte in VR zugänglich macht. Im Webaccount von VRex kann ein Projekt erstellt und das Landshuter Modell hochgeladen werden. Unter Einstellungen kann BIMcollab als der Standard Issue Manager gewählt werden. Ist dies geschehen, kann nun der VRex Viewer, falls noch nicht getan, heruntergeladen (siehe Abb. 6.7) und die Anwendung gestartet werden.

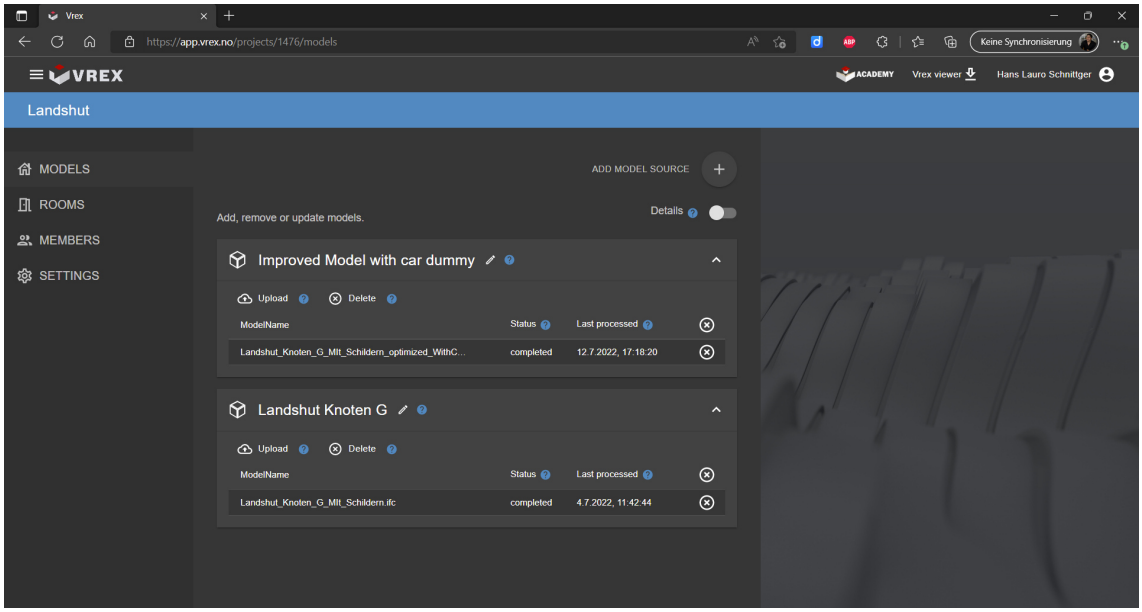


Abbildung 6.7: Hochgeladene Modelle im Landshuter Projekt im Vrex Webaccount

In VRex muss das entsprechende Projekt gewählt und zwischen dem VR- oder Laptopmodus gewählt werden. In der virtuellen Realität können nun die Konflikte über die Verbindung zum BIMcollab-Server eingesehen werden. Um diese herzustellen, muss rechts unten im Bild das Symbol angewählt (siehe Abb. 6.8) und der Button eingeschaltet werden. Danach kann eine Verbindung zum Server hergestellt und autorisiert werden. Die Konflikte sind nun in VRex einsehbar.

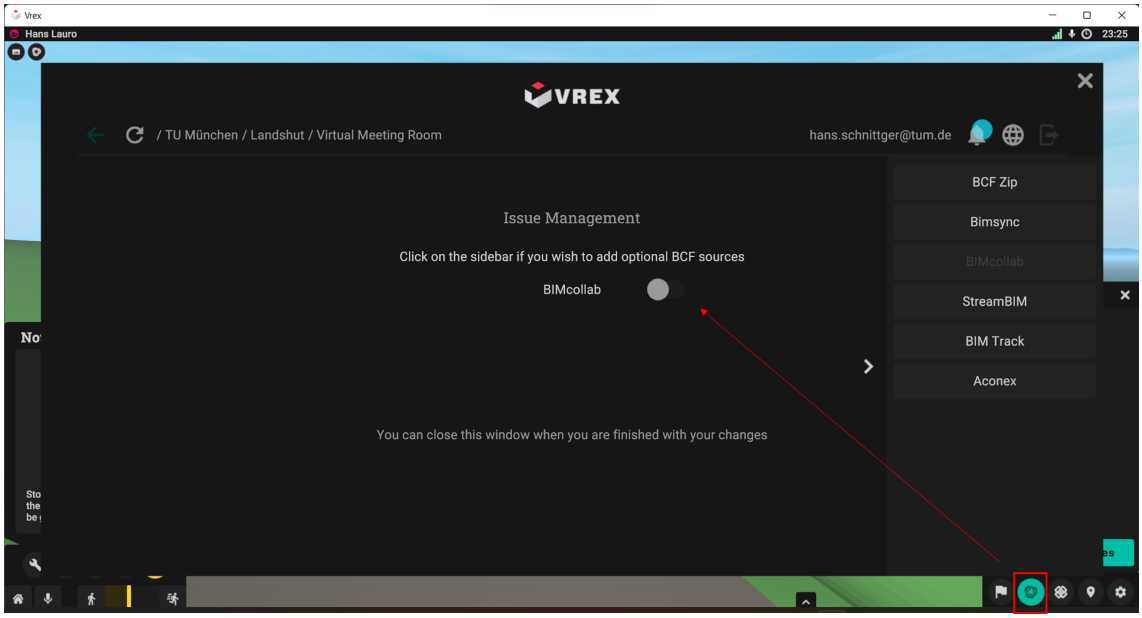


Abbildung 6.8: Wahl des BCF-Servers für das Konfliktmanagement

Die jeweiligen Konflikte können nun angeklickt und untersucht werden. Alle Informationen, wie der Name, die Beschreibung und der Ersteller, die bei Entwurf des Konflikts hinzugefügt wurden, sind hier auch zu sehen. Die Viewpoints enthalten Bilder für ein besseres Verständnis und über eine Teleportationsfunktion, kann der Anwender sich direkt zu die-

sem bringen lassen und das Objekt inspizieren. Über die Objektwahlfunktion, kann der Anwender, das auszuwählende Element anklicken und die Informationen einsehen (siehe Abb. 6.9). Für beispielsweise den Konflikt der zu geringen Querneigung kann hierbei der Radius geprüft und bei einer tatsächlich vorhandenen Gefahrenstelle die entsprechende Kilometrierung als Sicherheitsdefizit aufgenommen werden.

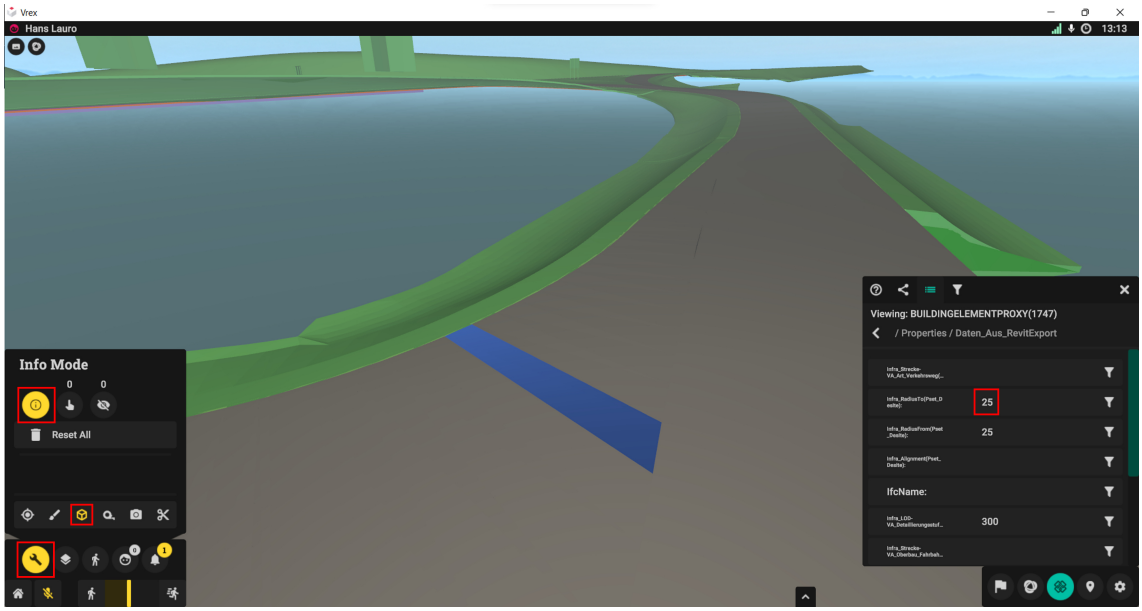


Abbildung 6.9: Infomodus des VRex-Viewers

6.3 Konzeptanwendung: IFC-Manipulation

Die zweite Variante wird über die Ansichten ausgeführt. Hier kann ein Fenster geöffnet werden, das den User auffordert, neben der IFC-Datei des Modells alle nötigen Parameter anzugeben. Das Prinzip ist zur ersten Variante sehr ähnlich, doch erstellt der Anwender in diesem Fall eine neue IFC-Datei, die Hinweissymbole über den Konfliktstellen enthält und so für die weitere Arbeit genutzt werden kann.

6.3.1 Seitenraumprüfung

Für die Überprüfung des unsicheren Seitenraums müssen dieses Mal statt zwei Filter nur einer genutzt werden. Dies ist der Tatsache geschuldet, dass bei der Entwicklung der Ansicht es nicht möglich ist zusätzlich einen Filter zu inkludieren, wie bei den entwickelten Regeln. Dementsprechend kann nur der Standardfilter verwendet werden, um die Objekte in den Auswahlkorb zu laden. Zur Veranschaulichung werden die beiden Konzepte an dieser Stelle gegenübergestellt.

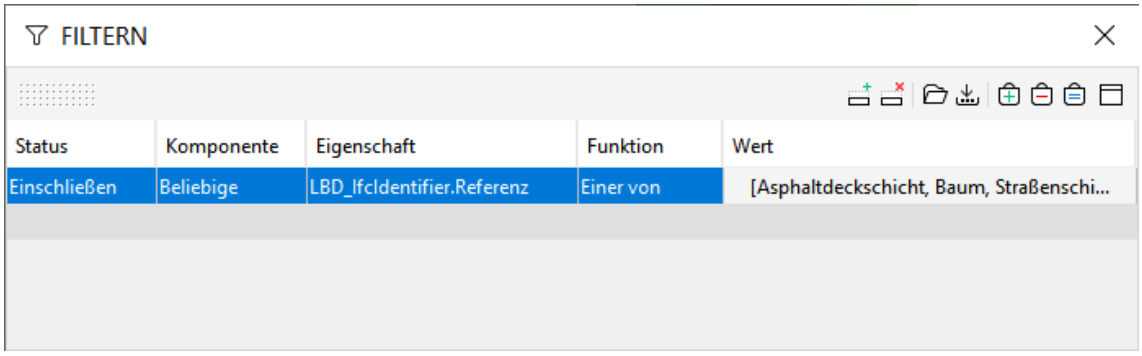


Abbildung 6.10: Filter für alle im Test involvierten Objekte

Bei den RuleSets ist es möglich eine Gruppe an Objekten im Filter zu definieren und eine andere Gruppe über den Standardfilter dem Auswahlkorb zuzuweisen, also hier die Unterscheidung der Straßen und den angrenzenden Elementen. Die Objekte beider Gruppen können nun miteinander auf Distanz geprüft werden. Bei der Verwendung der Ansichten gibt es jedoch nur die Möglichkeit, den Standardfilter für die Zuweisung zum Auswahlkorb zu nutzen. Ersteres hat den Vorteil, dass die beiden Gruppen leichter differenziert werden können, selbst bei einer schlechten Datenqualität. Nachteil ist jedoch, dass es mehr Aufwand benötigt, die beiden Gruppen zu definieren. Die zweite Möglichkeit benötigt nur einen Filter, der alle Objekte enthält, die für den Test benötigt werden. Hierzu wird in der Drop-Down-Liste der Ansicht "Filternäusgewählt und alle Objekte gefiltert, die Teil der Untersuchung sein sollen, daher die Straßenoberfläche, Bäume und Schilder (siehe Abb. 6.10). Um diese wiederum in zwei Gruppen aufzuteilen, muss der Anwender den Eigenschaftssatz und den Wert der Eigenschaft angeben, der die entsprechende Gruppe, hier die Straßenoberfläche, eindeutig definiert. Das heißt, in diesem Fall wäre das "LBD_IfcIdentifieründ Äsphaltdeckschicht"(siehe Abb. 6.11). Dadurch kann das Programm die gefilterten Elemente in zwei Gruppen einordnen und den Test durchführen. Der Vorteil dieser Möglichkeit ist, dass die zu untersuchenden Objekte insgesamt definiert werden und dies sehr schnell und einfach geht. Nachteil ist jedoch, dass die zu untersuchende Gruppe über eine Eigenschaft eindeutig differenzierbar sein muss, was eine hohe Datenqualität voraussetzt.

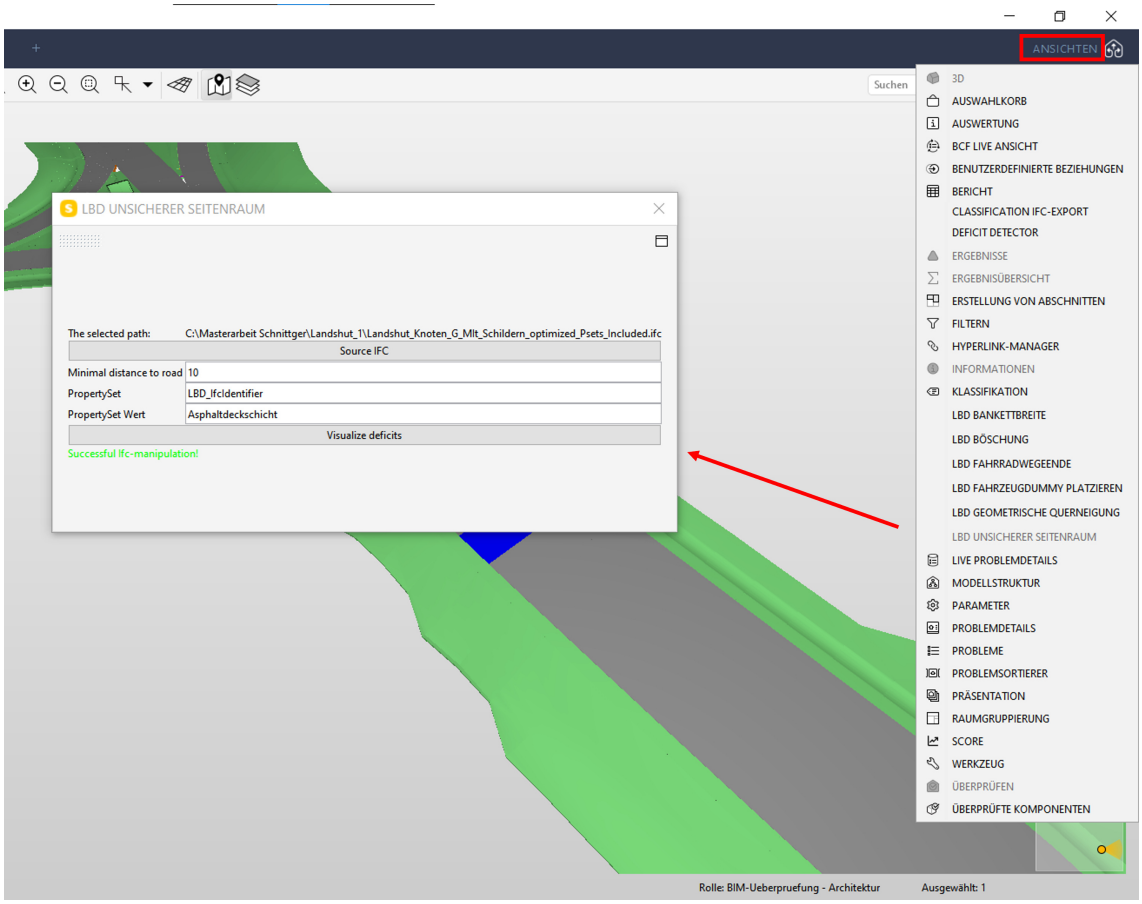


Abbildung 6.11: Solibri Ansicht

Nach der Eingabe des Mindestabstands kann das Programm mit dem Button "Visualize Deficits"gestartet werden. Die Durchlaufzeit hängt von der Anzahl der Konflikte ab. Das Einsetzen der Hinweisobjekte über die Processbuilderfunction ist nämlich relativ langsam und braucht pro Element bis zu 117 Sekunden bei einer Modellgröße von 140 MB. Ist das Programm abgeschlossen erscheint ein grüner Text mit der Nachricht SSuccessfull Ifc-manipulationünd die neue IFC-Datei ist im Ordner der Originaldatei zu finden.

6.3.2 Anzeigen von Radwegenden

Wie bei der Seitenraumprüfung muss auch hier nur ein Filter definiert werden. Bei dieser Regel sind es jedoch die Oberflächen aller Straßen. Damit das Programm die Straßenelemente in Rad- und Fahrzeugverkehr unterscheiden kann, muss der Anwender die Parameter entsprechend der Abbildung 6.12 angeben.

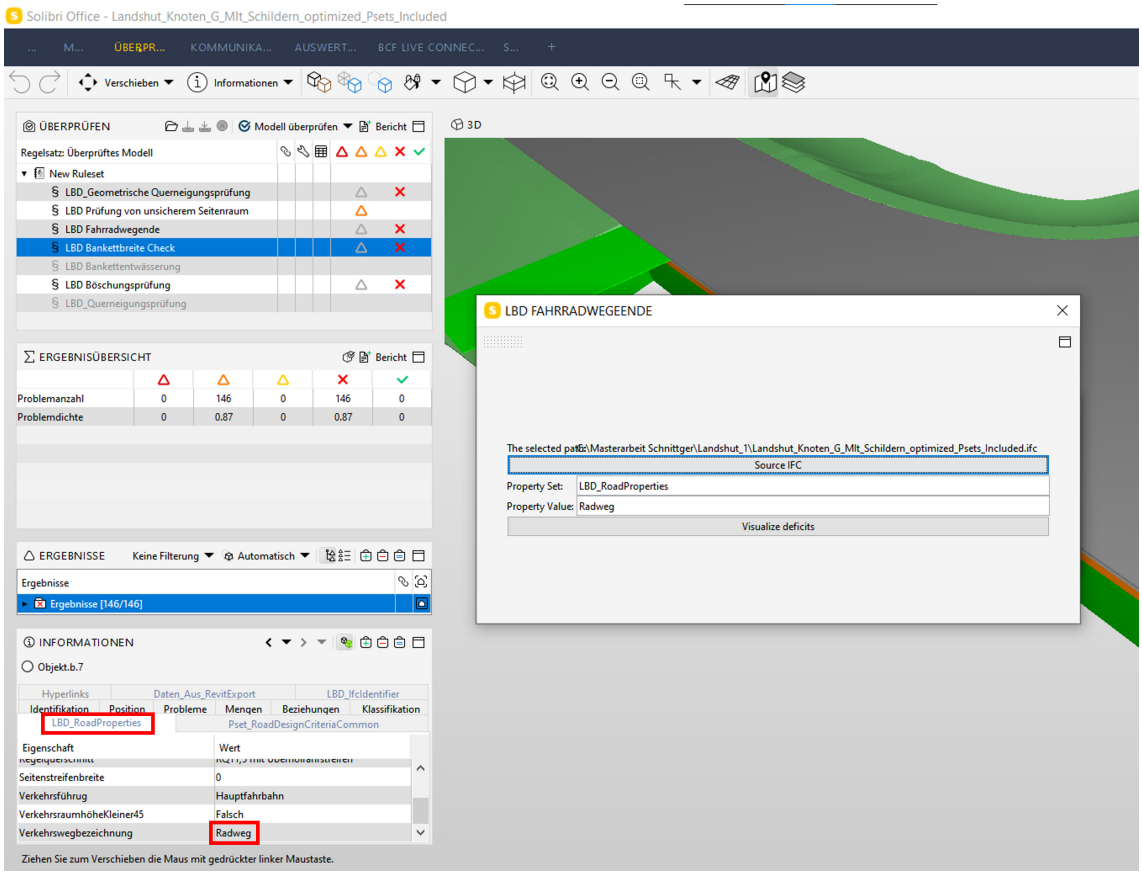


Abbildung 6.12: Filter für Radweg

Gerade bei dieser Regel erscheint das Hinweissymbol passender als das Hervorheben des Elements, da es schließlich nicht gezwungenermaßen fehlerhaft ist, sondern lediglich für eine besonders hohe Aufmerksamkeit auf diese Stelle gesorgt werden soll.

6.3.3 Prüfen der Bankettbreite

Bei der Prüfung der Bankettbreite gibt es zwischen den beiden Varianten einen Unterschied, der aus Leistungsgründen implementiert werden musste. Wie bei den anderen Ansichten muss der Anwender hier die originale IFC-Datei angeben, die Mindestbreite des Banketts definieren und mit dem Filter das Bankettmaterial dem Auswahlkorb zuordnen. Das Programm findet in diesem Fall die gleichen Bankette wie zuvor, nur besteht hierbei das Problem, dass bei entsprechendem Parameter sehr viele Bankette als fehlerhaft eingestuft werden und dementsprechend teilweise über 146 Hinweisobjekte eingesetzt werden müssen. Bei einer durchschnittlichen Laufzeit von 115 Sekunden pro Hinweisobjekt entspricht das über 279 Minuten für alle Objekte. In manchen Fällen mag eine lange Rechenzeit für die Analyse eines Modells gerechtfertigt sein, doch ist sie in diesem Fall nicht angemessen, weshalb die Bedingung eingeführt wurde, dass die Fehler nur angezeigt werden, wenn das Objekt sich in einem Abstand nicht näher als 50 m von anderen Hinweisobjekten befindet. Demnach hat ein Auditor die umliegenden Bankette bei einer Fehlererkennung ebenso zu überprüfen.

6.3.4 Querneigungsprüfung

Ähnlich wie bei der Prüfung der Bankettbreite kommt es auch bei der Querneigungsermittlung zu einer stellenweisen erheblichen Anzahl an fehlerhaften Elementen, was zu einer nicht mehr vernünftigen Menge an Hinweisobjekten führen würde. Auch hier wurde die Prämisse von einem Mindestabstand von 100 m vorgegeben, um die Laufzeit auf ein adäquates Minimum zu setzen. Für die Ausführung der Prüfung muss der Anwender den Begriff für den Eigenschaftssatz eingeben, der den Radius enthält und den, der den Wert des Kurvenradius enthält. In gegebenem Beispiel ist das "Daten_Aus_RevitExportünd Înfra_RadiusFrom(Pset_Desite)". Danach wird mittels des Filters die Asphaltdeckschicht ausgewählt und das Programm gestartet. Wie bei der Prüfung mittels der Rückgabe in Solibri besteht auch hier das Problem, dass nicht klar ist, auf welcher Seite des Elements die Querneigung gemessen wurde und zusätzlich kann in diesem Fall nicht auf die Visualisierung zurückgegriffen werden. In Anbetracht der Tatsache jedoch, dass die Hinweisobjekte ohnehin nicht für alle Fehler erstellt werden können, muss der Auditor den Bereich sowieso manuell überprüfen. Eine fehlerhafte Zuordnung der Querneigung zum Kurvenradius kann demnach einen Mehraufwand bedeuten, da der Konflikt entweder falsch oder schlimmer erst gar nicht detektiert wurde.

6.3.5 Böschungsprüfung

Die Böschungsprüfung ist zur ersten Variante praktisch gleich und unterscheidet sich nur durch den Filter, der nicht direkt im Interface, sondern in der Ansicht zu finden ist. Auch hier bestünde das Problem, dass bei falsch angelegten Böschungen eine zu hohe Anzahl an einzusetzenden Hinweisobjekten die Laufzeit unverhältnismäßig in die Länge ziehe. Da diese Regel jedoch nur einen geplanten Fehler finden kann und einen rein demonstrativen Zweck erfüllt, wurde auf die Anpassung des Programms an dieser Stelle verzichtet.

6.3.6 Car-dummy platzieren

Um Sichtverhältnisse besser nachvollziehen zu können, ist es möglich, einen Fahrzeugdummy zu platzieren. Hierzu wählt der Anwender die Ansicht "LBD Fahrzeugdummy platzierenäus. Hat er das originale IFC-Schema eingelesen, muss er nur noch mit der Auswahlfunktion Solibris die Objekte anklicken, über denen der Dummy platziert werden soll und das Programm starten.

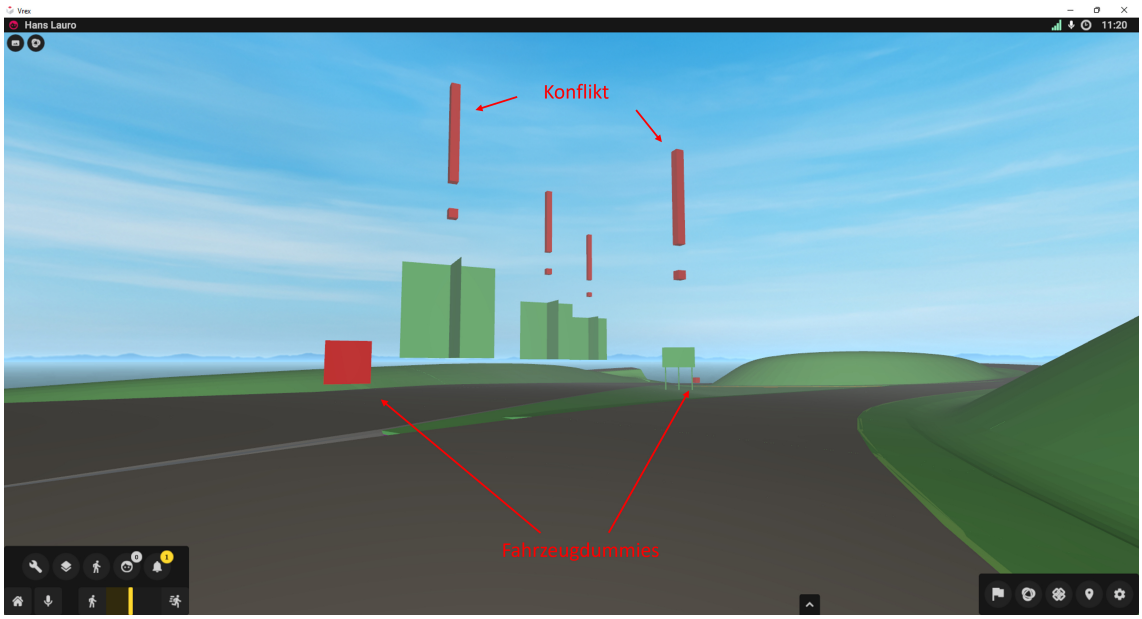


Abbildung 6.13: VRex: Virtuelle Ansicht auf Konflikte und Fahrzeugdummies

Die Ansichten müssen einzeln verwendet und die SSource IFC"nach jeder Veränderung entsprechend angepasst werden, da die Hinweise in einer neuen IFC-Datei mit anderem Namen generiert werden. Die an den Dateinamen angehängten Begriffe geben Auskunft über die ausgeführten Schritte. Um erstellte Modelle in VRex einsehen zu können, muss es zuerst im Webaccount hochgeladen werden. Danach kann es wie bereits erläutert im Viewer geöffnet werden (siehe Abb. 6.13).

6.3.7 Arbeiten in VR

Im Gegensatz zu der Arbeit mit BCF werden die Konflikte in diesem Fall alle gleichzeitig angezeigt und sind "physisch"hervorgehoben durch Ausrufezeichen. So kann durch einen Blick eingeschätzt werden, wo und wie viele Konflikte es im Modell gibt. Um die jeweiligen Konflikte zu untersuchen, muss der User auf die Hinweisobjekte klicken und kann über ihre Eigenschaftssätze Details über den Konflikt auslesen (siehe Abb. 6.14).

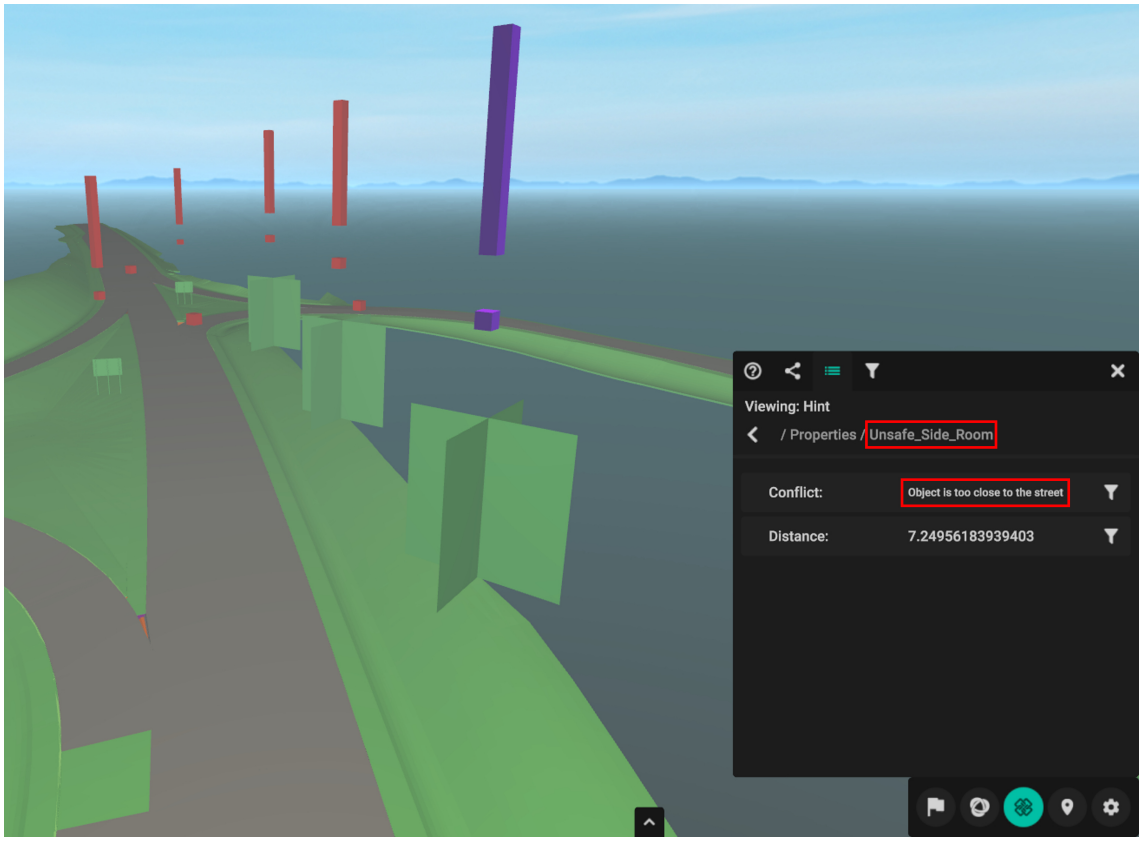


Abbildung 6.14: Konfliktdetails in den Eigenschaftssätzen

6.4 Überführung der Ergebnisse in einen Bericht

Für das Sicherheitsaudit muss der Auditor je Defizit einen Bericht erstellen, der den Mangel erklärt und durch die Kilometrierung lokalisierbar macht. Im Falle der Fehlerrückgabe innerhalb Solibris kann im Überprüfungslayer für jeden Konflikt eine Folie erstellt werden (siehe Abb. 6.15). Diese beinhaltet den Titel, den Blickwinkel zusammen mit einem Snaphot und die Beschreibung des Konflikts, die zusätzlich den Ort im Streckenverlauf enthält. Hierdurch kann der Auditor die einzelnen Defizite durchgehen, entscheiden, welche ein tatsächliches Problem bilden und diese im Bericht aufnehmen (siehe Abb. 6.16). Die fertige Version kann der Auditor in Excel ausgeben lassen und von dort die relevanten Beschreibungen für die Berichterstellung in BAYSIS nutzen.

Bei Anwendung der zweiten Variante werden die Beschreibungen des Konflikts in die Eigenschaftssätze der Hinweisobjekte geladen. Um eine Auflistung dieser zu erhalten, müssen sie über den Filter gesucht und ausgewählt werden. Sind die Hinweisobjekte gelistet, können die Konfliktbeschreibungen und die charakteristischen Werte für die weitere Nutzung kopiert werden.

An dieser Stelle sei angemerkt, dass erst sehr spät aufgefallen ist, dass die Kilometrierung beim Exportvorgang aus Civil3D nicht überliefert wurde und diese nicht wie der Kurvenradius wiederhergestellt werden konnte. Diesen Fehler zu korrigieren hätte einen erheblichen Mehraufwand bedeutet, der durch den Zweck an dieser Stelle nicht gerechtfertigt wurde.

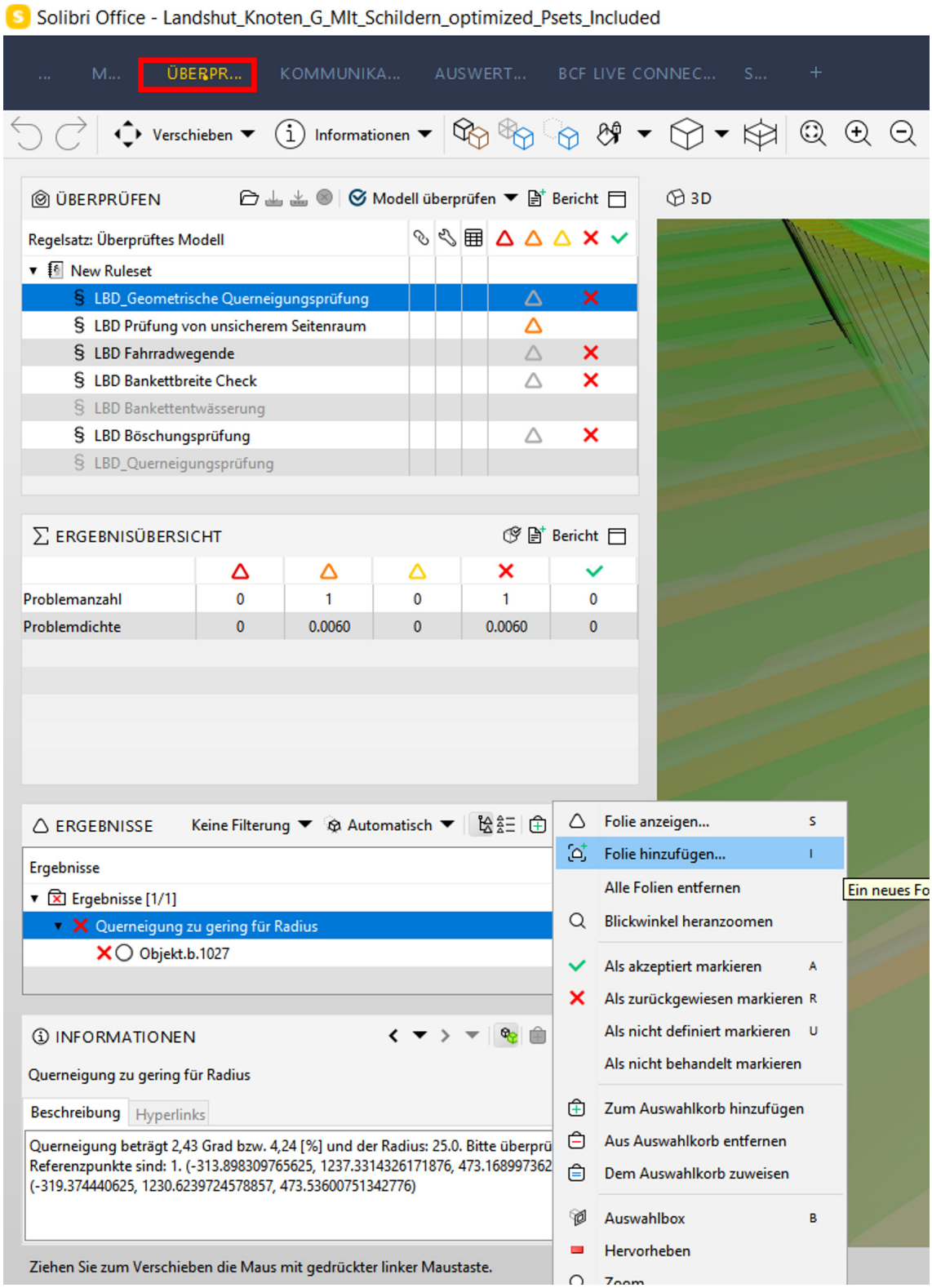


Abbildung 6.15: Erstellen einer Folie für den Bericht

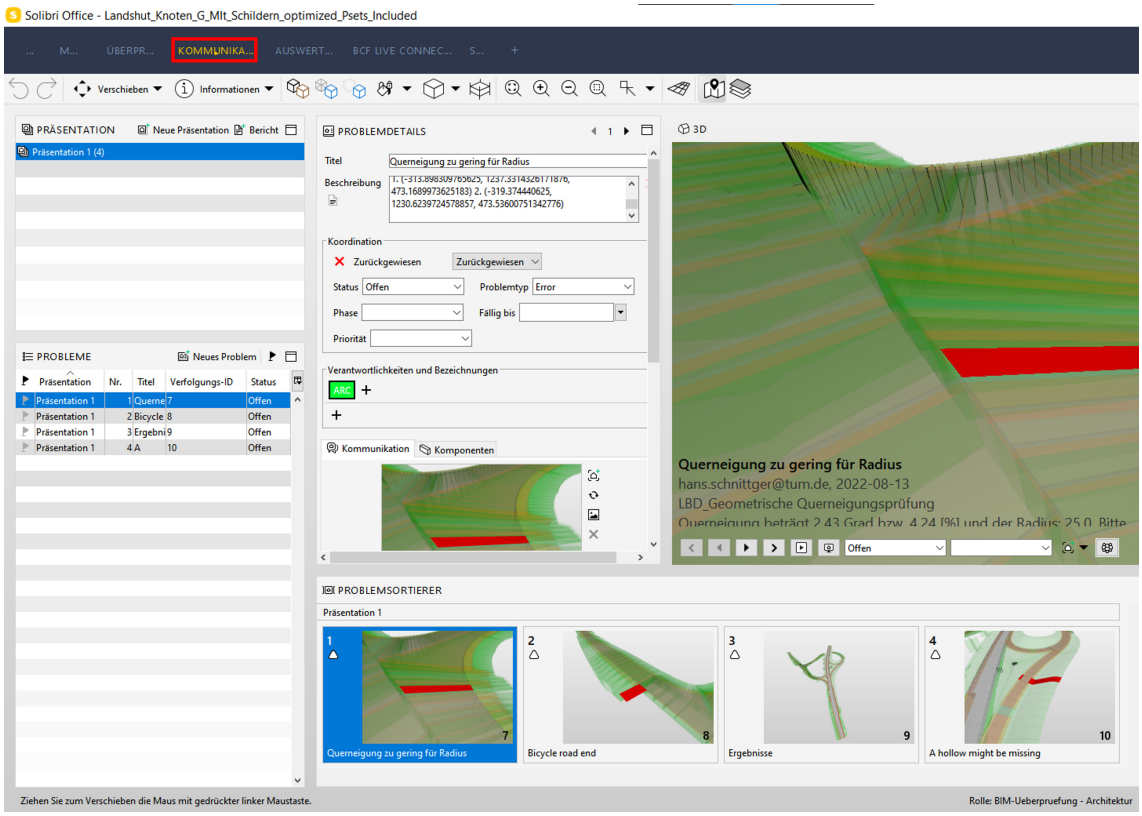


Abbildung 6.16: Berichterstellung mit Konflikten

6.5 Auswertung

Generell wurden die vorgestellten Überprüfungstools als sehr positiv wahrgenommen, da eine unterstützende Anwendung als prozessbeschleunigend und arbeitserleichternd eingestuft wurde. Besonders positiv wurde das räumliche Verständnis hervorgehoben, dass durch die Anwendung von VR gewonnen werden konnte. Die Auditoren zeigten großes Interesse an der Digitalisierung ihrer Arbeit und hatten bereits ein grundlegendes Verständnis für die Möglichkeiten und Grenzen der Automatisierung. Es wurde jedoch auch angemerkt, dass die notwendige Vorarbeit, also das Eingeben von Parametern und das Einstellen von Filtern nicht zu komplex beziehungsweise aufwendig sein darf, da sich sonst der Vorteil der Automatisierung wieder aufheben würde. Dieser hängt zum großen Teil von einer guten Modellierung ab, die konsequent die Informationsanforderungen einhält. So lassen sich Filter einfacher konfigurieren, wenn die Identitäten von Elementgruppen eindeutig bestimmt sind, die Straße also einerseits unter IfcRoad als Gesamtentität, aber andererseits die Straßenoberfläche als Einzelentität erfasst werden kann. Die Informationsanforderung kann hierbei direkt als IfcEntität oder als Eigenschaftssatz vorhanden sein, was im ersten Fall voraussetzt, dass die entsprechende Software das jeweilige IFC-Schema unterstützt und in beiden Fällen, dass die Exportvorgänge mit den korrekten Einstellungen für die Eigenschaftssätze ausgeführt werden.

6.5.1 Konfliktrückgabe innerhalb Solibris

Die Überprüfungen, die den Konflikt innerhalb Solibris anzeigen, haben den eindeutigen Vorteil, dass sie sehr schnell sind und selbst bei sehr großen Modellen meist unter einer Minute liegen. Ein weiterer Vorteil ist die Möglichkeit der Arbeit mit dem BCF-Manager, da hierbei die gefundenen Konflikte mitsamt der Konfliktbeschreibung in den BCF-Server geladen werden können und dadurch in VRex verfügbar sind. An dieser Stelle wurde angemerkt, dass eine Überführung der Konfliktbeschreibungen in BAYSIS die Arbeit zusätzlich erleichtern würde. Die Beschreibungen, die bei der Konflikterkennung generiert werden, könnten also bei der Berichterstellung als Satzfragmente dienen. Der Auditor müsste diese nur noch in den korrekten Kontext setzen. Vorstellbar wäre hierbei beispielsweise eine Solibriberichtausgabe im Comma-Separated Values (CSV)-Format, welche dann in BAYSIS hochgeladen und dort weiter bearbeitet werden könnte. Eine andere Möglichkeit wäre natürlich auch, dass BAYSIS BCF-kompatibel wird und so direkt auf die Konflikte zugreifen kann.

Weiterhin ermöglichte die Verwendung zweier Filter die Entwicklung eines stabileren Algorithmus, da durch zwei Elementgruppen, die über mehrere Attribute eindeutig ihrer Gruppe zugeordnet werden, statt nur mittels eines Attributs aus den Eigenschaftssätzen, eine zuverlässigere Gruppendefinition gebildet werden kann. Fehler im Modell auf alphanumerischer Ebene können dadurch leichter kompensiert werden.

6.5.2 Konfliktrückgabe über IFC-Manipulation

Die Überprüfungen, die Hinweiseobjekte an den Konfliktstellen erzeugen, sind relativ langsam und sollten daher mit Bedacht verwendet werden. Sind die Filter korrekt eingestellt und ist abzusehen, dass die Anzahl der Konflikte sich im Rahmen des sinnvollen halten, können sie vorteilhaft verwendet werden, doch ist eine Bedingung eingestellt, die eine sehr hohe Anzahl an Konflikten erzeugt und hat die Regel nicht wie bei der Bankettprüfung eine Mindestabstandsprämisse, dann werden so viele Hinweisobjekte eingefügt, dass der Prozess eine vernünftige Zeitspanne übersteigen kann und daher eventuell vorzeitig abgebrochen werden müsste. Die Hinweisobjekte werden als IfcBuildingElementProxy-Klassen erstellt, da es ansonsten keine passende Klasse für die Objekte gab. Sie können, daher nur durch ihren Eigenschaftssatz eindeutig gefiltert werden.

Der Vorteil der IFC-Veränderung liegt jedoch in der Plattformunabhängigkeit und in der Übersichtlichkeit. So kann die erzeugte Datei mittels jedes IFC-kompatiblen Tools geöffnet werden und ist somit nicht auf eine BCF-kompatible Anwendung wie VRex angewiesen, sondern kann auch mit visuell ansprechenderen Applikationen geöffnet werden. Der Effekt eines auffallenden Objekts, das in VR einen Konflikt anzeigt, kann um einiges eindrucksvoller sein und sich daher gut für eine Demonstration eignen. Weiterhin sind alle Konflikte auf einmal lokalisierbar, was eine große Übersichtlichkeit über das Modell liefert und eine schnelle Einschätzung ermöglicht.

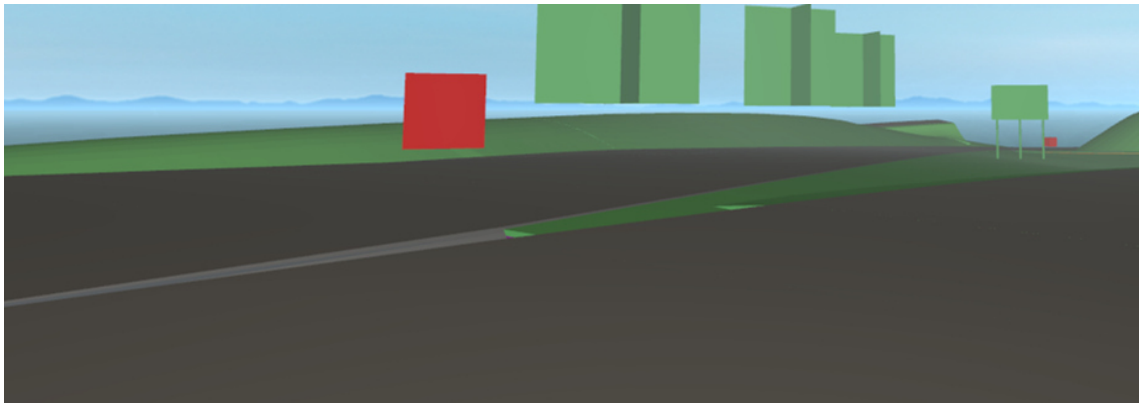


Abbildung 6.17: Fahrzeugdummies als heranfahrende PKW

Ein weiterer großer Nutzen besteht in der Möglichkeit Fahrzeugdummies einzufügen, um Sichtverhältnisse besser nachvollziehen zu können (siehe Abb. 6.17). Die Arbeit in VR dient in erster Linie dazu, dem Auditor eine realistische Einsicht auf einen Streckenabschnitt geben zu können. Hierbei spielen räumliche Dimensionen eine zentrale Rolle und ein Objekt als Referenzpunkt erzeugen zu können, kann in vielen Fällen hilfreich sein. So ist es für einen Auditor sehr relevant zu verstehen, ob bei einer Einfahrt von einer untergeordneten Strecke in eine übergeordnete, Schilder oder Bewuchs die Sicht auf heranfahrende Fahrzeuge versperrt. Selbes gilt in Bezug auf Leitplanken, die eventuell die Sicht auf vorausfahrenden Verkehr im Kurvenbereich versperren und so Auffahrunfälle provozieren. Mittels des Lasers können Abstände zu Objekten gemessen werden und helfen, Mindestabstände besser einzuhalten. Dieser Punkt des Visibilitätschecks wurde von den Auditoren als einer der bedeutendsten der VR-Anwendung genannt.

Weiterhin kann die Technik der IFC-Manipulation genutzt werden, um einfache Objekte wie Würfel und Quader von variabler Form zu erzeugen, die beispielhafte Referenzobjekte bilden. Diese minimalen Modelliermöglichkeiten, können helfen den Einfluss von noch nicht in die Planung einbezogenen, aber absehbaren Veränderungen, wie Bewuchs und andere Umwelteinflüsse abschätzen zu können.

7. Diskussion und Zusammenfassung

Im letzten Kapitel werden die Arbeit und die Ergebnisse zusammengefasst und die potenziellen Verbesserungsmöglichkeiten in der modellbasierten Unterstützung des Sicherheitsauditors hervorgehoben. Durch die gesammelte Erfahrung mit den Pilotprojekten der LBD wird im Kontext des aktuellen Stands der Technik anschließend eine Aussicht auf zukünftige Entwicklungen und Möglichkeiten gegeben.

7.1 Zusammenfassung

Das Ziel dieser Arbeit war es zu untersuchen, wie der Auditor mittels eines BIM und der Anwendung von VR beim Sicherheitsaudit unterstützt werden kann. Dies wurde erfolgreich demonstriert, indem ein Teilmodell des Projektes "B299, dreistufiger Ausbau Geisenhausenßo modifiziert wurde, dass es den notwendigen alphanumerischen und geometrischen Informationsgehalt enthält, um mittels automatisierter Konformitätsprüfungen auf Sicherheitsdefizite verifiziert werden zu können. Es wurde detailliert auf den Prozess der Modifikation eingegangen. Hierzu wurde gezeigt, wie in Revit das IFC-Schema angepasst und mittels einer Mapping-Tabelle die korrekten Attribute in das neue Modell exportiert wurden. Im Anschluss wurde die Attribuierung entsprechend der zuvor definierten alphanumerischen Daten in Civil3D erläutert und das Beheben der beim Export entstanden Modellfehler demonstriert. Danach wurden die mittels der Solibri API in Java entwickelten Überprüfungsprogramme und das in Python geschriebene Programm, welches IfcOpenShell zur Modellmodifikation nutzt, erklärt. Im vorletzten Kapitel wurden zusammen mit den Sicherheitsauditoren in einer virtuellen Begehung des vorbereiteten Modells mittels der VRex-Anwendung die entwickelten Tests angewendet und bewertet.

7.2 Ausblick

Die Arbeit hat gezeigt, dass es noch einige Hürden gibt, um den Sicherheitsauditor optimal unterstützen zu können. Die Prozesse stoßen in vielen Bereichen an ihre Grenzen. Die Erkenntnisse aus dieser Arbeit wurden daher genutzt, um die notwendigen Optimierungsschritte zu ermitteln und möglichst prägnant aufzulisten. Im ersten Abschnitt wird das Aufgabenfeld der LBD abgehandelt, danach wird sich auf die generelle technische Entwicklung konzentriert:

- Einheitliches IFC-Schema in allen Teilmodellen.
- Entwicklung eines unternehmensinternen Standards für:
    - geometrische Informationsbedarfstiefe

    - alphanumerische Informationsbedarfstiefe
    - Integrität der IFC-Semantik, um dessen Funktionalitäten nutzen zu können.
- Entwickeln einer entsprechenden Checkliste oder eines Überprüfungsprogramms für die Bewertung der Modellqualität.
- Klassifizierung der Modellelemente so weit wie möglich entsprechend des aktuellen IFC-Schemas auch in den Eigenschaftssätzen, um eine saubere Überleitung zu ermöglichen (Beispiel: LBD_Identifier: IfcRoadPart).
- Konstante Verbesserung und Aktualisierung der Überprüfungsprozesse und IFC-Standards, um immer mehr Abläufe zu standardisieren und zu automatisieren.
- Softwareanbieter wirtschaftlich motivieren, neueste IFC-Schemas in allen Anwendungen des Arbeitsablaufs einheitlich zu implementieren.
- Gegenüberstellung des konventionellen und des "digitalenÄudits, um Praxisbezug zu dokumentieren und Optimierungsmöglichkeiten zu erkennen.
- Aufbau einer internen Expertise für das automatisierte Sicherheitsaudit und BIM.
- Anlegen einer Bibliothek an standardisierten Regeln, die ihren Nutzen in der Praxis erwiesen haben.
- Anstreben einer BCF-Kompatibilität von BAYSIS für den Entwurf des Auditberichts.

Weiterhin gibt es Bereiche, auf die die LBD keinen direkten Einfluss hat, die aber generell notwendig sind, um den Arbeitsfluss mit IFC und BIM zu verbessern. Im Folgenden wird auf die Punkte eingegangen, die im Verlauf zu Konflikten geführt haben und Vorschläge gemacht, wie diese zu beheben sind.

Da Civil3D und Revit beide Teil der Autodesk-Familie sind, wäre eine gewisse Konformität zu erwarten, doch ist der Im- und Export unterschiedlich aufgebaut. Hier die Punkte, die vereinheitlicht werden sollten:

- Revit arbeitet mit einem Bauteillistensystem und lässt den Export von Eigenschaftssätzen nur mit den korrekten Einstellungen oder mittels einer zuvor definierten Mapping-Tabelle zu.
- Civil3D wiederum exportiert die Eigenschaftssätze ohne besondere Einstellungen.
- Der Import derselben IFC 4 Datei lässt sich in Revit über die Verknüpfungsfunktion sichtbar machen, was wiederum in Civil3D nicht möglich ist.
- Beide Anwendungen ermöglichen nicht das Filtern nach Eigenschaftssätzen und deren Werte. Für Civil3D gibt es jedoch ein kommerzielles Add-in, das dies ermöglicht (PSED documentation von Camilion).
- Revit stellt die hinzugefügten Hinweisobjekte anders als alle anderen Anwendungen relativ zum Projektursprung dar.

- Civil3D und Revit erzeugen unterschiedliche Exportfehler bei der gleichen Datei.

Hätte Revit also eine Funktion, um die Objekte nach ihren Eigenschaftssätzen entsprechend dem IFC-Standard zu filtern und neue hinzuzufügen, wäre die Überführung in Civil3D erst gar nicht nötig gewesen. Die Lösung, das Modell in Revit zu bearbeiten, um das Schema in IFC 2x3 zu ändern, damit es in Civil3D öffenbar ist, war demnach eine Notlösung. Weitere Punkte sind:

- Die Beziehungen zwischen den Objekten müssen in den IFC-Dateien vorhanden sein. Diese existieren in den Anwendungen innerhalb der nativen Umgebung durchaus (Beispiel: Ashphaltdeckschicht berührt Bankett) werden aber beim Export beziehungsweise Import nicht korrekt übersetzt.
- Einheitliche Implementierung des IFC 4.3 Schemas in den Anwendungen mit allen Funktionalitäten.
- Vereinheitlichung der Interpretation von IfcAlignment.
- Kompatibilitätsprobleme von Solibri mit Windows 11 lösen, da dies zu regelmäßigen Abstürzen der Anwendung führte.

7.2.1 Fazit

Die Implementierung von Automatisierungsprozessen und BIM im generellen wird anfänglich einen Mehraufwand bedeuten. Der Fokus sollte hierbei wie geschildert auf einer Etablierung eines Modellstandards liegen. Diese Arbeit dient dementsprechend als eine erste Vorlage für die Entwicklung eines solchen LBD-Standards.

Die Fallstudie hat jedenfalls gezeigt, dass es aufseiten der Auditoren ein großes Interesse an Automatisierungen gibt, was eine wichtige Voraussetzung ist, da der Prozess dorthin ihre Mitarbeit verlangen wird, indem sie in der Thematik geschult werden und ein Grundverständnis für die zugrundeliegende Semantik des IFC-Schemas aufbauen, um nachvollziehen zu können, was beim Ablauf der Prüfungen passiert und die Modellqualität bewerten zu können. Aus den in dieser Arbeit entwickelten Regeln zusammen mit der Erfahrung der Auditoren sollten daher eine Auswahl an Semiautomatisierungsprozessen abgeleitet werden, die dann in der Praxis geprüft werden. Hierdurch kann mit der Zeit eine Bibliothek an Regeln angelegt werden.

Ob die Arbeit in VR den damit zusammenhängenden Aufwand und Hardwarebedarf rechtfertigt, hängt von der Bedeutung des räumlichen Verständnisses ab, das innerhalb der virtuellen Realität besser entwickelt werden kann, als vor dem bloßen Bildschirm. Die virtuelle Begehung mit den Auditoren hat gezeigt, dass es nicht der Dauerbetrieb sein wird, der den großen Nutzen bringen wird, sondern die punktuellen Stellen in einem Modell, die einer genaueren Untersuchung bedürfen und stark von der subjektiven Einschätzung des Auditors abhängen. In Situationen, bei denen beispielsweise eine Kurve aus Sicht des Fahrers nachempfunden werden muss, ist die Verwendung von VR vorteilhaft, da

hierbei der dynamische Aspekt des Fahrens besser berücksichtigt werden kann. Ebenso vermittelt die virtuelle Realität die Dimensionen besser und Objekte, die in der Planung als klein oder kurz empfunden werden, zeigen in VR ihren wahren Umfang.

A. Programme

```
@Override
32      public Collection<Result> check(Component component, ResultFactory
    resultFactory) {

34          Collection<Result> results = new ArrayList<>();

36          //The Selection basket defines what is street
            Set<com.solibri.smc.api.model.Component> modelStreet = SMC.
    getSelectionBasket().get();
38          for (Component checkedStreetPart : modelStreet){
                double distanceObjects = checkedStreetPart.distance(component).get
    ();
40              if(checkedStreetPart.distance(component).isPresent() &&
    checkedStreetPart.distance(component).get() < securityDistance.getValue())
    {

42                  String name = "Unsafe Side Room";
                    String description = "Object is too close to the street.
    Distance: " + df.format(distanceObjects) + "m";
44
                    Result result = resultFactory
46                          .create(name, description)
                            .withInvolvedComponent(component)
48                          .withVisualization(visualization -> {
                                visualization.addComponentWithColor(component,
    ColorRed);
50                              visualization.addComponentsWithColor(modelStreet,
    ColorTransGreen);
                            });
52
                    results.add(result);
54                  return results;
                }}
56          return results;
        }
```

Algorithmus A.1: LBD Prüfung von unsicherem Seitenraum

```
188         Collection<Vector3d> getCornerPoints(List<Vector2d>convexHullAsphalt,
    Collection<Vector3d> pointZ){

190         Collection<Vector2d>cornerPoints2d = new ArrayList<>();
            Collection<Vector3d>cornerPoints3d = new ArrayList<>();
192
            List<Vector2d> convexHullAsphaltClosed = new ArrayList<>(
    convexHullAsphalt);
194         convexHullAsphaltClosed.add(convexHullAsphalt.get(0));
            convexHullAsphaltClosed.add(convexHullAsphalt.get(1));
196
```

```
        Vector2d firstDirection = convexHullAsphaltClosed.get(1).subtract(
    convexHullAsphaltClosed.get(0));
198     Vector2d referenceDirection = firstDirection;

200     for (int i = 0; i < convexHullAsphaltClosed.size()-1; i++) {

202         firstDirection = convexHullAsphaltClosed.get(i + 1).subtract(
    convexHullAsphaltClosed.get(i));

204         double angle = firstDirection.angle(referenceDirection) *
    57.295779513;

206         if (!firstDirection.equals(referenceDirection) && angle > 15.0) {
    //goes through the polygon points and identifies corner points when the
    angle between them indicates that
                cornerPoints2d.add( (convexHullAsphaltClosed.get(i)));
208         }
            referenceDirection=firstDirection;
210

        }
212     for (Vector2d point2D: cornerPoints2d){
            for (Vector3d point3D: pointZ){
214             if(point2D.equalsWithTolerance(point3D.to2dVector(),0.0001)) {
                    cornerPoints3d.add(point3D);
216                 break;
                }
218         }
        }
220

        return cornerPoints3d;
222 }
```

Algorithmus A.2: getCornerPoints()

```
        @Override
36  public Collection<Result> check(Component component, ResultFactory
    resultFactory) {

38      Collection<Result> results = new ArrayList<>();

40      Collection<PropertySet> psetCompany = component.getPropertySets(
    propertySetString.getValue());
        Collection<Component> all = SMC.getModel().getComponents(
    ComponentFilter.ACCEPT_ALL);
42

        for (PropertySet pCs : psetCompany){
44          Set<Property<?>> propertiesCompany = pCs.getProperties();

46          Optional<?> dataMulde = null;
            Optional<?> dataEinschnitt = null;
48

            for(Property <?> propertyC:propertiesCompany) {
50              String nameC = propertyC.getName();
```

```
                if (Objects.equals(nameC, propertySetMulde.getValue()))
                    dataMulde = propertyC.getValue();

                if (Objects.equals(nameC, propertySetEinschnitt.getValue()))
                    dataEinschnitt = propertyC.getValue();
            }
            if (dataEinschnitt.isPresent() && dataMulde.isPresent()) {
                if ((boolean)dataEinschnitt.get() && !(boolean)dataMulde.get())
    {

                    String name = "A hollow might be missing";
                    String description = "Verify whether the hollow is there.";

                    Result result = resultFactory
                            .create(name, description)
                            .withInvolvedComponent(component)
                            .withVisualization(visualization -> {
                                visualization.addComponentWithColor(component,
    ColorRed);
                                visualization.addComponentsWithColor(all,
    ColorTransGreen);
                            });

                    results.add(result);
                    return results;
                }
            }
        }

        return results;
    }
```

Algorithmus A.3: LBD Böschungsprüfung

```
# Define and associate the material
material = ifcfile.createIfcMaterial("hint material")
material_layer = ifcfile.createIfcMaterialLayer(material, 0.0, None)
material_layer_set = ifcfile.createIfcMaterialLayerSet([material_layer], None)
material_layer_set_usage = ifcfile.createIfcMaterialLayerSetUsage(
    material_layer_set, "AXIS2", "POSITIVE", -0.1)
ifcfile.createIfcRelAssociatesMaterial(create_guid(), owner_history,
    RelatedObjects=[hint], RelatingMaterial=material_layer_set_usage)

quantity_values = [
    ifcfile.createIfcQuantityLength("Length", None, None, 0),
    ifcfile.createIfcQuantityArea("Area", None, None, 0),
    ifcfile.createIfcQuantityVolume("Volume", None, None, 0)
]

element_quantity = ifcfile.createIfcElementQuantity(create_guid(),
    owner_history, "BaseQuantities", None, None, quantity_values)
ifcfile.createIfcRelDefinesByProperties(create_guid(), owner_history, None,
    None, [hint], element_quantity)
```

Algorithmus A.4: Hint material definition

B. CD

Die beigelegte CD enthält:

- Die digitale Version dieser Arbeit
- Den Quellcode der Regeln, der Ansichten sowie das Pythonscript
- Die jar-file für die Installation der Plug-ins in Solibri
- Die Tabelle mit den alphanumerischen Daten
- Die Modelle:
  - Des Testentwurfs
  - Das originale Teilmodell des Landshuter Pilotprojekts (B299)
  - Das Teilmodell nach dem Export aus Revit
  - Das Teilmodell nach dem Export aus Civil3D mit den Eigenschaftssätzen und Darstellungsfehlern
  - Das Teilmodell nach dem Export aus Civil3D mit den Eigenschaftssätzen ohne Darstellungsfehler
  - Das Teilmodell mit den Hinweisobjekten für die Sicherheitsdefizite

Literaturverzeichnis

AMTSBLATT. (1906). Amtsblatt der Königlichen Regierung zu Cassel. *Amtsblatt der Königlichen Regierung zu Cassel*, *38*(38), 313–329.

AMTSBLATT. (1910). Bekanntmachung, den Vollzug der Verordnung über den Verkehr mit Kraftfahrzeugen vom 3. Februar 1910 betr. *Ministerial-Amtsblatt Königreich Bayern*, *12*(12), 195–218.

APSTEX. (2022). *The apstex IFC Framework*. Verfügbar 22. April 2022 unter http://www.apstex.com/

BMVBS. (2012). *Richtlinien zum Planungsprozess und für die einheitliche Gestaltung von Entwurfsunterlagen im Straßenbau* (Techn. Ber. FGSV-Nr.: 2070). BMVBS - Bundesministerium für Verkehr, Bau und Stadtentwicklung. Bundesministerium für Digitales und Verkehr, Invalidenstraße 44, D-10115 Berlin.

BMVI. (2015). *Stufenplan Digitales Planen und Bauen*. Verfügbar 22. April 2022 unter https://www.bmvi.de/SharedDocs/DE/Publikationen/DG/stufenplan-digitales-bauen.pdf?__blob=publicationFile

BMVI. (2019a). *Handreichung BIM Fachmodelle und Ausarbeitungsgrad*. Verfügbar 25. April 2022 unter https://bim4infra.de/wp-content/uploads/2019/07/BIM4INFRA2020_AP4_Teil7.pdf

BMVI. (2019b). *Allgemeines Rundschreiben Straßenbau Nr. 4/2019* (Techn. Ber. StB 11/7122.1/4/2985041). Bundesministerium für Verkehr und digitale Infrastruktur. Robert-Schuman-Platz 1, 53175, Bonn.

BOHLIN, N. I. (1959). SE1101987XA. https://worldwide.espacenet.com/patent/search?q=pn%5C%3DDE1101987B

BORRMANN, A., KÖNIG, M., KOCH, C., & BEETZ, J. (2020). *Building Information Modeling - Technologische Grundlagen und industrielle Praxis*. Springer Vieweg.

BUILDINGSMART. (2022a). *Core data schemas*. Verfügbar 19. April 2022 unter https://standards.buildingsmart.org/IFC/DEV/IFC4_2/FINAL/HTML/

BUILDINGSMART. (2022b). *Domain specific data schemas*. Verfügbar 19. April 2022 unter https://standards.buildingsmart.org/IFC/DEV/IFC4_2/FINAL/HTML/schema/chapter-7.htm

BUILDINGSMART. (2022c). *IfcProductExtension*. Verfügbar 20. April 2022 unter https://standards.buildingsmart.org/IFC/DEV/IFC4_2/FINAL/HTML/

BUILDINGSMART. (2022d). *IfcRoot*. Verfügbar 19. April 2022 unter https://standards.buildingsmart.org/IFC/RELEASE/IFC4/ADD1/HTML/schema/ifckernel/lexical/ifcroot.htm

BUILDINGSMART. (2022e). *Resource definition data schemas*. Verfügbar 20. April 2022 unter https://standards.buildingsmart.org/IFC/DEV/IFC4_2/FINAL/HTML/

CHEN, K. (2015). *Creating a simple wall with property set and quantity information*. Verfügbar 13. September 2022 unter https://academy.ifcopenshell.org/posts/creating-a-simple-wall-with-property-set-and-quantity-information/

CMS. (2014). *IfcTunnel*. Verfügbar 19. April 2022 unter https://www.cms.bgu.tum.de/de/17-research-projects/100-ifctunnel-de

DIN. (2020). *Bauwerksinformationsmodellierung - Informationsbedarfstiefe* (Techn. Ber. DIN EN 17412-1). DIN.

DSGS. (2022). *Die Anfänge der Fahrbahnmarkierung*. Verfügbar 18. August 2022 unter https://dsgs.de/historie.html

DVR. (2022). *Organisation*. Verfügbar 13. September 2022 unter https://www.dvr.de/ueber-uns/organisation

DVW. (2022). *Historie - Deutsche Verkehrswacht*. Verfügbar 13. September 2022 unter https://deutsche-verkehrswacht.de/uber_uns/historie/

EU-Parlament. (2008). *RICHTLINIE 2008/96/EG DES EUROPÄISCHEN PARLAMENTS UND DES RATES vom 19. November 2008 über ein Sicherheitsmanagement für die Straßenverkehrsinfrastruktur* (Techn. Ber. 2008/96/EG). Amtsblatt der Europäischen Union. 2 rue Mercier, L-2985 Luxemburg.

FGSV. (2002). *Empfehlungen für das Sicherheitsaudit von Straßen* (Techn. Ber. FGSV-Nr.: 250). Forschungsgesellschaft für Straßen- und Verkehrswesen, Arbeitsgruppe Verkehrsführung und Verkehrssicherheit. FGSV e. V., An Lyskirchen 14, 50676 Köln.

FGSV. (2006). *Empfehlungen zum Schutz vor Unfällen mit Aufprall auf Bäume* (Techn. Ber. FGSV-Nr.: 250). Forschungsgesellschaft für Straßen- und Verkehrswesen, Arbeitsgruppe Verkehrsführung und Verkehrssicherheit. FGSV e. V., An Lyskirchen 14, 50676 Köln.

FGSV. (2008). *Richtlinien für integrierte Netzgestaltung* (Techn. Ber. FGSV-Nr.: 121). Forschungsgesellschaft für Straßen- und Verkehrswesen, Arbeitsgruppe Verkehrsführung und Verkehrssicherheit. FGSV e. V., An Lyskirchen 14, 50676 Köln.

FGSV. (2010). *Empfehlungen für Radverkehrsanlagen* (Techn. Ber. FGSV-Nr.: 284). Forschungsgesellschaft für Straßen- und Verkehrswesen, Arbeitsgruppe Verkehrsführung und Verkehrssicherheit. FGSV e. V., An Lyskirchen 14, 50676 Köln.

FGSV. (2012a). *Merkblatt zur Örtlichen Unfalluntersuchung in Unfallkommissionen* (Techn. Ber. FGSV-Nr.: 316/1). Forschungsgesellschaft für Straßen- und Verkehrswesen, Arbeitsgruppe Verkehrsführung und Verkehrssicherheit. FGSV e. V., An Lyskirchen 14, 50676 Köln.

FGSV. (2012b). *Richtlinien für die Anlage von Landstraßen* (Techn. Ber. StB 11/7122.3/4-RAL-1739728a). Forschungsgesellschaft für Straßen- Verkehrswesen. An Lyskirchen 14, 50676, Köln.

FGSV. (2015). *Handbuch für die Bemessung von Straßenverkehrsanlagen* (Techn. Ber. FGSV-Nr.: 299). Forschungsgesellschaft für Straßen- und Verkehrswesen, Arbeitsgruppe Verkehrsführung und Verkehrssicherheit. FGSV e. V., An Lyskirchen 14, 50676 Köln.

Focus. (2020). *Als der ADAC noch Guerilla-Aktionen unternahm: Die Geschichte des Verkehrsschilds*. Verfügbar 17. August 2022 unter https://www.focus.de/auto/ratgeber/unterhaltung/5x-meilensteine-in-der-geschichte-der-verkehrszeichen-vom-zarten-pflaenzchen-bis-zum-wald_id_12196505.html

GERLACH, J. (2004). Sicherheitsaudits von Stadtstraßen - Stärkung der Verkehrssicherheitsbelange in der Straßenplanung. *KommunalPraxis spezial - Kommunale Verkehrsüberwachung*, (2).

IFCOPENSHELL. (2022). *IfcOpenShell*. Verfügbar 22. April 2022 unter http://www.ifcopenshell.org/

JAUD, S., ESSER, S., BORRMANN, A., WIKSTRÖM, L., MUHIC, S., & MIRTSCHIN, J. (2021). *A critical analysis of linear placement in IFC models*. TUM.

KHEMLANI, L. (2005). *CORENET e-PlanCheck: Sinapores's Automated Code Checking System*. Verfügbar 27. Mai 2022 unter https://www.aecbytes.com/feature/2005/CORENETePlanCheck.html

KLEMT-ALBERT, P. K., BERGMANN, P. M., KÖHNCHE, M., NEUBAUR, K., & RAHEBI, H. (2021). *BIM-Leitfaden* (Techn. Ber. Version 2.0). Bayerisches Staatsministerium für Wohnen, Bau und Verkehr. Franz-Josef-Strauß-Ring 4, 80539 München.

KUEHN, D. (2019). *Fußgänger Radfahrer*.

LAAKSO, M., & KIVINIEMI, A. (2012). THE IFC STANDARD - A REVIEW OF HISTORY, DEVELOPMENT, AND STANDARDIZATION. *ITcon - Journal of Information Technology in Construction*, *17*(1874-4753), 141–155.

LKSOFTWARE. (2022). *JSDAI overview*. Verfügbar 22. April 2022 unter https://jsdai.net/overview

MAZS. (2019). *Merkblatt für die Ausbildung und Zertifizierung der Sicherheitsauditoren von Straßen, Arbeitsgruppe SStraßenentwurf"* (Techn. Ber.). Forschungsgesellschaft für Straßen- Verkehrswesen. An Lyskirchen 14, 50676, Köln.

MEISTER, A., SCHOLZ, F., & BANEMANN, S. (2021). *Masterplan BIM Bundesfernstraßen* (Techn. Ber.). Bundesministerium für Verkehr und digitale Infrastruktur. Invalidenstraße 44, 10115 Berlin.

MÖSER, K. (2002). *Geschichte des Automobils*. Campus Verlag.

OSSENKOPP, M. (2018). *Erste Ampel der Welt vor 150 Jahren*. Verfügbar 18. August 2022 unter https://www.stuttgarter-nachrichten.de/inhalt.erste-ampel-der-welt-erste-ampel-der-welt-vor-150-jahren.82c0eb6b-d81b-4c14-8a39-844fd1d2a81c.html

POSMIK, K. (2010). Einführung der Gurtpflicht: Anschnallen bitte! *Spiegel*. https://www.spiegel.de/geschichte/einfuehrung-der-gurtpflicht-a-946925.html

PREIDEL, C., & BORRMANN, A. (2015). *Automated Code Compliance Checking Based on a Visual Language and Building Information Modeling* (Techn. Ber.). Chair of Computational Modeling und Simulation, Technische Universität München. München.

PREIDEL, C. F. D. (2020). *Automatisierte Konformitätsprüfung digitaler Bauwerksmodelle hinsichtlich geltender Normen und Richtlinien mit Hilfe einer visuellen Programmiersprache* (Diss.). Technische Universität München. München, BY.

RAMPF, F. (2017). *Development of an IFC Alignment Import Export Plug-In for Autodesk AutoCAD Civil 3D* (Magisterarb.). TUM. Munich.

REICHSGESETZBLATT. (1934). *Reichs-Straßenverkehrs-Ordnung nebst Einführungsverordnung* (Techn. Ber. Nr. 59). Reichsministerium des Innern. Berlin.

REICHSGESETZBLATT. (1937). *Reichs-Straßenverkehrs-Ordnung nebst Einführungsverordnung* (Techn. Ber.). Reichsministerium des Innern. Berlin.

RSAS. (2019). *Richtlinie für das Sicherheitsaudit von Straßen, Arbeitsgruppe SStraßenentwurf"* (Techn. Ber. StB 11/7122.1/4/2985041). Forschungsgesellschaft für Straßen- Verkehrswesen. An Lyskirchen 14, 50676, Köln.

SCHADE, J., & ENGELN, A. (2008). *Fortschritte der Verkehrspsychologie*. VS Research.

SCHNIEDER, E., & SCHNIEDER, L. (2013). *Verkehrssicherheit - Maße und Modelle, Methoden und Maßnahmen für den Straßen- und Schienenverkehr*. Springer Vieweg.

SCHRANZ, C., GERGER, A., & URBAN, H. (2020). Augmented Reality im Bauwesen: Teil 1 - Anwendungs- und Anforderungsanalyse. *www.bauinger.de*, 379–388.

SCHUBERT, A. (2022). Wie der Zebrastreifen in die Bundesrepublik kam. *Süddeutsche Zeitung*.

SPIEGEL. (2001). Die deutsche Hauptuntersuchung. *Spiegel*. https://www.spiegel.de/auto/aktuell/50-jahre-tuev-die-deutsche-hauptuntersuchung-a-169239.html

SPIEGEL. (2008). *Der Lappen, der die Welt bedeutet*. Verfügbar 19. August 2022 unter https://www.spiegel.de/geschichte/120-jahre-fuehrerschein-der-lappen-der-die-welt-bedeutet-a-949511.html

STATISTA. (2000). *Entwicklung der Anzahl der Kraftfahrzeuge in Deutschland 1906-59*. Verfügbar 17. August 2022 unter https://de.statista.com/statistik/daten/studie/249900/umfrage/historische-entwicklung-von-kraftfahrzeugen-in-deutschland/#professional

TULLO, C. (1988). *Road Traffic Act chapter 52* (Techn. Ber.). The Stationery Office Limited Acts of Parliament. London UK.

UNITEDBIM. (2022). *Leading Countries With BIM Adoption*. Verfügbar 22. Juli 2022 unter https://www.united-bim.com/leading-countries-with-bim-adoption

WERTHER, B. (2006). *Kognitive Modellierung mit Farbigen Petrinetzen zur Analyse menschlichen Verhaltens* (Diss.). Braunschweig. Braunschweig.

WÜST, W. (2019). *Technische Regelwerke, Richtlinien für das Sicherheitsaudit von Straßen* (Techn. Ber.). Bayerisches Staatsministerium für Wohnen, Bau und Verkehr. Postfach 22 12 53, 80502, München.

ZAHLENBILDER. (2020). Pkw-Bestand in Deutschland 1950-2020. *Zahlenbilder Bergmoser + Höller Verlag AG*, *53*(400530).

ZHANG, J., & EL-GOHARY, N. M. (2017). Integrating semantic NLP and logic reasoning into a unified system for fully-automated code checking. *Automation in Construction*.

Eidesstattliche Erklärung

Ich versichere hiermit, dass ich die von mir eingereichte Abschlussarbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

Ort, Datum, Unterschrift